Khaue Rezende Rodrigues

DIInCX: Uma abordagem para Descoberta de Restrições de Integridade Semântica Implícitas em Dados XML

Florianópolis – SC
2007

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Khaue Rezende Rodrigues

DIInCX: Uma abordagem para Descoberta de Restrições de Integridade Semântica Implícitas em Dados XML

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Ronaldo dos Santos Mello
Orientador

Florianópolis, Dezembro de 2007

DIInCX: Uma Abordagem para Descoberta de Restrições de Integridade Semântica Implícitas em Dados XML

Khaue Rezende Rodrigues

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____

Prof. Dr. Mario Antônio Ribeiro Dantas

Coordenador do curso

Banca Examinadora

_____

Prof. Dr. Ronaldo dos Santos Mello

orientador

_____

Prof. Dr. Frank Augusto Siqueira

_____

Profa. Dra. Renata de Mattos Galante

_____

Profa. Dra. Carina Friedrich Dorneles

AGRADECIMENTOS

Pensei em diversas formas de iniciar esta seção, pensei em diversas pessoas que poderia agradecer, e acho que um bom começo para este capítulo é iniciar por quem me me acompanhou desde o inicio dessa caminhada de aprendizado, minha família. Não tenho como agradecer com palavras aos meus pais, o seu Inácio e a dona Marta, que sempre me incentivaram e ajudaram de todas as formas, inclusive abrindo mão do único carro para que eu pudesse chegar a tempo nas aulas que coincidiam com o horário de trabalho. Com meu pai aprendi a ter paciência, às vezes comigo mesmo. Gostaria de agradecer aos meus avós, o seu Ivo e dona Iolanda e a minha *"dindinha"* Ivani. Agradeço ao meu avô por me ensinar que o conhecimento é a única coisa que permanece sempre conosco. Com minha mãe e minha avó aprendi que nem só de conhecimento se faz o mundo, sem pessoas com corações tão grandes como os delas o mundo seria racional demais. Obrigado!

Gostaria de dedicar um parágrafo inteiro para agradecer à minha esposa Cris, que não só faz parte da minha família mas com quem formei um nova família. Agradeço pela compreensão a minhas ausências em muitos momentos. Espero que os obstáculos que tive durante o andamento de meu curso de mestrado não a desencorage a prosseguir na área acadêmica e a realizar o dela! Te amo Pequena!

Agradeço ao meu orientador Ronaldo pela amizade e confiança nessa jornada. Suas contribuições e críticas fizeram com que esta dissertação se tornasse realidade. Agradeço também aos membros da banca pelas sugestões e críticas.

Agradeço a todo o grupo de banco de dados da UFSC pelas sugestões. Desejo sucesso a todos! Não poderia deixar de agradecer à minha segunda família, meus amigos, pelo incentivo quando nem tudo ocorria no tempo que esperava. Aos meus amigos Rodrigo e Aguirre, amigos do início ao fim do mestrado, agradeço pelas palavras de incentivo quando o cansaço batia.

Gostaria de concluir agradecendo a todas as pessoas que me ajudaram de alguma forma durante o desenvolvimento deste trabalho. Obrigado!

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

HTML	Hyper Text Markup Language
XML	eXtensible Markup Language
RI	Restrição de Integridade
RIS	Restrição de Integridade Semântica
RISi	Restrição de Integridade Sintática
OWL	Ontology Web Language
DIInCX	Discovery of Implicit Integrity Constraint from XML data
CDX-Tree	Complete Domain XML Tree
SGBD	Sistema Gerenciador de Bancos de Dados
SWRL	Semantic Web Rule Language
DTD	Document Type Definition
W3C	World Wide Web Consortium
XSLT	XML Stylesheet Language Transform
SII	Sistema de Integração de Informação
RA	Regra de Associação

RESUMO

A *Web* tem sido adotada como uma grande fonte e meio para troca de informações. No entanto, os dados presentes nela se encontram sob os mais variados modelos de dados, principalmente XML. Em função do amplo uso do modelo de dados XML, questões como a descoberta de conhecimento e a manutenção da integridade sobre dados XML têm crescido em importância. A descoberta de conhecimento é relevante no suporte a decisões, enquanto a manutenção da integridade visa manter este conhecimento consistente. Dada esta relevância, é proposta uma abordagem semi-automática para descoberta de Restrições de Integridade Semânticas (RIS) a partir de instâncias XML chamada *DIInCX* (*Discovery of Implicit Integrity Constraint from XML data*).

DIInCX define um processo que coleta informações sobre as instâncias XML, aplica um algoritmo de mineração de regras de associação com adaptações e traduz as regras descobertas para RIS's especificadas na linguagem *SWRL* (*Semantic Web Rule Language*). A abordagem provê suporte a sistemas de manipulação de dados que desejam gerenciar RIS's. Outra contribuição deste trabalho é uma taxionomia para RIS's XML segundo os componentes de uma RI em bancos de dados. Esta taxionomia auxilia na avaliação de expressividade de linguagens de especificação de RIS's XML e da robustez de sistemas que controlam RIS's.

Palavras-chave: Restrições de integridade, XML, DIInCX.

ABSTRACT

Web today is a very large data repository as well as a vehicle for information interchange. However, data on the Web are defined over many data models, mainly XML model. Because of the broad use of the XML data model, issues like knowledge discovery and integrity maintenance have been increasing in importance. Knowledge discovery is relevant for decision support, while integrity maintenance focus on maintain this knowledge consistent. Given such motivation, this work proposes a semi-automatic approach for discovery of XML Semantic Integrity Constraints (SIC), based on XML instances, called DIInCX (Discovery of Implicit Integrity Constraint from XML data).

DIInCX defines a process that collects information about XML instances, applies an association rule mining algorithm with proposed adaptations, and translates the discovered association rules to SIC's defined in SWRL (Semantic Web Rule Language). The proposed approach provides a support to information systems that need to manage SIC's. Another contribution of this work is a taxonomy for XML SIC's based on the components of a database IC. This taxonomy aims at helping in the expressivity evaluation of XML SIC's language specification as well as robustness evaluation of information systems that manage SIC's.

Keywords: *Integrity constraints, XML, DIInCX.*

1 INTRODUÇÃO

Cada vez mais a *Web* se torna uma grande fonte e meio para troca de informações. No entanto, os dados existentes nela apresentam alta heterogeneidade, estando distribuídos sob os mais variados formatos e modelos de dados, principalmente dados semi-estruturados. Dentre estes, destacam-se a linguagem HTML (*Hyper Text Markup Language*) e mais recentemente, a linguagem *XML* (*eXtensible Markup Language*) (XML, 2007). No entanto, há uma relevante diferença entre estas: a linguagem XML apresenta um maior poder de expressão, além de apresentar uma definição clara do conteúdo representado, enquanto a HTML foca em questões envolvendo a apresentação do dado (NAYAK et al., 2002). Este fato tem transformado a linguagem XML em um padrão emergente para troca e representação de dados semi-estruturados na Web.

Em função do amplo uso da XML, questões como a descoberta de conhecimento e a manutenção da integridade sobre dados XML têm crescido em importância (BUNEMAN et al., 2001; FAN, 2005). A descoberta de conhecimento é relevante no suporte a decisões, nas mais diversas áreas (BÜCHNER et al., 2000), enquanto a manutenção da integridade visa manter este conhecimento consistente (CODD, 1980). Por descoberta de conhecimento entende-se extrair tanto informação explicitamente declarada, através da estrutura ou regras inerentes ao modelo de dados, quanto informação implícita existente em fontes de dados. Por manutenção de integridade entende-se a garantia da consistência dos estados válidos dos dados e das possíveis transições de estado entre estes.

Por conseguinte, a fim de prover a garantia da integridade, é usual a definição de regras específicas ou Restrições de Integridade (RI). De fato, CODD (1980) destaca a importância das RI's definindo-as como um conceito básico de um modelo de dados. Entende-se desta forma, que além de ter como função a manutenção de conhecimento, as RI's por si só representam uma forma de conhecimento, uma vez que fazem parte do modelo de dados. Com isto, ressalta-se a importância das RI's na definição de um modelo de dados baseado na linguagem XML. Por conseguinte, RIS's quando observadas sobre dados semi-estruturados como XML, apresentam maior complexidade

do que sobre dados estruturados. Este fato se deve principalmente ao formato irregular inerente ao modelo de dados XML.

Em se tratando de RI`s, este trabalho distingue RI's sintáticas de RI's semânticas. RI`s sintáticas (RISi) consideram a garantia da consistência da estrutura, por exemplo, não deve haver espaços na composição do nome de uma tag XML. Enquanto, as RI's semânticas (RIS) consideram o real significado do dado, por exemplo, uma estrutura em lista indicando uma ordem de relevância entre seus elementos. Considerando a alta importância das RIS, este trabalho foca no seu tratamento para dados XML, pois estas têm como finalidade a garantia da manipulação consistente dos dados XML relacionados ao domínio da aplicação.

No que tange à descoberta de conhecimento, a descoberta de RIS's e sua posterior incorporação a esquemas de dados traria inúmeros benefícios, acrescentando semântica e um maior grau de consistência a modelos de dados. De fato, observa-se que a especificação exaustiva de RIS's por um usuário especialista é um processo difícil, seja pela falta de conhecimento do mesmo a respeito do domínio ou pela grande quantidade de RIS's a serem especificadas para o domínio a fim de torná-lo completo no que tange a sua manutenção de integridade.

Neste contexto, este trabalho propõe uma abordagem semi-automática para descoberta de RIS implícitas a partir de instâncias XML chamada *DIInCX*[1]. A abordagem proposta compreende um processo composto por três fases: *Pré-Processamento*, *Descoberta* e *Conversão*.

A fase de Pré-processamento consiste na aplicação de um conjunto de regras cuja intenção é uniformizar e simplificar a estrutura de um conjunto de instâncias XML, removendo elementos ou atributos que não auxiliam no processo de descoberta de RIS. Esta fase é responsável por construir uma instância de um esquema conceitual hierárquico chamada CDX-Tree (Complete Domain XML Tree), que é especificada em OWL (Ontology Web Language) (OWL, 2007). Esta instância do esquema representa a estrutura simplificada das instâncias XML e serve de guia para as próximas fases da abordagem. O esquema conceitual CDX-Tree é discutido em detalhes na seção 5.1.

---

[1] ***DIInCX*** é um acrônimo para **D**iscovery of **I**mplicit **In**tegrity **C**onstraint from **X**ML data.

A fase de Descoberta é baseada na aplicação de um algoritmo adaptado de mineração de regras de associação. Os dados de entrada para o algoritmo são tomados a partir das instâncias XML e acessados através dos conceitos presentes na instância do esquema CDX-Tree. A fase é dita baseada em um algoritmo adaptado de mineração de regras de associação, pois a este são propostas adaptações que proporcionam a descoberta de regras de associação que possam ser traduzidas para RIS's de menor complexidade.

A fase de Conversão é responsável por traduzir as regras de associação descobertas para RIS especificadas em uma linguagem de representação de conhecimento chamada *SWRL* (*Semantic Web Rule Language*) (SWRL, 2007). A SWRL foi escolhida devido ao seu alto nível de abstração, ao invés de uma linguagem de definição de RIS's específica. Adotar linguagens de definição de RIS's específicas tais como a DTD (DTD, 2007), XML Schema (XML SCHEMA, 2007) ou outras, é considerado um ponto negativo de trabalhos relacionados (NESTOROV et al., 1998; HACID et al., 2000; CASTANO et al., 2002; CHIDLOVSKII, 2002; LIU et al., 2004; HEGEWALD et al., 2006) conforme discutido no capítulo 4.

1.1 Hipóteses de pesquisa

Um desafio da comunidade de pesquisa atual é manter consistente os dados em formato XML, seguindo não apenas regras simples como formatos de dados primitivos, mas regras de negócio complexas, como atualmente é possível definir sobre modelos de dados estruturados. Estas regras de negócio complexas são usualmente definidas através de RIS's e, no contexto XML, são consideradas por poucos sistemas de informação.

Este trabalho propõe uma abordagem para descoberta de RIS implícitas a partir de instâncias XML. O propósito é auxiliar no suporte a sistemas de informação, tais como Sistemas Gerenciadores de Bases de Dados XML (SGBD XML) e Sistemas de Integração de Informação XML (SII XML) que pretendam gerenciar RIS's. De fato, a incorporação de RIS's por estes sistemas traria inúmeros benefícios, como por exemplo, esquemas de dados mais consistentes.

1.2 Objetivos

Objetivo Geral

Este trabalho visa desenvolver uma abordagem capaz de descobrir informação implícita na forma de restrições de integridade semânticas a fim de complementar a semântica de fontes de dados XML pré-existentes.

Objetivos específicos

Os objetivos específicos deste trabalho são:

- **Definir uma abordagem para descoberta de RIS's implícitas.** A abordagem proposta define um processo que é capaz de descobrir RIS's, a partir de instâncias XML em um mesmo domínio de aplicação. O processo consiste de uma técnica de mineração de dados conhecida como mineração de regras de associação e da análise da instância de um esquema conceitual hierárquico definido a partir da estrutura das instâncias XML;

- **Buscar uma abordagem tão automatizada quanto possível.** O processo adota como dados de entrada apenas as instâncias XML, pois parte da premissa de que a representação de RIS's, envolvidas num referido domínio através de esquemas XML, é incompleta. Isto ocorre pela limitação de conhecimento do usuário especialista que define o esquema, ou ainda pela falta de expressividade de linguagens de especificação de esquemas. O usuário que interage com a abordagem apenas fornece parâmetros de entrada e seleciona, dentre as RIS's descobertas, as mais relevantes para um dado domínio;

- **Estabelecer uma forma de representação de alto nível para as RIS's descobertas.** A representação das RIS's descobertas através da SWRL e sua adoção como linguagem para especificação de RIS's objetiva representar as RIS's em uma linguagem com alto nível de abstração. O propósito é obter alto poder de expressão e facilitar sua posterior tradução para linguagens específicas de definição de RIS's;

- **Elaborar uma taxionomia para RIS's XML a fim de definir o escopo da abordagem e auxiliar na avaliação de expressividade de linguagens de**

especificação de RIS. A partir do estudo de trabalhos relacionados, observa-se que poucos trabalhos apresentam uma taxionomia completa para RIS's XML, embora alguns apresentem poucas categorias. Não havendo uma taxionomia consolidada no contexto XML para delimitar a abrangência das RIS descobertas, é proposta uma taxionomia com base na análise dos componentes de uma RIS.

1.3 Justificativas

O principal objetivo deste trabalho é desenvolver uma abordagem capaz de descobrir informação implícita a fim de complementar fontes de dados pré-existentes com a agregação de semântica na forma de RIS's. O foco da abordagem proposta são as RIS's implícitas, uma vez que RIS's explícitas representam conhecimento já presente no modelo de dados ou esquemas associados a instâncias XML. Já RIS's implícitas representam um novo conhecimento que tem por função complementar a semântica de domínios de dados. Conseqüentemente, a abordagem se apresenta particularmente útil para fontes de dados sem esquemas, além de servir como complemento a fontes de dados que possuem esquemas definidos.

A abordagem proposta busca prover um melhor suporte a sistemas de informação no que tange a incorporação de RIS's. De fato a incorporação de RIS's pode trazer contribuições a sistemas de informação nas mais diversas áreas de pesquisa. No contexto do modelo de dados XML destacam-se SII's XML (tanto de instâncias quanto de esquemas) e SGBD's XML. Dentre as contribuições às áreas de pesquisa citadas, destacam-se: *(i)* uma maior consistência de esquemas de dados XML; *(ii)* sistemas de consultas mais "*robustos*"; e *(iii)* maior precisão na integração quando os dados estiverem acompanhados de RIS's. Por exemplo, em sistemas baseados em mediadores para o acesso a fonte de dados XML heterogêneas, a análise de RIS's pode evitar o acesso a fontes com dados irrelevantes para a consulta.

No entanto, no contexto de gerência de RIS's sobre dados XML, ainda há diversas questões em aberto ou sendo trabalhadas, onde se destacam: *(i)* técnicas de mineração de dados (*data mining)* aplicadas a dados XML (tanto a sua estrutura quanto ao seu conteúdo) (NAYAK et al., 2002; BRAGA et al., 2002; GARBONI et al., 2006); *(ii)* abordagens sobre recuperação de informação e extração de informação (SMITH &

LOPEZ, 1997; EMBLEY et al., 1999; NAYAK et al., 2002; BRAGA et al., 2002; GARBONI et al., 2006); *(iii)* a incorporação e uso de RI's na otimização de processos em SII's e SGBD's (REYNAUD et al., 2001; SCHÖNING, 2001; ERDMANN & STUDER, 2001; MEIER, 2002; CALI et al., 2002; DELOBEL et al., 2003; CRUZ et al., 2003; WIWATWATTANA et al., 2003; LETHI & FANKHAUSE, 2004; MELLO & HEUSER, 2005); *(iv)* problemas de decidibilidade e satisfabilidade sobre RI's (ARENAS et al., 2002a; ARENAS et al. 2002b; FAN & SIMÉON, 2003; DEUTSCH & TANNEN, 2003); *(v)* o uso de RI's na otimização de consultas (problemas de reformulação/decomposição de consultas) (DEUTSCH & TANNEN, 2003; MA & SCHEWE, 2003).

Os itens *(i)* e *(ii)* estão diretamente relacionados à abordagem proposta, enquanto o item *(iii)* é relacionado com as contribuições da mesma. Embora os itens *(iv)* e *(v)* estejam fora do escopo da abordagem, ambos são discutidos na conclusão deste trabalho, na forma de trabalhos futuros. No mais, a fim de mensurar as RIS's implícitas descobertas a partir da abordagem proposta, é apresentada uma taxionomia para classificar as RIS's segundo os seus componentes, de acordo com SANTOS (1980).

1.4 Organização dos capítulos

Este trabalho está organizado da seguinte forma. O segundo capítulo apresenta a base conceitual envolvendo RI's, necessária para a compreensão da abordagem proposta. Ainda neste capítulo é apresentada uma taxionomia para RIS sobre o modelo de dados XML e uma breve exemplificação de sua aplicação sobre linguagens de especificação de RIS's e SGBD's XML.

O terceiro capítulo apresenta alguns conceitos básicos relacionados à mineração de dados com foco na categoria denominada de mineração de regras de associação, empregada na abordagem. O quarto capítulo apresenta os trabalhos relacionados presentes na literatura que se relacionam com a abordagem. O foco do capítulo é apresentar o estado da arte no que tange RIS's através das limitações e contribuições dos trabalhos relacionados.

O quinto capítulo detalha a abordagem DIInCX, seu conceitos básicos e o processo exemplificando o mesmo a cada etapa. O sexto capítulo apresenta um estudo

de caso ilustrando a abordagem. Finalmente, o capítulo sete apresenta as conclusões, realçando contribuições e trabalhos futuros.

2 RESTRIÇÕES DE INTEGRIDADE E SUAS CATEGORIZAÇÕES

Este capítulo foi baseado no artigo *A Faceted Taxonomy of Semantic Integrity Constraints for the XML Data Model*, publicado no evento *International Conference on Database and Expert Systems Applications* (*DEXA*) (RODRIGUES & MELLO, 2007b), o qual apresenta uma proposta para taxionomia de RIS para o modelo de dados XML. Esta taxionomia teve por base um estudo sobre RIS realizado sobre o modelo de dados relacional (alguns trabalhos relacionados são apresentados na seção 2.1) e XML (trabalhos relacionados são apresentados na seção 4.1), realizado a fim de definir a expressividade necessária para a abordagem DIInCX. Este capítulo apresenta alguns conceitos básicos sobre manutenção de integridade, discutindo-os a partir de conceitos para o modelo de dados relacional e semi-estruturado.

O termo *manutenção de integridade* refere-se ao ato de garantir que fontes de dados se mantenham íntegras, ou seja, consistentes em seus estados possíves e transições entre eles. Por conseguinte, a fim de prover a garantia da integridade, é usual a definição de regras específicas, denominadas Restrições de Integridade (RI). SANTOS (1980) afirma que integridade é uma propriedade de um banco de dados que se refere à validade do seu conteúdo e à forma pela qual o mesmo foi alcançado. Ainda segundo o autor, uma restrição de integridade é uma regra que restringe o conjunto de estados íntegros e/ou o conjunto de transições válidas em um banco de dados. CODD (1980) define RI como um conceito básico de um modelo de dados. Entende-se, desta forma, que além de ter como função a manutenção de conhecimento, as RI's por si só representam uma forma de conhecimento, uma vez que fazem parte do modelo de dados. RI's quando observadas sobre dados semi-estruturados como o modelo de dados XML, apresentam maior complexidade do que sobre dados estruturados. Este fato se deve principalmente ao formato irregular inerente ao modelo de dados semi-estruturados.

Em se tratando de RI`s, é possível distinguir dois tipos não exclusivos: sintática e semântica (RODRIGUES & MELLO, 2007c). Elas são consideradas não exclusivas, pois diversos tipos de RI's sintáticas apresentam aspectos semânticos. Por exemplo, uma RI que imponha uma ordem entre elementos pode possuir tanto aspectos sintáticos

(forma de definir uma estrutura em lista) quanto semânticos (um grau de relevância entre estes elementos). Desta forma, enquanto RI`s sintáticas (RISi) consideram a garantia da consistência da estrutura, as RI semânticas (RIS) consideram o real significado do dado. Considerando sua alta importância, este trabalho foca no tratamento de RIS's para dados XML, pois estas têm como finalidade a garantia da manipulação consistente dos dados XML relacionados a um domínio da aplicação.

O trabalho de SANTOS (1980) decompõe uma RI segundo os seguintes componentes: *(i) restringente* - objetos de dados utilizados na especificação da restrição; *(ii) restringido* - objetos de dados sobre os quais se aplicam a restrição; *(iii) condição restritiva* - expressão lógica a ser validada que relaciona *restringentes* e *restringidos*; *(iv) pontos de verificação* - momento de início da validação da restrição; e *(v) ações de violação* - ações a serem executadas após a violação da restrição a fim de manter a integridade dos dados. A partir destes componentes, pode-se definir que a manutenção de integridade de dados é realizada através da validação em um *ponto de verificação* específico, de um *objeto restringido* considerando uma *condição restritiva* que compreende *objetos restringentes* e, por fim, executando *ações de violação* em caso de falha.

A seção 2.1 a seguir apresenta trabalhos relacionados no contexto do modelo de dados relacional que foram adotados como base para a taxionomia proposta. A seção 2.2 apresenta a proposta de taxionomia para o modelo de dados XML. Por fim a seção 2.3 apresenta, como forma de aplicação da taxionomia proposta, um experimento prático, que compara um conjunto de linguagens de especificação de RIS's e SGBD's XML.

2.1 Restrições de integridade no modelo relacional

A literatura clássica de banco de dados apresenta algumas categorias e propostas de taxionomias para RIS's (CODD, 1980; SANTOS et al., 1980; ELMASRI & NAVATHE, 2003; DATE, 2003; SILBERSCHATZ et al., 2005), sendo este tópico já amplamente discutido no contexto de modelo de dados estruturados, especialmente o modelo relacional de dados. Um trabalho recente faz uma "*aglutinação*" destas propostas heterogêneas com o objetivo de propor uma classificação (LAZZARETTI,

2005).

Em SANTOS et al. (1980) é apresentada uma taxionomia baseada nos seguintes aspectos: *origem* - significando o agente que impõe a restrição (requisito natural imposto pelo modelo de dados ou pela implementação de restrições); *substância* - significando a propriedade de uma restrição e seus propósitos (transição de estados, intenção, etc.); *forma da especificação* - significando a forma com que a restrição pode ser considerada (explicitamente declarada, implicitamente presente no modelo de dados ou uma conseqüência lógica de outra restrição) e *modo de aplicação* - significando características dinâmicas de uma restrição (ativação manual ou ativação automática, estratégia de inspeção direta ou estratégia de inspeção indireta).

Em DATE (2003) é apresentada uma taxionomia que classifica RI's conforme segue: *restrições de tipo* (*domínio*), *restrições de atributo*, *restrições de variáveis de relação* (*tuplas*), *restrição de banco de dados*, *restrições de transição de estados*, *restrições de chaves, restrições de integridade referencial*, e *restrições quanto ao momento de verificação* (o momento que a RI é verificada). Em CODD (1980) esta última categoria é chamada de *"pontos de integridade"*.

Em SILBERSCHATZ et al. (2005) as RI's são classificadas em: *restrições de domínio, restrições de chaves, restrições de formas de relacionamentos* e *restrições de integridade referencial*. ELMASRI & NAVATHE (2003) classifica as RI's da seguinte forma: *restrições de domínio, restrições de chave, restrições de entidade* e *restrições de integridade referencial*.

O trabalho de LAZZARETTI & MELLO (2005) propõe para o modelo relacional as seguintes categorias: *restrições de domínio* (*restrições de atributo, restrições de tipo, restrições de tupla, restrições de banco de dados* e *restrições de transição de estado*), *restrições de chav*e (*chaves candidatas, chaves primárias ou alternativas* e *chaves estrangeiras), restrições de integridade referencial, restrição de momento de verificação* e *restrições baseadas em eventos*.

Através destes trabalhos observa-se que, mesmo no contexto do modelo de dados relacional, já amplamente consolidado, não há consenso em relação a uma taxionomia para RI's. Entretanto, muitas categorias, como *Restrições de chaves* e

Restrições de integridade referencial, se mostram presentes em diversos trabalhos denotando um consenso dentre a maior parte dos autores.

2.2 Taxionomia para RIS XML

Esta seção detalha uma proposta de taxionomia para RIS XML, descrita em RODRIGUES & MELLO (2007b). A referida taxionomia visa suprir a falta de uma taxionomia consolidada para RIS no modelo de dados XML. Seu objetivo principal foi servir como forma de mensurar a qualidade da abordagem proposta, definindo claramente as categorias de RIS capazes de serem descobertas por ela.

Há na literatura algumas propostas de taxionomias para RIS's XML (PAVLOVA et al., 2000; JACINTO et al., 2002a; HU & TAO, 2004; LAZZARETTI & MELLO, 2005). Contudo, estas propostas se apresentam heterogêneas, com fraco embasamento teórico ou incompletas. A maioria das categorias presentes na literatura se apresenta restrita a um pequeno grupo de categorias do universo de RIS's para o modelo de dados XML, tais como as que envolvem integridade referencial (BUNEMAN et al., 2001; ARENAS et al., 2002a; KLARLUND et al., 2002; DEUTSCH & TANNEN, 2003; FAN & SIMÉON, 2003; NENTWICH, 2005). A importância de uma taxionomia para RIS XML abrangente é realçada pela dificuldade de avaliação da expressividade de linguagens de especificação de RIS, assim como de sistemas que dêem suporte a elas. A maioria das categorias ou taxionomias propostas (BUNEMAN et al., 2001; ARENAS et al., 2002a; JACINTO et al., 2002a; DEUTSCH & TANNEN, 2003; FAN & SIMÉON, 2003; HU & TAO, 2004; LAZZARETTI & MELLO, 2005) tem por função delimitar o escopo de trabalhos específicos, não sendo o principal objetivo deles definir uma classificação abrangente. Cada trabalho foca em apresentar apenas as categorias de RIS relacionadas às características que aborda, como por exemplo FAN & SIMÈON (2003), que apresenta categorias de RIS's envolvendo chaves e integridade referencial. No mais, uma taxionomia consolidada se demonstra de grande valia para avaliação do poder de expressão de SGBD's XML, no que diz respeito à expressividade do seu sistema de gerenciamento de RI's. Desta forma, as limitações de propostas presentes na literatura e a falta de um consenso com relação à classificação de RIS XML foram as motivações para a proposta de taxionomia para RIS XML, apresentada nesta seção.

A proposta de taxionomia para RIS's XML (mostrada na Figura 2.1) distingue diversas *facetas* de uma RIS. O conceito de faceta, como um componente de uma *classificação facetada*[2] (RANGANATHAN, 1967), permite classificar uma RIS sobre diferentes pontos de vista, com o propósito de prover uma base para a avaliação de sua expressividade. As facetas propostas consideram os componentes de uma RI (SANTOS, 1980) e conceitos para o modelo relacional de dados (CODD, 1980) em sua definição. Por conseguinte, as categorias que compõe cada faceta são baseadas em taxionomias e conceitos para o modelo de dados relacional (CODD, 1980; SANTOS et al., 1980; ELMASRI & NAVATHE, 2003; DATE, 2003; LAZZARETTI, 2005; SILBERSCHATZ et al., 2005), além de contribuições de trabalhos relacionados no contexto XML (PAVLOVA et al., 2000; ARENAS et al., 2002a; JACINTO et al., 2002a; DEUTSCH & TANNEN, 2003; FAN & SIMÉON, 2003; HU & TAO, 2004; LAZZARETTI & MELLO, 2005). No entanto, a taxionomia é tão simples e consistente quanto possível, propondo apenas as categorias necessárias para atingir as contribuições propostas e suplantar as limitações dos trabalhos relacionados. De fato, uma taxionomia facetada analisada a partir dos componentes de uma RI se mostra uma boa escolha, pois permite distinguir as RIS's com relação a suas principais propriedades. Propriedades estas que sistemas que pretendam dar suporte a RIS's devem considerar.

[2] A classificação facetada foi proposta por Ranghanathan (RANGHANATHAN, 1967) e argumenta que a informação pode ser analisada sobre diversos aspectos ou propriedades, denominadas pelo autor de *facetas*.

1. **Quanto ao tipo de limitação imposta**. Considera restrições a valores de dados em um documento XML
   - **Estado.** Restringem o valor de um elemento atômico, atributo ou conjunto destes.
     - **Estáticas**. Impõem a mesma limitação a qualquer instância do documento XML.
       - **Quantificada.** Impõem uma limitação quantificável ao valor de um elemento ou atributo, ou a conjunto destes;
       - **Qualificada.** Impõem uma limitação qualificável ao valor de um elemento ou atributo, ou conjunto destes;
       - **Existência dependente.** Limita a existência de um elemento, atributo à existência de um elemento/atributo, conjunto destes ou de seus valores.
     - **Dinâmicas.** Impõem diferentes limitações a cada instância do documento XML;
       - **Quantificada.** Idêntico à categoria Estática, aplicada a um contexto dinâmico;
       - **Qualificada.** Idêntico à categoria Estática, aplicada a um contexto dinâmico;
       - **Existência dependente.** Idêntico à categoria Estática, aplicada a um contexto dinâmico.
   - **Transição de estados.** Restringem as transições de valores de elementos, atributos ou conjunto destes.
     - **Estáticas**. Impõem a mesma limitação a transições de valores de elementos, atributos ou conjunto destes.
       - **Quantificada.** Impõem uma limitação quantificável à transição de valores de elementos, atributos ou conjuntos destes;
       - **Qualificada.** Impõem uma limitação qualificável à transição de valores de elementos, atributos ou conjuntos destes;
       - **Existência dependente.** Limita a existência de um elemento, atributo à existência de um elemento/atributo, conjunto destes ou de seus valores em um estado diferente do atual.
     - **Dinâmicas.** Impõem diferentes limitações a transições de valores de elementos, atributos ou conjunto destes;
       - **Quantificada.** Idêntico à categoria Estática, aplicada a um contexto dinâmico;
       - **Qualificada.** Idêntico à categoria Estática, aplicada a um contexto dinâmico;
       - **Existência dependente.** Idêntico à categoria Estática, aplicada a um contexto dinâmico;
2. **Quanto ao alcance.** Consideram a distância hierárquica entre os nodos em um documento XML ou repositório XML, envolvidos na especificação da restrição ou sendo restringidos.
   - **Item de dado.** Considera um único atributo ou elemento simples;
   - **Tupla.** Considera um conjunto de atributos e/ou de elementos simples de um elemento complexo;
   - **Elemento.** Considera um conjunto de atributos e/ou de sub-elementos de um mesmo elemento complexo;
   - **Documento.** Considera atributos e/ou elementos simples de elementos complexos distintos.
3. **Quanto à forma.** Considera a notação utilizada para definir a restrição.
   - **Restrições baseadas em expressões booleanas.** Avaliam um predicado retornando verdadeiro ou falso como resultado.
     - **Simples.** Consideram apenas um elemento/atributo;
     - **Composta.** Consideram mais de um elemento/atributo.
   - **Restrições baseadas em regras condicionais.** Possuem expressões booleanas embutidas, podendo ou não executar ações sobre o documento XML a fim de garantir a integridade semântica.
     - **Simples.** Considera somente uma estrutura condicional;
     - **Composta.** Considera mais de uma estrutura condicional (aninhamentos).
4. **Quanto ao momento da verificação.** Corresponde ao momento de início da validação da restrição.
   - **Imediata.** Valida a restrição no momento que uma operação ocorre sobre um dado;
   - **Postergada.** Valida a restrição em algum momento posterior.
5. **Quanto à ação a ser executada**. Considera o tipo de ação que a restrição executa ao ser violada.
   - **Informativas.** Apenas informa a violação ocorrida executando uma operação de restauração do estado anterior (*ROLLBACK*);
   - **Ativas.** Executa operações sobre o documento XML a fim de manter a integridade dos dados.

Fonte: Primária.

Figura 2.1 – Taxionomia para RIS XML.

Assim, na taxionomia apresentada, é proposta uma análise de RIS para os seguintes aspectos: *(i)* o *tipo de limitação* que a RIS é capaz de impor a elementos ou atributos – considerando o componente *restringido* de uma RI; *(ii)* o *alcance* da RIS,

isto é, a quantidade de nodos do documento XML envolvidos na especificação da restrição ou sendo restringidos – considerando os componentes *restringente* e *restringido*; *(iii)* a *forma* como ela se apresenta, significando a sintaxe da especificação da RIS – considerando o componente *condição restritiva*; *(iv) quanto ao momento da verificação* da restrição – considerando o componente *pontos de verificação*; e *(v) quanto à ação a ser executada* após a violação de uma restrição – considerando o componente *ações de violação*.

As principais contribuições da taxionomia proposta são: *(i)* flexível, por ser especificada como uma classificação facetada é facilmente extensível; *(ii)* maior abrangência em relação ao universo de RIS para documentos XML; *(iii)* apresenta um embasamento teórico para justificar os itens da classificação (baseado em propostas do modelo relacional com adaptações para o modelo XML e nas propostas existentes na literatura para o modelo de dados XML); e *(iv)* uma taxionomia que permite a avaliação da expressividade de linguagens de especificação de RIS XML e conseqüente comparação entre estas. No mais, a taxionomia proposta, conforme demonstrado nas seções seguintes, se mostra útil na avaliação de sistemas que dão suporte a RIS's quanto à expressividade destes em termos de RIS's.

As próximas subseções analisam a relevância de cada faceta e suas categorias, a fim de destacar as contribuições da taxionomia proposta, assim como limitações de trabalhos relacionados. Uma discussão mais detalhada sobre trabalhos relacionados se encontra no capítulo quatro. Ainda, a última seção deste capítulo apresenta uma análise comparativa de um conjunto de linguagens de especificação de RIS e SGBD's XML com o suporte da taxionomia proposta. O fragmento de documento XML mostrado na Figura 2.2 é utilizado como base para os exemplos adiante. Ele representa um documento com dados sobre um professor em específico, pertencente a um banco de dados de professores de uma universidade.

```
<professor firstName=”John” middleName=”Albert” lastName=”Data” >
    <salary>1000.00</salary>
    <graduated>
        <university>SCFU</university>
        <degree>doctor</degree>
   </graduated>
</professor>
```

Fonte: Primária.

Figura 2.2 – Fragmento de documento XML.

2.2.1 Faceta Quanto ao tipo de limitação imposta

A faceta *Quanto ao tipo de limitação imposta* é relacionada ao componente de uma RI denominado *restringido*. Seu objetivo é distinguir RIS's que impõem restrições sobre o estado de um dado (categoria *estado*) de RIS's mais complexas, que impõem limitações sobre transições de estados válidos de dados (categoria *transição de estados*). A distinção entre *restrições estáticas*, que impõem a mesma limitação sobre os dados e *restrições dinâmicas*, que são capazes de aplicar diferentes limitações, também é proposta. Estas categorias são relevantes para avaliar quão expressiva é uma linguagem de especificação de RIS's em relação ao tipo de limitação que a linguagem é capaz de impor. A XML Schema, por exemplo, não é capaz de representar restrições dinâmicas, enquanto que a *XCML* (*XML Constraint Markup Language*) (HU & TAO, 2004) sim (Tabela 2.1). Assim, observa-se que a XCML é mais expressiva que a XML Schema com respeito a esta faceta. Por sua vez, o SGBD XML *Tamino* (SCHÖNING, 2001) é capaz de especificar restrições de estado, enquanto o SGBD XML *eXist* (MEIER, 2002) e *Timber* (WIWATWATTANA et al., 2003) não provem tal suporte. Desta forma, estes sistemas são pouco expressivos com relação a esta faceta.

A categoria *estado* é baseada na categoria *domínio* presente em propostas para o modelo relacional de dados (ELMASRI & NAVATHE, 2003; DATE, 2003; LAZZARETTI & MELLO, 2005; SILBERSCHATZ et al., 2005), assim como trabalhos relacionados (JACINTO et al., 2002a; FAN & SIMÉON, 2003). A categoria chamada *transição de estados* é principalmente derivada de taxionomias para o modelo relacional de dados (SANTOS et al., 1980; DATE, 2003; LAZZARETTI & MELLO, 2005), sendo também discutida no contexto XML por LAZZARETTI & MELLO (2005).

A fim de comparar esta faceta com trabalhos relacionados, considera-se o seguinte exemplo de RIS: *"O salário de um professor não pode ser reduzido e deve ser*

maior que 900, caso o professor possua graduação". Considerando o fragmento de documento XML apresentado na Figura 2.2, observa-se que esta RIS depende da comparação entre um valor anterior e um novo valor do elemento *salary*, assim como da existência do elemento *graduated* para definir os estados válidos para o elemento *salary*. Com respeito aos trabalhos relacionados, trabalhos como de FAN & SIMÉON (2003) e JACINTO et al. (2002a), não são capazes de classificar adeqüadamente a RIS exemplificada, pois não distinguem RIS envolvendo transições de estado ou contextos dinâmicos. Em HU & Tao (2004), a RIS pode ser classificada como uma *restrição dinâmica* (*dynamic constraint*) em relação ao *tipo de limitação imposta* (*kind of imposed limitation*), porém esta categoria não distingue transições de estado. Em LAZZARETTI & MELLO (2005), é apresentada uma categoria para transições de estado, porém, não há categorias para distinguir RIS de caráter dinâmico. Segundo a taxionomia proposta, é possível classificar a RIS exemplificada como uma RIS de *transição de estados dinâmica quantitativa*, capturando assim a sua intenção completa.

2.2.2 Faceta Quanto ao alcance

A faceta *Quanto ao alcance* considera os componentes de uma RI denominados *restringido* e *restringente*. Esta faceta distingue os elementos ou atributos que compõem uma restrição (atuando seja como *restringido* ou *restringente*), quanto aos seus níveis de relacionamento (distância entre nós na estrutura hierárquica XML). Esta faceta é importante no suporte à avaliação de expressividade, pois ajuda a definir o tipo de relacionamento entre elementos, ou atributos XML, que a linguagem de especificação de RIS deverá dar suporte. Pelo mesmo motivo, é igualmente importante para SGBD's XML, na definição de um sistema de gerenciamento de RI's, indicando o nível de relacionamento permitido para elementos ou atributos XML envolvidos na especificação de uma RIS. Uma linguagem de especificação de RIS pode ser capaz de representar diferentes níveis de relacionamento entre os elementos envolvidos em sua especificação, para cada tipo de RIS presente em outras facetas.

A faceta *Quanto ao alcance* considera os diferentes níveis hierárquicos que podem ser verificados por uma RIS para dados XML. Para tanto, categorias para o modelo relacional de dados (DATE, 2003; ELMASRI & NAVATHE, 2003) como *tuplas*, *banco de dados* (DATE, 2003) e *entidade* (ELMASRI & NAVATHE, 2003),

assim como trabalhos relacionados no contexto XML (FAN & SIMÉON, 2003; ARENAS et al., 2002a; JACINTO et al., 2002a; LAZZARETTI & MELLO, 2005) foram tomados como base.

Para fins de comparação desta faceta com trabalhos relacionados, considera-se o seguinte exemplo de RIS: *"O nome de um professor não pode exceder 30 caracteres e precisa ser composto por um título, primeiro nome, nome do meio e último nome"*. Poucos trabalhos apresentam categorias para classificar uma RIS como esta. Os trabalhos de FAN & SIMÉON (2003) e HU & TAO (2004), por exemplo, não possuem categorias com significado similar ao desta faceta. Em JACINTO et al. (2002a), é possível classificar a RIS em uma categoria chamada *dependência entre dois nós de documentos* (*dependencies between two document nodes*). Entretanto, esta categoria considera apenas o relacionamento entre dois elementos. De igual forma, os trabalhos de ARENAS et al. (2002a) e LAZZARETTI & MELLO (2005) não apresentam categorias adequadas. Em ARENAS et al. (2002a), a RIS pode ser classificada em uma categoria genérica chamada multi-atributo (m*ulti-attribute*). Em LAZZARETTI & MELLO (2005), o exemplo pode ser classificado em uma categoria chamada *banco de dados*. No entanto, esta apenas distingue se o elemento ou atributo estão em um mesmo nível hierárquico ou não. Segundo a taxionomia proposta neste trabalho, é possível classificar o exemplo de RIS na categoria denominada *elemento*, pois ela envolve atributos e um subelemento de um mesmo elemento complexo. Desta forma, demonstra-se que a faceta proposta é mais detalhada que os trabalhos relacionados, apresentando quatro níveis de relacionamento possíveis para elementos envolvidos na especificação de RIS's XML.

2.2.3 Faceta Quanto à forma

A faceta *Quanto à forma* considera o componente de uma RI denominado *condição restritiva* e foi baseada em trabalhos que consideram aspecto similar ao desta faceta (BUNEMAN et al., 2001; FAN & SIMÉON, 2003; HU & TAO, 2004). A principal contribuição desta faceta é permitir distinguir RIS quanto à notação adotada em sua especificação. HU & TAO (2004) comentam a importância desta faceta para linguagens de especificação de RIS, afirmando que uma RIS baseada em *regras condicionais* permite uma especificação mais natural para muitos tipos de RIS do que

RIS baseadas em *expressões booleanas*. Este fato é especialmente importante para o projeto de linguagens de especificação de RIS's e como conseqüência, para SGBD's XML no que diz respeito ao seu mecanismo de gerenciamento de RI's. Isto por quê, a referida faceta representa como a RIS deve ser expressa pelo usuário e se é possível especificá-la na forma de *expressões booleanas* ou *regras condicionais*.

Embora alguns trabalhos relacionados abordem aspectos similares (BUNEMAN et al., 2001; FAN & SIMÉON, 2003), apenas HU & TAO (2004) considera tal faceta, sendo a faceta proposta neste trabalho baseada no trabalho de HU & TAO (2004). Entretanto, o trabalho de HU & TAO (2004) não considera as outras facetas abordadas pela taxionomia proposta neste trabalho. Uma discussão sobre diferentes tipos de notação e sua expressividade está fora do escopo deste trabalho. HU & TAO (2004) apresenta brevemente uma análise neste sentido.

A fim de exemplificar a faceta em questão, segue o seguinte exemplo de RIS: "*O salário de um professor deve ser 5000.00, se o professor tiver doutorado como título*". Esta faceta analisa um aspecto muito particular de uma linguagem de especificação de RIS. Embora a maioria das RIS's possa ser representada tanto através de expressões booleanas quanto de regras condicionais, é proposto que apenas as regras condicionais possam realizar operações de atualização a fim de manter um dado consistente. Contudo, o exemplo em questão pode ser definido como um gatilho (por exemplo uma *trigger* em um SGBD XML), que atualiza o elemento *salary* para *5000.00* quando o valor do elemento *degree* se tornar *"doctor"*. Desta forma, o exemplo dado é classificado segundo a taxionomia proposta, como pertencendo à categoria *baseada em regras condicionais simples*.

2.2.4 Faceta Quanto ao momento da verificação

A faceta *Quanto ao momento da verificação* considera o componente de RI's chamado *pontos de verificação*. Esta faceta distingue o momento em que a *condição restritiva* que compõe uma RI deve ser verificada. Considerando a proposta de DATE (2003) para o modelo relacional de dados, são definidas duas categorias para esta faceta, chamadas: *imediata* e *postergada*.

Uma RIS *postergada* é uma restrição que, no escopo de uma operação em uma

transação, deve ser verificada em algum momento posterior à execução desta operação. Tal aspecto é reforçado por CODD (1980) que argumenta que alguns tipos de RI devem permitir que um dado passe por estados inconsistentes até alcançar o novo estado consistente. O aspecto considerado pela faceta em questão é relevante na avaliação de mecanismos de controle de transações em SGBD's XML. O projetista de banco de dados pode definir quais categorias de RIS poderão ser tratadas como postergadas. Os SGBD's XML Tamino e eXist, por exemplo, dão suporte apenas a RIS's *imediatas*, não sendo capazes de representar RIS's *postergadas*.

Um exemplo no contexto em questão é dado a seguir: *"Dado que 80% dos professores de uma universidade devem ter o título de doutor, se uma atualização sobre um conjunto de elementos do tipo professor é realizada, a RIS que garante esta consistência deve ser verificada somente após todas as instâncias do elemento professor serem atualizadas"*. Na taxionomia proposta, este é um exemplo de RIS *postergada*, pois a mesma não deve ser verificada imediatamente após a atualização de cada elemento *professor*, mas apenas ao final da operação completa. Nenhum trabalho relacionado no contexto XML considera tal faceta.

2.2.5 Faceta Quanto à ação a ser executada

A faceta *Quanto à ação a ser executada* considera o componente de uma RI chamado *ações de violação*, e foi baseada em trabalhos sobre mecanismos de validação para linguagens de especificação de RIS XML (CLARK & MAKOTO, 2001; DODDS, 2001; JACINTO et al., 2002a; HU & TAO, 2004; LAZZARETTI & MELLO, 2005). A referida faceta considera o tipo de ação a executar quando uma violação de integridade ocorre.

Considere o seguinte exemplo: *"Um professor com título de mestre não pode ter salário menor que 4000.00"*. Supondo que um professor receba o valor *"master"* em seu elemento *degree* e seu salário não seja incrementado para *4000.00*, duas alternativas podem ser adotadas: *(i)* uma operação de atualização é executada a fim de incrementar o elemento *salary* (RIS *ativa*); ou *(ii)* uma mensagem de erro é gerada e uma operação de *rollback* é executada (RIS *informativa*). A execução destas alternativas depende do mecanismo de controle de transações e do sistema de gerenciamento de RI's em

SGBD's XML.

Os SGBD's XML Tamino e eXist, por exemplo, permitem a implementação apenas de RIS's *informativas*. Por sua vez, os mecanismos de validação das linguagens de especificação de RIS XML *XCML*, *XCSL* (*XML Constraint Specification Language*) (JACINTO et al., 2002a) e *XDCL* (*XML Domain Control Language*) (LAZZARETTI & MELLO, 2005) também consideram apenas RIS *informativas*. Com relação aos trabalhos relacionados, nenhuma das taxionomias propostas distingue o componente de RI's chamado *ações de violação*. Desta forma, a principal contribuição desta faceta é considerar as ações que uma RIS pode executar a fim de manter a integridade dos dados.

2.3 Aplicação da Taxionomia

A fim de exemplificar a aplicabilidade da taxionomia proposta, esta seção apresenta uma avaliação de algumas linguagens de especificação de RIS e SGBD's XML. A Tabela 2.1 apresenta as facetas da taxionomia e sua aplicação em uma análise de expressividade com relação a RIS XML.

Com base na Tabela 2.1, destacam-se a linguagem XDCL e o SGBD Tamino como sendo os mais "*robustos*" linguagem e SGBD XML, respectivamente, com respeito ao suporte dado às RIS's. Observa-se ainda com base na Tabela 2.1, que no contexto do modelo de dados XML, a maioria dos SGBD's não considera RIS como forma de otimizar seus processos, ou consideram fracamente.

Vale observar que a consideração de todas as facetas pelas abordagens apresentadas é limitada. Por exemplo, poucas linguagens de especificação de RIS são capazes de especificar RIS's envolvendo *transações entre estados* e/ou *regras condicionais compostas*, categorias capazes de serem representadas na maioria dos sistemas de gerenciamento de RI's para o modelo relacional de dados. De fato, categorias como estas são a muito abordadas no contexto de modelos de dados estruturados. Assim, nota-se que o suporte dado a categorias como *Quanto à ação a ser executada* e *Quanto ao momento da verificação* não é satisfatório no contexto XML, estando na maioria dos casos restrita a RIS de ações *informativas* e validações *imediatas*.

Tabela 2.1 – Taxionomia vs. SGBD's XML e linguagens de especificação de RIS.

Linguagem de especificação de RIS e SGBD's XML	Quanto à limitação imposta		Quanto ao alcance	Quanto à forma		Quanto à ação a ser executada	Quanto ao momento de verificação
	Estado	Transição de Estados		Expressão booleana	Regras condicionais		
1. XML Schema (XML SCHEMA, 2007)	Estática	Não	Elemento	Simples	Não	Informativa	Imediata
2. Schematron (DODDS, 2001)	Dinâmica	Não	Documento	Composta	Simples	Informativa	Imediata
3. XCSL (JACINTO et al., 2002a)	Dinâmica	Não	Documento	Composta	Simples	Informativa	Imediata
4. XCML (HU & TAO, 2004)	Dinâmica	Não	Documento	Composta	Composta	Informativa	Imediata
5. XDCL (LAZZARETTI & MELLO, 2005)	Dinâmica	Dinâmica	Documento	Composta	Composta	Ativa	Imediata
6. eXists XML DBMS (MEIER, 2002)	Não	Não	Não	Não	Não	Não	Não
7. Tamino XML DBMS (SCHÖNING, 2001)	Estática	Não	Elemento	Simples	Não	Informativa	Imediata
8. Timber XML DBMS (WIWATWATTANA et al., 2003)	**Não**	**Não**	**Não**	**Não**	**Não**	**Não**	**Não**
*A categoria atribuída a cada trabaho representa a categoria mais abrangente suportada na faceta em questão.							

Fonte: Primária.

Assim sendo, conclui-se que, no que diz respeito a RIS's, as abordagens presentes na literatura para SGBD's XML não se encontram em estágio maduro quando comparadas com soluções para modelos de dados estruturados. No entanto, nota-se que diversas soluções para linguagens de especificação de RIS têm surgido, assim como mecanismos de validação para estas, o que demonstra o interesse crescente da comunidade de pesquisa pelo tema em questão.

3 MINERAÇÃO DE DADOS

Este capítulo apresenta alguns conceitos básicos envolvendo a descoberta de informação através da *Mineração de Dados*. Estes conceitos são empregados na abordagem DIInCX. Ainda neste capítulo são apresentadas brevemente algumas técnicas de mineração dados, embora uma discussão mais detalhada possa ser encontrada na literatura, em trabalhos como AGRAWAL et al. (1993), AGRAWAL & SRIKANT (1994), GAROFALAKIS et al. (1999), BÜCHNER et al. (2000), BAYARDO (2004), CHI et al. (2005), dentre outros.

O avanço da tecnologia de armazenamento possibilitou o aumento da capacidade de dispositivos de armazenamento e conseqüente queda no valor destes. Este fato propiciou às empresas armazenar informação antes desperdiçada. De fato, o emprego de *Armazéns de dados* (*Data Warehouse*) (KIMBALL, 1996) popularizou o armazenamento de grandes massas de dados, mesmo em empresas com pouca infra-estrutura. O principal objetivo do armazenamento destas grandes massas de dados tem sido auxiliar no suporte à tomada de decisões. Entretanto, técnicas tradicionais de análise de dados não são adequadas para tratar volumes consideráveis de informação. Muitas vezes, o processo de análise destes se torna proibitivo quando analisado sob o fator tempo. Neste contexto, surgem então as técnicas de *Mineração de Dados,* como forma de analisar a um baixo custo de tempo, grandes massas de dados.

A *Mineração de Dados* (*Data Mining*) está usualmente inserida no contexto do processo de *Descoberta de Conhecimento em Banco de Dados* (*KDD - Knowledge Discovery in Database*) (AMO, 2004) e consiste na análise de conjuntos de dados a fim de descobrir padrões de informação implícitos aos dados. Estes padrões de informação, descobertos a partir de dados, referem-se a distribuições probabilísticas e podem ser representados de inúmeras formas, como por exemplo regras de associação (RA) entre objetos de dados ou agrupamentos destes. A mineração de dados pode receber suporte de diversas áreas de pesquisa como: Aprendizado de máquina, Inteligência artificial e Estatística. KDD é um processo amplo, sendo a Mineração de Dados uma de suas etapas. Em geral, o processo em si é semi-automático, necessitando da intervenção de um usuário especialista.

No contexto de documentos semi-estruturados, a aplicação de técnicas de mineração de dados traz novos desafios. Em NAYAK et al (2002), a mineração de dados é discutida no contexto de dados semi-estruturados, sendo definido o termo *XML Mining*. Segundo o autor, a XML Mining difere da mineração de dados tradicional, pois consiste na mineração tanto da estrutura de documentos XML (esquemas como XML Schema e DTD) como de seu conteúdo (valores de atributos e elementos). Definição semelhante é dada por SNOUSSI et al. (2002) para a extração de informação sobre dados semi-estruturados.

Em GAROFALAKIS et al. (1999) e NAYAK et al. (2002) são apresentadas técnicas de mineração de dados no contexto do modelo de dados semi-estruturados, tais como: Mineração de RA's, Mineração de árvores freqüentes, Classificação e Agrupamento. CHI et al. (2005) discutem técnicas de mineração de árvores freqüentes, traçando comparativos entre estas.

A seção 3.1 a seguir discute brevemente a mineração de RA's. Embora diversas técnicas de mineração possam ser empregadas na abordagem DIInCX, é adotada a mineração de RA's. Isto se deve ao fato da definição de uma RA apresentar semelhanças com a definição de uma RIS, o que propicia a tradução entre elas, conforme discutido na seção 3.2. Uma discussão mais detalhada das demais técnicas de mineração de dados está fora do escopo deste trabalho.

3.1 Mineração de regras de associação

A descoberta de RA's é uma importante técnica da Mineração de Dados e tem por objetivo descobrir padrões freqüentes como relacionamentos e co-ocorrências entre objetos ou conjuntos de dados. Uma RA é um padrão descritivo que representa a declaração de uma implicação na forma $X \rightarrow Y$. A mineração de RA's possui importante papel nas organizações. Áreas como marketing e estratégia empresarial são um exemplo clássico, pois permitem que estas sejam planejadas pontualmente. Como exemplo de uma regra, que poderia ser descoberta a partir de um banco de dados de transações de uma loja, seria o fato de que 70% dos consumidores que compram o produto X, também compram, na mesma ocasião, o produto Y.

O problema denominado de mineração de RA's, pode ser definido como segue (AGRAWAL & SRIKANT, 1994): Seja $I=\{i_1,...,i_m\}$ um conjunto de *m* itens de dados. Seja $D=\{t_1,...,t_n\}$ um conjunto de *n* transações de um banco de dados de transações. Cada transação possui um identificador único chamado *TID* e um *itemset I* (conjunto de itens). *I* é dito um *k-itemset*, onde k é o número de itens em *I*. Uma transação *T* contém *X*, onde *X* é um subconjunto de *I*. O suporte de um *itemset I* é a fração de transações em *D* contendo *I*, por exemplo, $suporte(I)=\{t \in D \mid I \subseteq t\}/\{t \in D\}$. Por sua vez, uma RA representa uma implicação da forma $I_1 \rightarrow I_2$, onde $I_1, I_2 \subset I$ e $I_1 \cap I_2 = \varnothing$. A regra $I_1 \rightarrow I_2$ se faz presente no conjunto de transações *D* com confiança *c* se *c%* das transações em *D* que contêm I_1, também contêm I_2. A regra $r:I_1 \rightarrow I_2$ têm suporte *s* no conjunto de transações *D* se *s%* das transações em *D* contêm $I_1 \cup I_2$.

Quanto aos componentes de uma dada RA $I_1 \rightarrow I_2$, o lado esquerdo da implicação é chamado de componente *antecedente*, enquanto o lado direito é chamado de componente *conseqüente*. Quanto aos componentes antecedente e conseqüente, este trabalho classifica-os ainda em simples e complexos. Por complexo antecedente/conseqüente significa que o componente é composto por mais de um *item*. Já um simples antecedente/conseqüente significa que este é composto por apenas um item. Por exemplo, a regra $X \rightarrow Y.Z$, seria classificada como uma regra de antecedente simples e conseqüente complexo composto por 2 itens.

3.2 Mineração de regras de associação na descoberta de RIS

A mineração de RA's é uma importante técnica empregada em diversas áreas do conhecimento. Na descoberta de RIS's esta técnica é de grande valia pois, como comentado anteriormente, sua definição possui estreita relação com a definição de uma RIS. Esta estreita relação foi a motivação para a escolha desta técnica para descoberta de RIS's na abordagem DIInCX.

Com respeito à definição de uma RIS XML, este trabalho aplica a definição de RI presente em SANTOS et al. (1980) para o modelo relacional de dados.

Definição: $Condição\_restritiva(Restringente) \rightarrow Restringido$ (1)

Onde *Restringente* representa um elemento ou atributo XML (objetos de dados), ou ainda conjuntos destes, usados na especificação da RIS, e *Restringido* é o conjunto de objetos de dados sobre os quais se aplica a restrição. *Condição restritiva* é a expressão lógica a ser avaliada que conecta os elementos *Restringente* e *Restringido* da definição.

Conforme apresentado na seção 3.1, uma RA pode ser definida como uma implicação na forma a seguir.

Definição: $Antecedente_n \rightarrow Consequente_n$ (2)

Onde *antecedente* e *conseqüente* são conjuntos de itens (também denominados *itemsets*) de tamanho *n*. No contexto XML, um item é definido como um elemento ou atributo XML. Cada RA pode ser descoberta através de um conjunto de transações *T*. Cada $t_n \in T$ representa um conjunto de valores de elementos ou atributos XML que pertence a uma instância XML. Um conjunto de instâncias XML pode ser representado como um banco de dados *D* ($T \subseteq D$).

Considerando que uma RIS pode ser compreendida como um subconjunto do conjunto de RA's e, como consequência, pode ser descoberta através de um subconjunto *T',* uma RIS XML descoberta pode ser definida como a seguir.

Definição: *Condição_restritiva(Antecedente)* → *Consequente* (3)

Onde o *conseqüente* corresponde ao componente restringido, o *antecedente* corresponde ao restringente e a *condição restritiva* representa uma expressão lógica. Cada componente restringente presente no antecedente de uma RA possui uma correspondente condição restritiva que pode ser expressa na forma de uma expressão booleana, significando, com relação ao tipo de limitação imposta*,* uma restrição *estática* de um dos tipos a seguir:

- *Restrição de existência dependente*: a existência de um elemento depende da existência de um elemento/atributo, ou conjunto destes, ou ainda de seus valores. Como exemplo, um elemento *professor* – *antecedente* - obrigatoriamente possui um atributo *degre*e – conseqüente - e seu valor não pode ser vazio;

- *Restrição quantificada*: o valor de um elemento ou atributo é maior que outro, um elemento que representa uma coleção de elementos tem sua quantidade de elementos filhos restringida pelo valor de um elemento, atributo, valor constante ou ainda outra coleção. Como exemplo, dependendo do valor do atributo *degree – antecedente*, o número de elementos filhos do elemento *course – conseqüente*, é limitado;

- *Restrição qualificada*: um elemento ou atributo possui seu valor restrito a um conjunto de valores ou seu valor é o mesmo de outro elemento ou atributo. Como exemplo o valor do atributo *degree* é limitado ao conjunto *V={"graduated","master","doctor"}*.

Análogo à classificação dos componentes antecedente/consequente de RA (seção 3.1), este trabalho classifica a complexidade do componente *condição restritiva* de uma RIS como sendo *complexa* ou *simples*. Condições restritivas simples são compostas apenas por uma expressão booleana, enquanto condições restritivas complexas são compostas por um conjunto de expressões booleanas. Segundo a classificação proposta, a RIS $X.Y \rightarrow Z$, teria sua componente condição restritiva classificada como complexa de tamanho 2. A mesma classificação se aplica ao componente restringido. Algumas vezes, é possivel decompor uma condição restritiva complexa em um conjunto de condições restritivas simples. Isto se dá localizando as condições restritivas simples que compõem a condição restritiva complexa no conjunto de RIS descobertas. Um exemplo de RIS descoberta a partir de uma RA ($X.Y \rightarrow Z = X \rightarrow Z \cup Y \rightarrow Z$) é *degree("doctor").position("director")→salary(">=3000To<=5000")* = *degree("doctor")→salary(">=3000To<=5000")* ∪ *position("director")→salary(">=3000To<=5000")*. A mesma observação se aplica ao componente consequente e tem como único objetivo tornar as regras descobertas mais simples (podendo ser incorporadas por sistemas com mecanismos de gerenciamento de integridade mais simples) e legíveis ao usuário especialista. Vale ressaltar que decomposição apenas é válida em contextos onde a avaliação de ambas as RIS's é tomada como atômica, ou seja dentro de uma única transação.

Com relação aos critérios de restrição usuais no contexto da mineração de RA's, esta abordagem considera os conceitos de *confiança* e *suporte*. O critério *suporte*

representa a *relevância* da RIS descoberta, significando que quanto maior o suporte da regra, maior sua relevância dentro do domínio. Quanto ao conceito *confiança*, ele significa o percentual de vezes que um antecedente X na RA $X \rightarrow Y$ ocorre em T' do tipo $X \rightarrow Y$, ou seja, o percentual de vezes que X provoca $X \rightarrow Y$. A fim de garantir que a RIS descoberta seja válida, o conceito de confiança deve ser obrigatoriamente 100%, significando que a RIS é válida para todo o domínio de dados. Uma $t_n \in T'$ que não respeite esta regra representa uma violação à RIS, invalidando-a. Desta forma, tomando como base que todas as RIS's descobertas são baseadas em RA's, descobertas a partir de um algoritmo de mineração de RA's com confiança igual a 100%, pode-se afirmar que todas as RIS descobertas são válidas para o dado domínio.

Este trabalho define por RIS válida toda RIS que seja considerada verdadeira quando avaliada sobre todo o conjunto T de transações pertencentes ao domínio de dados tomado como entrada para a construção da CDX-Tree. Em contrapartida, uma RIS é considera inválida quando há no domínio de dados uma transação que a torne falsa.

4 TRABALHOS RELACIONADOS

Este capítulo apresenta os trabalhos presentes na literatura que se relacionam com a abordagem proposta em alguma das áreas que esta abrange: restrições de integridade no modelo de dados XML, abordagens de descoberta de informação e sistemas de informação que se utilizam de RIS. O foco deste capítulo é apresentar o estado da arte no que diz respeito a RIS XML através das limitações e contribuições de trabalhos presentes na literatura, realçando as áreas que se relacionam com a abordagem.

A pesquisa sobre o modelo de dados XML tem ganho crescente importância. No entanto, dentre as áreas relacionadas à abordagem proposta ainda há diversas questões em aberto ou sendo trabalhadas, dentre as quais se destacam: *(i)* uma linguagem padrão para especificação de RIS - há diversas propostas (CLARK & MAKOTO, 2001; DODDS, 2001; JACINTO et al., 2002a; KLARLUND et al., 2002; HU & TAO, 2004; LAZZARETTI & MELLO, 2005; NENTWICH, 2005), porém não há unanimidade; *(ii)* uma taxionomia para RIS adequada para dar suporte à avaliação do poder de expressão de linguagens de especificação de RIS (PAVLOVA et al., 2000; JACINTO et al., 2002a; FAN & SIMÉON, 2003; HU & TAO, 2004; LAZZARETTI & MELLO, 2005); *(iii)* problemas de decidibilidade e satisfabilidade sobre RI's (ARENAS et al., 2002a; ARENAS et al. 2002b; DEUTSCH & TANNEN, 2003; FAN & SIMÉON, 2003); *(iv)* o uso de RI's na otimização de consultas (problemas de reformulação/decomposição de consultas) (DEUTSCH & TANNEN, 2003; MA & SCHEWE, 2003). *(v)* técnicas de mineração de dados (*data mining)* aplicadas a dados XML (tanto a sua estrutura quanto ao seu conteúdo) (NAYAK et al., 2002; BRAGA et al., 2002; GARBONI et al., 2006); *(vi)* abordagens sobre recuperação de informação e extração de informação (SMITH & LOPEZ, 1997; EMBLEY et al., 1999; BRAGA et al., 2002; NAYAK et al., 2002; GARBONI et al., 2006); *(vii)* a incorporação e uso de RI's na otimização de processos em SII's e SGBD's (ERDMANN & STUDER, 2001; REYNAUD et al., 2001; SCHÖNING, 2001; CALI et al., 2002; CRUZ et al., 2003; DELOBEL et al., 2003; LETHI & FANKHAUSE, 2004; MELLO & HEUSER, 2005);

Os itens *(i), (ii), (iii)* e *(iv)* são abordados na seção 4.1 que aborda RI's no modelo de dados XML, focando em RIS's. Esta seção busca traçar um estado da arte com relação a categorias de RIS consideradas em trabalhos presentes na literatura.

Os itens *(iii)* e *(iv)* estão intimamente relacionados, sendo discutidos na seção 4.2. Esta seção tem por objetivo realçar as contribuições e limitações destes trabalhos no que diz respeito a recuperação de informação que possa ser traduzida para informação semântica na forma de RIS.

Por sua vez, o item *(v)* é relacionado com as abordagens ou processos que podem tirar proveito da incorporação de RIS. Este item é discutido na seção 4.3 e alguns trabalhos sobre SGBD-XML e SII-XML são comentados no que tange à utilização de RIS.

4.1 Restrições de integridade no modelo de dados XML

Esta seção tem por objetivo distinguir as categorias de RIS's XML presentes na literatura, traçando assim um panorama destas no contexto XML. Esta seção também é a base para o estudo realizado a fim de definir uma taxionomia para RIS XML, expressiva o suficiente, para avaliar linguagens de especificação de RIS. Como conseqüência, este estudo também foi usado como base para definir as categorias de RIS a serem tratadas na abordagem DIInCX. Ainda, são apresentadas limitações e contribuições dos trabalhos relacionados dentro dos objetivos traçados para a abordagem DIInCX.

Dentre os trabalhos presentes na literatura que lidam com categorias de RIS, alguns focam em problemas relacionados à decidibilidade e satisfabilidade de RI's (ARENAS et al., 2002a; ARENAS et al. 2002b; DEUTSCH & TANNEN, 2003; FAN & SIMÉON, 2003). Outros consideram a análise de RIS's em consultas e/ou armazenamento sobre documentos XML (TATARINOV et al., 2001; DEUTSCH & TANNEN, 2003; MA & SCHEWE, 2003). Um terceiro grupo aborda linguagens e mecanismos de validação para RIS's (CLARK & MAKOTO, 2001; DODDS, 2001; JACINTO et al., 2002a; KLARLUND et al., 2002; HU & TAO, 2004; LAZZARETTI & MELLO, 2005; NENTWICH, 2005), motivados principalmente por limitações da XML Schema (DODDS, 2001; JACINTO et al., 2002a; BUNEMAN, 2003; HU &

TAO, 2004). Um quarto grupo aborda a expressividade de linguagens de especificação de RIS, apresentando comparativos entre linguagens (LEE & CHU, 2000; JACINTO et al., 2002b; HU & TAO, 2004; LAZZARETTI & MELLO, 2005), ou buscando para tal avaliação, base em outras áreas de pesquisa, tal como na teoria da gramática de árvores regulares (MURATA et al., 2005). Dentre este último grupo, a fim de traçar um comparativo do poder de expressão de linguagens de especificação, alguns propõem taxionomias para RIS's XML ditas "completas" (PAVLOVA et al., 2000; JACINTO et al., 2002a; HU & TAO, 2004; LAZZARETTI & MELLO, 2005).

Com relação aos trabalhos que abordam decidibilidade e satisfabilidade de especificação de RI's, o trabalho de ARENAS et al. (2002a), a fim de distinguir categorias de RIS com relação à decidibilidade, considera as seguintes categorias: *Restrições de integridade absolutas* e *Restrições de integridade relativas* - distinguindo restrições de integridade que requerem uma análise completa do documento XML das que requerem a análise parcial ao serem validadas - e *Restrições de integridade unárias* e *Restrições de integridade multi-atributos* – distinguindo restrições de acordo com o número de atributos envolvidos. Outras categorias específicas são também apresentadas: *Restrição de integridade de chave*, *Restrição de integridade de chave estrangeira*, *Restrição de integridade de cardinalidade* e *Restrições de integridade baseada em expressões regulares*.

O trabalho de FAN & SIMÉON (2003) apresenta um estudo de complexidade e decidibilidade da classe de problemas sobre RI denominada *problemas de implicação de restrições*. Esta classe de problemas aborda RIS com relação à decidibilidade de sua validação. Ele propõe três linguagens de especificação de RIS em níveis crescentes de expressividade e analisa suas decidibilidades. A fim de delimitar o escopo do estudo, é considerado um subconjunto do universo de RI's denominado pelo autor de *Restrições básicas XML*. As categorias de RI's presentes no trabalho são exemplificadas na Figura 4.1. No mais, são também elencadas algumas categorias de RI's originadas de modelos de dados estruturados.

- **Restrições básicas XML**
  - **Chave.** Identificação única definida pelo valor da chave e identidade do nó (a localização do nó na estrutura hierárquica do documento XML)**;**
  - **Chave única.** Uma restrição de chave composta por apenas um atributo ou elemento;
  - **Referência.** Uma referência entre dois atributos ou elementos que devem possuir o mesmo valor;
  - **Conjunto valorado de referências.** Um conjunto de restrições de referência;
  - **Chave estrangeira.** Uma composição de dois tipos de restrição: Referência e Chave;
  - **Chave estrangeira unária.** Uma composição de uma única restrição de Referência e uma Chave única;
  - **Conjunto valorado de chave estrangeira.** Um conjunto restrições de referência relacionadas a restrições de Chave;
  - **Identificador Único.** Identificação única definida no contexto do documento inteiro;
  - **Relacionamento bidirecional (inversível).** Um conjunto de duas restrições de referência relacionadas mutuamente;
- **Restrições de caminho.** Uma restrição geral definida em termos dos caminhos de navegação (expressões de caminho no documento XML)
  - **Funcional.** Uma restrição que define o conceito de função a relacionamentos entre elementos e atributos;
  - **Referência.** Uma restrição similar a restrição de Referência presente nas Restrições básicas XML, porém definida através de expressões de caminho;
  - **Inversa.** Uma restrição que estabelece um relacionamento mútuo entre expressões de caminho.

Fonte: Adaptada de FAN & SIMÉON (2003).

Figura 4.1 - Categorias de RIS propostas em FAN & SIMÉON (2003).

O trabalho de DEUTSCH & TANNEN (2003), a fim de contribuir na pesquisa para otimização de consultas, analisa RI's com relação à decidibilidade do problema de determinação de equivalência. O foco são RI's que possuem expressões *XPath* em sua especificação, sendo estabelecida uma categoria de RI's denominada *Restrições de integridade XPath simples* como forma de delimitar o escopo do trabalho.

BUNEMAN et al. (2001) discutem a definição de chaves no contexto do modelo de dados XML. O trabalho considera duas categorias de RIS: *Restrições de Chave* – derivada do conceito de chave primária presente no modelo de dados relacional, com adaptações para a noção de igualdade de um nó XML que envolve não apenas seu valor,

mas sua localização - e *Restrições de chave relativa* – composta por uma chave e uma expressão representando o caminho do nó XML.

Observa-se, com base nos trabalhos relacionados, que ao contrário dos modelos de dados estruturados, diversas categorias de RIS no modelo de dados XML ainda possuem questões em aberto envolvendo a decidibilidade das mesmas. De fato categorias de RIS consolidadas, presentes no modelo relacional de dados, ganham nova discussão no modelo de dados XML.

Sobre linguagens de especificação de RIS no contexto XML, há na literatura diversas propostas. Entretanto não há consenso quanto a um padrão. A XML Schema é a recomendação mais recente da W3C para a definição de esquemas de documentos XML e seria a adoção natural como linguagem padrão para especificação de RIS. Entretanto, seu poder de expressão é limitado (DODDS, 2001; JACINTO et al., 2002a; HU & TAO, 2004). Vários tipos de RIS não podem ser especificadas em um XML Schema, como por exemplo, restrições baseadas em regras condicionais sobre elementos ou atributos. Tais limitações deixam questões em aberto e abrem espaço para a comunidade de pesquisa desenvolver propostas no contexto de RIS para documentos XML. Dentre estes trabalhos, alguns propõem taxionomias para RIS XML, como forma de comprovar sua expressividade e como conseqüência suas contribuições.

JACINTO et al. (2002a) propõem uma linguagem para especificação de RIS chamada XCSL (*XML Constraint Specification Language*) e um mecanismo de validação para a mesma. Em JACINTO et al. (2002b) é apresentado um estudo comparando a XCSL com o trabalho de DODDS (2001) que propõe a linguagem para especificação de RIS denominada *SCHEMATRON*. O trabalho afirma que ambas as linguagens são equivalentes em expressividade, porém, a XCSL se apresenta mais simples quanto à sua especificação. Mesmo assim, ambas as linguagens são mais expressivas que a XML Schema. O trabalho de JACINTO et al. (2002a) também apresenta, como forma de delimitar o escopo de sua proposta de linguagem de especificação de RIS, uma taxionomia demonstrada na Figura 4.2. O trabalho de JACINTO et al. (2002a) organiza as RIS apenas quanto ao tipo de limitação que estas são capazes de impor, desconsiderando aspectos como: *(i)* alcance – quantidade de nodos do documento XML necessários para verificar a restrição; e *(ii)* forma – tipo de

expressão utilizada para especificar a restrição. Na verdade, todas as categorias de restrições não classificadas em categorias presentes na taxionomia proposta são agrupadas em uma categoria denominada "*outras restrições*". Desta forma, como referência para auxiliar na avaliação de expresividade de linguagens de especificação de RIS, esta taxionomia se mostra pouco adequada, uma vez que considera poucos aspectos de uma RI. De igual forma, se demonstra pouco adequada para avaliar sistemas de gerenciamento de restrições em SGBD's XML quanto às RIS's que é capaz de dar suporte. Além disso, a proposta não apresenta uma fundamentação teórica para a definição da taxionomia.

- **Garantia de intervalo de domínio**. Limitam o valor de um elemento ou atributo a um intervalo de valores;
- **Dependências entre dois nós do documento**. Consideram relacionamentos entre elementos ou atributos, estando estes presentes na mesma sub-árvore ou em sub-árvores diferentes do documento XML;
- **Padrões em expressões regulares.** Atuam sobre um conjunto de palavras adequando-as a um padrão específico;
- **Restrições complexas.** Abrangem outras restrições, divididas em três categorias:
  - **Restrições de conteúdo misto.** Aplicam-se a elementos de conteúdo misto;
  - **Restrição de singularidade.** Tratam a questão da unicidade;
  - **Outras restrições.** Abrangem outras restrições não pertencentes às categorias acima.

Fonte: Adaptada de JACINTO et al. (2002a).

Figura 4.2 – Taxionomia para RIS proposta em JACINTO et al. (2002a).

Similar ao trabalho de JACINTO et al. (2002a), o trabalho de HU & TAO (2004) apresenta uma linguagem de especificação de RIS XML e uma taxionomia como forma de delimitar o escopo de sua linguagem. A linguagem proposta é chamada *XCML* (*eXtensible Constraint Markup Language*) e a taxionomia proposta é apresentada na Figura 4.3. Quanto a sua expressividade, a XCML se demonstra mais expressiva que a XCSL proposta por JACINTO et al. (2002a) ao dar suporte a *Restrições baseadas em expressões condicionais compostas* quando analisada sobre a faceta *Quanto à forma da restrição* (tomada como base a taxionomia proposta no capítulo 2). O trabalho de HU & TAO (2004) apresenta contribuições em relação ao trabalho de JACINTO et al. (2002a), tal como permitir classificar as RIS's quanto ao tipo de limitação que a RIS é capaz de impor e também quanto à forma como a RIS é especificada. Entretanto, ela apresenta algumas limitações. Devido, ao alto nível de abstração de suas categorias, englobando

diversos tipos de RIS, sua taxionomia dificulta a comparação de linguagens de especificação de RIS quanto ao seu poder de expressão. Além disso, ele não distingue RIS quanto ao seu alcance e não deixa clara a fundamentação teórica adotada como base para as categorias propostas.

- **Em relação ao tipo de limitação**:
  - **Restrições dinâmicas**. Apresentam comportamento condicional, ou seja, podem impor diferentes limitações a instâncias de elementos ou atributos de um documento XML;
  - **Restrições estáticas**. Impõem a mesma limitação a instâncias de elementos ou atributos de um documento XML.
- **Em relação à forma**:
  - **Restrições baseadas em expressões.** Possuem a forma de uma expressão booleana, retornando verdadeiro ou falso;
    - **Simples**. Envolvem apenas um atributo ou elemento;
    - **Composta**. Envolvem mais de um atributo ou elemento;
  - **Restrições baseadas em regras**. Definem condicionais com expressões embutidas;
    - **Simples**. Apresenta apenas uma estrutura condicional;
    - **Composta**. Apresenta aninhamentos condicionais.

Fonte: Adaptada de HU & TAO (2004).

Figura 4.3 – Taxionomia para RIS proposta em HU & TAO (2004).

Em LAZZARETTI & MELLO (2005) é proposta uma linguagem para especificação de RIS XML e um mecanismo de validação. De forma similar aos trabalhos anteriores, é proposta também uma taxionomia como forma de delimitar o escopo e comprovar a expressividade da linguagem proposta. A linguagem proposta é chamada *XDCL* (*XML Domain Constraint Language*) e a taxionomia correspondente é demonstrada na Figura 4.4. Quanto à expressividade, a XDCL se mostra mais expressiva quando analisada sobre a faceta *Quanto ao tipo de limitação imposta*, à medida que considera *Restrições de transição de estados*. A principal contribuição do trabalho de LAZZARETTI & MELLO (2005) é adotar uma fundamentação teórica baseada no modelo relacional de dados. Entretanto, em relação à abrangência de sua taxionomia, semelhante à proposta de HU & TAO (2004), apresenta um alto nível de abstração, dificultando também a comparação de linguagens de especificação de RIS quanto ao seu poder de expressão. Além disso, ela não distingue RIS quanto ao seu alcance, e distingue RIS quanto à sua forma. Desta forma, análogo aos trabalhos de

JACINTO et al. (2002a) e HU & TAO (2004), sua taxionomia proposta é pouco adequada para dar suporte à comparação entre linguagens para especificação de RIS.

- **Tipo.** Considera restrições estruturais, como seqüências de elementos, cardinalidades e tipos de dados;
- **Atributo.** Consideram valores permitidos para um elemento ou atributo;
- **Tupla.** Consideram a validação do conteúdo de elementos ou atributos que pertençam ao mesmo nível hierárquico em um documento XML;
- **Banco de Dados.** Consideram a validação de elementos ou atributos que pertençam a diferentes níveis hierárquicos em um documento XML;
- **Transição de estados.** Consideram transições de valores para elementos ou atributos.

Fonte: Adaptada de LAZZARETTI & MELLO (2005).

Figura 4.4 - Taxionomia proposta em LAZZARETTI & MELLO (2005).

A partir dos trabalhos relacionados comentados, observa-se que grande parte dos trabalhos presentes na literatura com foco em linguagens de especificação de RIS, adotam, como forma de mensurar a abrangência e expressividade de sua abordagem, a proposta de uma taxionomia para RIS. No entanto, a falta de uma taxionomia consolidada de RIS para dados XML torna a avaliação de abordagens subjetiva e pouco confiável.

De fato, observa-se que poucos trabalhos na literatura têm por objetivo principal propor uma taxionomia dita "completa" para RIS XML. O trabalho de PAVLOVA et al. (2000) propõe a análise da definição de RI's com base em modelos de dados estruturados, mais especificamente os modelos relacional, orientado a objetos e temporal. Ela apresenta ainda aplicações de RI no contexto de dados semi-estruturados. O universo das RI's XML é dividido em dois grupos de restrições: *(i) Restrições de integridade* – consistência semântica dos dados; e *(ii) Restrições de validade dos dados* – uma categoria de RI que impõe condições sobre a validade dos dados. O item *(i)* é especializado em *Restrições de tipo*, *Restrições de caminho* e *Restrições complexas*, onde este último representa um conjunto formado pela união dos dois primeiros. Com relação ao item *(ii),* a principal categoria é chamada de *Restrições de validade temporal*, derivada diretamente de conceitos do modelo de dados temporal. A principal contribuição do trabalho de PAVLOVA et al. (2000) é propor a adoção do modelo estruturado de dados como base para sua proposta de taxionomia, embora seja evidente que seu foco é o modelo de dados temporal. No mais, como taxionomia para auxiliar na

avaliação de expresividade de linguagens de especificação de RIS, este trabalho se mostra pouco adequado, uma vez que considera poucos aspectos de uma RIS além dos aspectos temporais. De igual forma, ela se demonstra pouco adequada para avaliar sistemas de gerenciamento de restrições em SGBD's XML quanto às RIS's que é capaz de dar suporte.

Através do estudo realizado, conclui-se que dentre as propostas para especificação RIS no contexto XML ainda não há uma proposta consolidada. De igual forma, não há uma taxionomia padrão que possa ser empregada como forma de mensurar a expressividade de mecanismos ou ferramentas que lidem com RIS. Estes pontos realçam a escolha adotada na abordagem DIInCX por uma linguagem de alto nível que possa ser traduzida para linguagens específicas posteriormente.

4.2 Abordagens de descoberta de informação

Nesta seção, são discutidas abordagens para descoberta de informação, principalmente aquelas que de alguma forma descobrem informação relevante para RIS's, ou ainda, que gerem informação que possa ser traduzida na forma de uma RIS. A descoberta de informação sobre dados semi-estruturados não é trivial e têm sido amplamente estudada (FENG & DILLON, 2004; ALIMOHAMMADZADEH et al., 2006). Tal atividade constitui uma importante etapa em SII's, refletindo diretamente na qualidade do resultado obtido.

Diversos trabalhos relacionados se utilizam das mais diversas abordagens (sendo estas não exclusivas) para descoberta de informação, dentre elas destacam-se: *(i*) abordagens que se utilizam de técnicas de mineração de dados; *(ii)* abordagens que realizam a transformação da fonte de dados semi-estruturada para esquemas de dados com soluções consolidadas, e ainda; *(iii)* aquelas que se baseiam no uso de *ontologias* como tecnologia de apoio. Ontologias, assim como *thesaurus*, possuem um importante papel no suporte a representação de conhecimento e apoio a diversos processos. Diversas técnicas das mais diversas áreas, como Lingüística Computacional ou Inteligência Artificial, podem ser empregadas na descoberta de informação. No entanto, devido à complexidade de sua implementação, elas estão fora do escopo deste trabalho.

Em se tratando de fontes de dados, diversos trabalhos abordam a descoberta de

informação tendo como fonte de dados o formato HTML (LAENDER et al., 2002, SNOUSSI et al., 2002, ALANI et al., 2003). Entretanto, conforme já discutido, o formato HTML apresenta desvantagens quando comparado à XML em relação à sua expressividade semântica.

Importantes abordagens se encontram no campo da mineração de dados aplicada ao modelo de dados XML (NAYAK et al., 2002; BRAGA et al. 2002; FENG & DILLON, 2004; GARBONI et al., 2006). Segundo NAYAK et al. (2002), a aplicação de técnicas de mineração de dados sobre dados XML (*XML mining*), consiste em abordar sua estrutura (esquemas XML) e conteúdo (valores de atributos e elementos de instâncias XML). Definição semelhante é dada por SNOUSSI et al. (2002) para a extração de informação sobre dados semi-estruturados. No mais, o trabalho de NAYAK et al. (2002) discute técnicas e conceitos de mineração de dados aplicadas sobre o modelo de dados XML. O trabalho de GAROFALAKIS et al. (1999) discute conceitos e técnicas de mineração de dados, tal como a mineração de RA's no contexto estruturado e semi-estruturado.

A abordagem proposta por DEUTSCH et al. (1999) trata da aplicação de um algoritmo de mineração de padrões de árvores freqüentes sobre os dados semi-estruturados. Na seqüência da aplicação do algoritmo, é adotada uma linguagem declarativa de consulta chamada *STORED*, a fim de mapear os dados semi-estruturados para um modelo relacional de dados. Com isto, busca tirar proveito de conceitos e ferramentas para modelos estruturados de dados.

GARBONI et al. (2006), baseando-se em *XML mining*, apresenta uma técnica de classificação para documentos XML, centrada na estrutura destes. Ela consiste em transformar o documento XML em uma sequência e então aplicar técnicas de reconhecimento de padrões. Os algoritmos propostos pelo autor a serem adotados são derivados do algoritmo *Apriori* (AGRAWAL et al., 1993).

Em BRAGA et al. (2002), é apresentada uma ferramenta para extração de RA's presentes em documentos XML denominada *XMINE*, que foi baseada em expressões XPath e na sintaxe da *XQuery* (linguagem de consulta XML). A contribuição relevante da abordagem de BRAGA et al. (2002) é considerar tanto conteúdo quanto a estrutura dos documentos XML. Em WAN & DOBBIE (2004) é utilizada uma implementação do

algoritmo Apriori em XQuery para minerar RA's. Os autores argumentam que desta forma não são necessárias operações de pré-processamento e pós-processamento. Entretanto, a abordagem proposta parte da premissa de que os dados se encontram em um formato bem comportado, difícil de ocorrer no contexto XML.

O trabalho de FENG & DILLON (2004) aborda a mineração de RA's através de *templates*. Um template representa uma premissa ou uma informação que o usuário especialista busca descobrir. ALIMOHAMMADZADEH et al. (2006) (baseado em FENG & DILLON (2004)) apresenta um modelo prático para mineração de RA's em documentos XML. Embora diminua a complexidade da mineração de dados, um template torna o processo de descoberta dependente do conhecimento do usuário especialista sobre o domínio.

Outra linha de pesquisa se baseia na transformação do dado semi-estruturado para um formato estruturado, dito mais "comportado", a fim de aproveitar soluções existentes para modelos de dados estruturados (SMITH & LOPEZ, 1997; EMBLEY et al., 1999). Em SMITH & LOPEZ (1997), é proposto um processo de descoberta de informação onde a etapa de reconhecimento de conceitos é baseada em expressões regulares e conceitos pré-definidos. A fim de indicar a similaridade com os conceitos descobertos, é proposto associar pesos aos conceitos pré-definidos.

Uma terceira linha de pesquisa adota o uso de ontologias como ferramentas de apoio na descoberta de informações. Em LAENDER et al. (2002) são apresentadas ferramentas baseadas em Ontologias no contexto semi-estruturado. Nesta área, destacam-se também os trabalhos de EMBLEY et al. (1999), SNOUSSI et al. (2002), ALANI et. al. (2003) e SVÁTEK et al. (2006). Entretanto, uma limitação observada é que a utilização de ontologias como ferramentas de apoio torna a qualidade do processo como um todo (nível de automatização deste e qualidade da informação obtida) proporcional à qualidade da ontologia (LAENDER et al., 2002). No entanto, diversas propostas já tratam questões como a extração automática de ontologias de diversas fontes de dados, minimizando desta forma, a necessidade de intervenção de um usuário especialista na sua construção (BENSLIMANE et al., 2006).

Todas as linhas de pesquisa estudadas apresentam vantagens em nível de aplicação. *XML mining* aborda tanto o conteúdo quanto a estrutura de documentos

XML. Ferramentas baseadas em ontologias se mostram capazes de diminuir a complexidade de abordagens de descoberta de informação e melhorar a qualidade do resultado obtido. Técnicas de transformação de esquemas XML em esquemas estruturados tiram proveito de pesquisas consolidadas. Entretanto, apesar da variedade de trabalhos relacionados, poucos discutem a descoberta de RIS ou as abordam de maneira superficial, com foco em restrições de integridade sintáticas de caráter simples, como tipos primitivos de dados. A maioria destes foca no problema de extração de esquemas (NESTOROV et al., 1998; HACID et al., 2000; CASTANO et al., 2002; CHIDLOVSKII, 2002; HEGEWALD et al., 2006).

Em CHIDLOVSKII (2002) é proposto um algoritmo para extração de esquemas XML inspirado em métodos de inferência gramatical. O trabalho de HEGEWALD et al. (2006) propõe uma abordagem chamada *XStruct*, a qual extrai esquemas XML a partir de instâncias XML, aplicando heurísticas a fim de deduzir expressões regulares. No entanto, estas abordagens possuem como principal fraqueza o fato de representarem o conhecimento descoberto (esquemas) através de linguagens específicas para definição de RIS's, as quais possuem limitações. O trabalho de LEE & CHU (2000) apresenta algumas linguagens de definição de RIS's destacando suas limitações e traçando comparativos.

O trabalho de LIU et al. (2004), faz uso de técnicas do modelo relacional de dados, propondo mapear as RIS descobertas para um esquema relacional. O trabalho de HACID et al. (2000) apresenta como relevante contribuição propor uma representação mais abrangente que as propostas anteriormente comentadas, representando as RIS descobertas em axiomas em lógica descritiva.

NESTOROV et al. (1998) abordam a descoberta de esquemas sobre dados semi-estruturados adotando a técnica de mineração de dados denominada *Agrupamento* (*clustering*). Em CASTANO et al. (2002) é proposto o uso de ontologias a fim de reduzir a complexidade do processo de descoberta. Entretanto, conforme comentado anteriormente, esta abordagem torna o processo diretamente dependente da qualidade da ontologia empregada.

No mais, conforme discutido através de trabalhos relacionados, as principais abordagens presentes na literatura no contexto da descoberta de RIS focam na

descoberta de esquemas. Elas lidam com poucas categorias de RI's, ou ainda abordam RIS de caráter simples, tais como a descoberta de tipos primitivos de dados, deixando de lado RIS de maior complexidade.

4.3 Sistemas de informação que se utilizam de RIS

Conforme comentado previamente, a incorporação de RIS's pode trazer diversas contribuições a sistemas de informação nas mais diversas áreas de pesquisa. No contexto do modelo de dados XML, destacam-se SII's XML (tanto de dados quanto de esquemas) (REYNAUD et al., 2001; ERDMANN & STUDER, 2001; CALI et al., 2002; DELOBEL et al., 2003; CRUZ et al., 2003; LETHI & FANKHAUSE, 2004; MELLO & HEUSER, 2005) e SGBD's XML (SCHÖNING, 2001; MEIER, 2002; WIWATWATTANA et al., 2003). Dentre as contribuições destacam-se: *(i)* uma maior consistência do modelo de dados XML; *(ii)* sistemas de consultas mais robustos; e *(iii)* maior precisão na integração quando os dados estiverem acompanhados de RIS. Por exemplo, em sistemas baseados em mediadores para fonte de dados XML heterogêneas, a análise de RIS pode evitar o acesso a fontes com dados irrelevantes para a consulta. Entretanto, poucos SII's XML e SGBD's XML fazem uso de RIS em seus processos, ou as abordam fracamente.

No contexto de SII's, destacam-se duas principais abordagens: *(i)* conversão de esquemas de dados XML para um esquema global através de um processo de integração, a fim de que este, por sua vez, seja capaz de abstrair a alta heterogeneidade estrutural de cada esquema XML (CRUZ et al., 2003; LETHI & FANKHAUSE, 2004; MELLO & HEUSER, 2005), e *(ii)* tratar especificamente diferenças estruturais e semânticas inerentes a cada fonte (REYNAUD et al., 2001), não se utilizando de um esquema global. Segundo REYNAUD et al. (2001), ambas as abordagens possuem vantagens e desvantagens. No entanto, um estudo comparativo destas se encontra fora do escopo deste trabalho.

A respeito das propostas presentes na literatura, além da abordagem adotada (utilização ou não um esquema global), é possível distinguir outras importantes características, tais como: a utilização ou não de ferramentas de apoio (como ontologias

ou *thesaurus*), a forma de definição do esquema global ou ainda o tipo de tratamento dado às RIS.

REYNAUD et al. (2001) propõem a geração automática de mapeamentos entre conceitos presentes em fontes heterogêneas XML, não utilizando desta forma, um esquema global. Além de aspectos sintáticos, também são considerados aspectos semânticos com o auxílio de um *thesaurus* como ferramenta da apoio. Entretanto, seu foco é restrito a conceitos e seus relacionamentos, considerando desta forma apenas RIS diretamente relacionadas a este último, tais como: especialização entre conceitos ou relações de composição. A proposta se insere no contexto do projeto XYLEME (DELOBEL et al., 2003), cujo objetivo é permitir consultas a um grande repositório de documentos XML.

Em MELLO & HEUSER (2005) é proposto o processo de integração de esquemas XML chamado *BinXS* (*Bottom-up Integration of XML Schemata*). O resultado deste processo de integração é um esquema conceitual global especificado em OWL (OWL, 2007) e um conjunto de mapeamentos definidos na linguagem *XPath*. Os mapeamentos propostos relacionam os conceitos do esquema global aos elementos e atributos das fontes de dados XML heterogêneas. Já em LETHI & FANKHAUSE (2004), utiliza-se uma ontologia como esquema global. De forma semelhante ao trabalho de MELLO & HEUSER (2005), também é gerado como resultado um conjunto de mapeamentos de igual finalidade. Tanto estes quanto o esquema global são especificados em OWL. Ainda segundo os autores, a adoção de ontologias possui diversas vantagens, dentre as quais a base formal existente e a presença de mecanismos nativos para definição de mapeamentos semânticos. Outros trabalhos, como ERDMANN & STUDER (2001) e CRUZ et al. (2003), utilizam abordagens semelhantes ao empregar uma ontologia como esquema global. Em CRUZ et al. (2003), a ontologia global é especificada em RDF, sendo o resultado da integração de ontologias geradas individualmente sobre cada fonte de dados XML. Em NOY (2004), é apresentado um *survey* sobre abordagens de SII's que utilizam ontologias.

No que diz respeito às RIS, de forma semelhante ao trabalho de REYNAUD et al. (2001), os trabalhos anteriormente comentados ignoram um grande conjunto de RIS, focando basicamente em relacionamentos entre conceitos. Por fim, CALI et al. (2002)

apresenta um diferencial ao considerar a incorporação de RIS envolvendo unicidade e unicidade com referência entre elementos. No entanto, se restringe a apenas este tipo de RIS.

Quanto a SGBD's XML, conforme demonstrado na Tabela 2.1, poucos consideram RIS ou consideram fracamente (SCHÖNING, 2001; MEIER, 2002). As abordagens que consideram RIS apresentam categorias de tipos simples de RIS, não permitindo RIS envolvendo expressões condicionais ou transição de estados, categorias já a muito consideradas em modelos de dados estruturados. Estas poucas categorias de RIS são essenciais para definição de regras de negócio, possibilitando a construção de modelos de dados consistentes.

A partir do estudo levantado, conclui-se que tanto em SII's XML quanto em SGBD's XML, a adoção de RIS como forma de otimizar processos se encontra num estágio inicial, sendo desconsiderada ou abordada fracamente por estas. Esta constatação realça as contribuições da abordagen DIInCX como forma de incentivar a adoção de RIS nestas abordagens.

5 A ABORDAGEM DIINCX

Este capítulo detalha a abordagem *DIInCX* (*Approach to Discovery of Implicit Integrity Constraints from XML Data*) proposta neste trabalho. Os resultados deste capítulo geraram o artigo entitulado *DIInCX: An Approach to Discovery of Implicit Integrity Constraints from XML Data*, publicado na conferência *IEEE International Conference on Information Reuse and Integration* (IRI) (RODRIGUES & MELLO, 2007a). A abordagem proposta define um processo para descoberta semi-automática de RIS's a partir de instâncias XML. O processo localiza nas instâncias XML, através de uma técnica de mineração de dados, aspectos semânticos (co-ocorrências, padrões de relacionamentos estruturais e de conteúdo) que possam ser traduzidos para RIS's. Contudo, o processo é dito semi-automático, pois ao final do mesmo, o usuário especialista é chamado a avaliar as RIS's descobertas quanto à sua relevância e/ou definir novas RIS's pertinentes ao domínio. O usuário especialista também é responsável por fornecer um conjunto de parâmetros de entrada apresentados na Tabela 5.1, que ajudam a abordagem a convergir no resultado adequado para o dado domínio. Uma descrição detalhada de cada parâmetro é dada nas próximas seções.

O processo de descoberta de RIS's adota como dado de entrada um conjunto de instâncias XML ao invés de linguagens de definição de esquemas, tais como a XML Schema (XML SCHEMA, 2007), pois definições de RIS's a nível de esquemas são usualmente incompletas, como comentado anteriormente. Uma especificação de RIS completa a respeito de um dado domínio é difícil, seja pela grande quantidade de RIS's a serem especificadas, seja pela falta de conhecimento de um usuário humano a respeito do domínio, ou ainda, por limitações de linguagens para definição de esquemas. As instâncias XML tomadas como dados de entrada no processo devem pertencer a um mesmo domínio de dados, possuindo uma estrutura XML comum. A integração de esquemas e instâncias foge do escopo deste trabalho, principalmente devido ao fato de já haver disponível na literatura diversas abordagens para integração de informação sobre dados semi-estruturados tais como REYNAUD et al. (2001), ERDMANN & STUDER (2001), CALI et al. (2002), DELOBEL (2003), CRUZ et al. (2003), LETHI & FANKHAUSE (2004) e MELLO & HEUSER (2005).

O resultado da abordagem DIInCX é uma estrutura hierárquica representativa da estrutura das instâncias XML e um conjunto de RIS's, que podem ser incorporadas a qualquer sistema de informação que pretenda dar suporte a RIS's no contexto do domínio composto pelas instâncias XML de entrada.

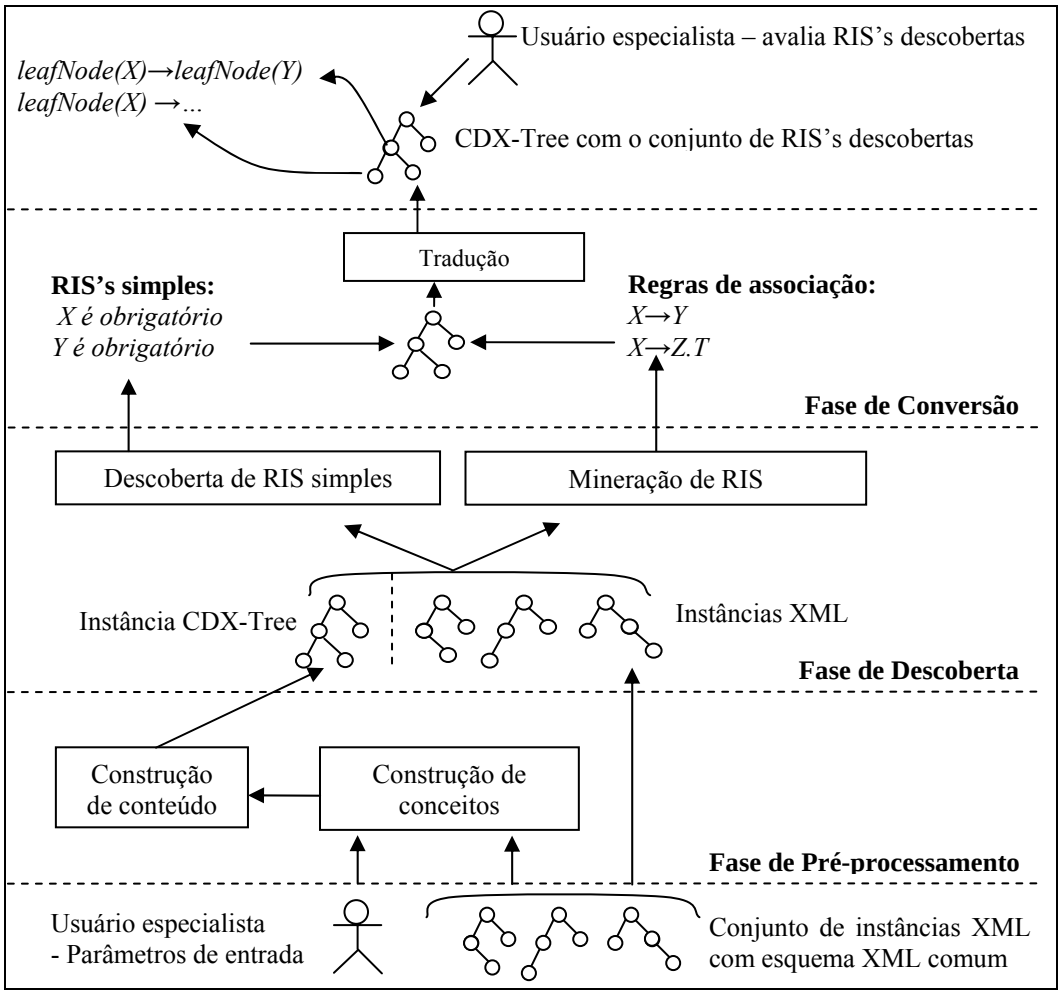
Tabela 5.1 – Parâmetros de entrada para a abordagem DIInCX.

Parâmetro	**Descrição**
Relevância mínima	Quantidade mínima de vezes que uma RA deve ocorrer no conjunto de transações que compõe o domínio de dados, a fim de ser considerada. Equivalente ao conceito "*suporte*" em mineração de RA's.
Percentual de discretização	Percentual a ser discretizado para um elemento quantificado.
Percentual de dispersão máximo	Percentual acima do qual um conceito deve ser considerado disperso.
Complexidade máxima do componente restringido	Complexidade máxima desejada para o componente restringido das RIS descobertas.
Complexidade máxima do componente condição restritiva	Complexidade máxima desejada para a componente condição restritiva das RIS descobertas.
Quantidade máxima de RIS descobertas	A quantidade máxima de RIS`s que se deseja descobrir.
Quantidade máxima de itens para enumeração	A quantidade máxima de valores cabíveis, que um conceito qualificado deve possuir para que seja gerada uma RIS que restrinja o domínio deste a um conjunto de valores.
Quantidade mínima de conceitos para consideração de subárvores	A quantidade mínima de conceitos que uma subárvore deve possuir para ser considerada pela abordagem na etapa de Mineração de RIS.

Fonte: Primária.

A abordagem é composta por três fases, conforme demonstrado na Figura 5.1. A primeira fase é chamada *Pré-processamento* e tem como foco uniformizar e simplificar a representação das instâncias XML através do mapeamento da estrutura em uma instância de um esquema conceitual hierárquico, definido em OWL, que serve de guia para as próximas etapas da abordagem. Este esquema conceitual hierárquico é chamado *CDX-Tree* (*Complete Domain XML Tree*) e tem por objetivo principal representar, através de uma única estrutura hierárquica, a estrutura completa e simplificada da

árvore de elementos existentes nas instâncias XML. Ele é definido na seção 5.1.1 e representa umas das contribuições deste trabalho. A fase de Pré-processamento é detalhada na seção 5.1.



Fonte: Primária

Figura 5.1 - Visão geral da abordagem DIInCX.

A segunda fase é chamada de fase de *Descoberta*. Esta fase consiste na aplicação de um algoritmo para mineração de regras de associação (RA) sobre as instâncias XML, tendo a CDX-Tree como guia, ou "*índice*", para acesso aos dados pertencentes às instâncias. O objetivo desta fase é descobrir padrões de relacionamentos estruturais e de conteúdo que possam ser traduzidos para RIS's válidas e relevantes. Esta fase ainda descobre algumas RIS's de menor complexidade, tais como itens restringidos a enumerações, através da análise da instância CDX-Tree. A seção 5.2 detalha esta fase.

A terceira fase é chamada de fase de *Conversão*. Nesta fase, as RA's descobertas são traduzidas efetivamente para RIS's especificadas em SWRL e incorporadas à CDX-Tree. Após a sua execução, o usuário especialista deve avaliar o conjunto de RIS's descobertas, e descartar as que julgar de baixa relevância, ou ainda criar novas RIS's caso julgue necessário. A fase de Conversão é detalhada na seção 5.3.

A partir da fase de Conversão a instância da estrutura CDX-Tree se encontra completa, representando todos os elementos presentes nas instâncias XML e suas RIS's descobertas, podendo ser utilizada por qualquer sistema de informação que pretenda dar suporte às RIS's. A seção 5.4 discute, segundo a taxionomia para RIS's XML proposta neste trabalho, as categorias capazes de serem identificadas pela abordagem proposta como forma de mensurar sua abrangência.

5.1 Fase de Pré-processamento

A fase de *Pré-processamento* tem como principal objetivo uniformizar e simplificar a representação das instâncias XML, preparando-as para a etapa de descoberta de RIS's propriamente dita. A uniformização e simplificação da estrutura das instâncias XML se dá com a sua representação através de uma instância do esquema conceitual hierárquico *CDX-Tree.*

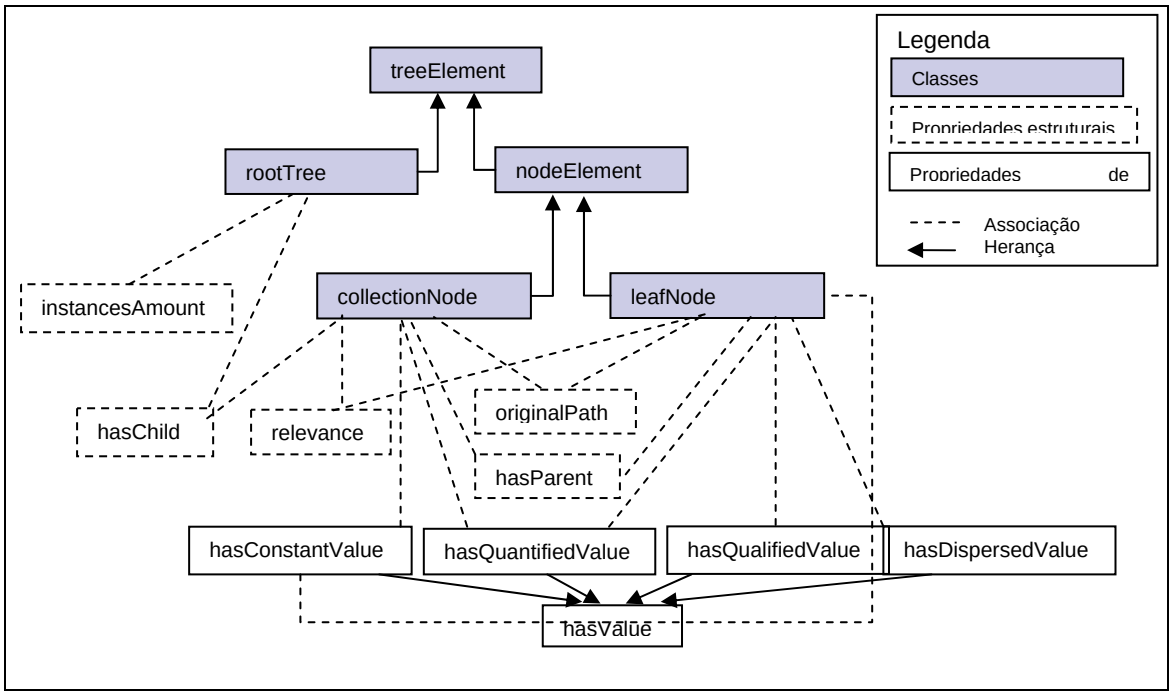
A construção da instância da estrutura CDX-Tree se dá através da execução de um conjunto de regras, que foram distinguidas, conforme sua função, em duas etapas: *Construção de conceitos* e *Construção de conteúdo*. Todas as regras são aplicadas através da leitura dos elementos e atributos presentes nas instâncias XML e posterior representação na instância da estrutura CDX-Tree. Instâncias das classes CDX-Tree são definidas na etapa de *Construção de conceitos* e a partir desta etapa denominadas como *conceitos* do esquema conceitual CDX-Tree. Na etapa de *Construção de conteúdo* os conceitos são qualificados em relação ao seu domínio de dados. As classes que compõem o esquema conceitual hierárquico CDX-Tree, são apresentadas na seção 5.1.1 As etapas de Construção de conceitos e Construção de conteúdo, na qual é criada a instância da estrutura CDX-Tree, são detalhadas na seção 5.1.2.

5.1.1 CDX-Tree - Complete Domain XML Tree

A CDX-Tree é definida neste trabalho como um esquema conceitual hierárquico especificado em OWL. Os metadados deste esquema são definidos por um conjunto de classes e propriedades especificadas em OWL. Desta forma, a instância de uma estrutura CDX-Tree pode ser comparada a um "*indivíduo*" de uma ontologia especificada em OWL. De forma complementar, as RIS's descobertas através da abordagem proposta são definidas em SWRL e incorporadas a este esquema conceitual hierárquico.

O objetivo principal da CDX-Tree é representar, através de uma única estrutura hierárquica, a hierarquia simplificada resultante da união das estruturas das instâncias XML. Além disso, a instância CDX-Tree também armazena informações qualitativas e quantitativas sobre os valores de elementos e atributos presentes nas instâncias XML. Desta forma, a instância CDX-Tree atua como um guia, ou "*índice*", provendo acesso direto aos dados das instâncias XML que sejam considerados relevantes para a descoberta de RIS's.

No processo de construção da instância da CDX-Tree, elementos e atributos presentes nas instâncias XML de entrada são traduzidos para instâncias das classes da CDX-Tree. (o processo de construção da CDX-Tree é detalhado na próxima seção). Estas classes, em conjunto com um grupo de propriedades definidas como *propriedades estruturais*, definem a estrutura hierárquica do esquema conceitual CDX-Tree (Figura 5.2).



Fonte: Primária

Figura 5.2 - Classes e propriedades da CDX-Tree.

As propriedades definidas como estruturais, apresentadas na Figura 5.3, são: *hasChild, hasParent, originalPath, instancesAmount* e *relevance*. A propriedade *hasParent* indica o elemento imediatamente superior na estrutura hierárquica, enquanto a propriedade *hasChild* indica o elemento imediatamente abaixo na estrutura hierárquica. Em uma estrutura hierárquica em árvore, cada nó só pode possuir uma única propriedade *hasParent*, com exceção da classe *rootTree* que não possui esta propriedade. No entanto, não há limite para a quantidade de propriedades *hasChild*, com exceção da classe *leafNode* que não possui esta propriedade. A propriedade *instancesAmount* é atribuída apenas à classe *rootTree* e significa a quantidade de instâncias XML que a estrutura CDX-Tree representa. A propriedade *originalPath* representa, através de uma expressão XPath absoluta, o caminho original do elemento ou atributo nas instâncias XML. A propriedade *relevance* indica a quantidade de vezes que o elemento ou atributo se faz presente nas instâncias. A Figura 5.3, asssim como os demais exemplos apresentados nesta seção são mostrados em sintaxe abstrata (OWL, 2007).

(a) Propriedade hasChild	**(b) Propriedade hasParent**
ObjectProperty(b:hasChild inverseOf(b:hasParent) domain(b:treeElement) range(b:nodeElement))	ObjectProperty(b:hasParent inverseOf(b:hasChild) domain(b:nodeElement) range(b:treeElement))
(c) Propriedade originalPath	**(d) Propriedade instancesAmount**
DatatypeProperty(b:originalPath domain(b:nodeElement) range(xsd:string))	DatatypeProperty(b:instancesAmount domain(b:rootTree) range(xsd:int))
(e) Propriedade relevance	
DatatypeProperty(b:relevance domain(b:nodeElement) range(xsd:int))	

Fonte: Primária

Figura 5.3 - Propriedades estruturais da CDX-Tree.

A fim de definir o domínio de dados das classes que compõem a CDX-Tree, é definida também um conjunto de propriedades chamadas *propriedades de conteúdo* (Figura 5.2). Estas propriedades têm por função distinguir e agrupar os conceitos compostos a partir das *propriedades estruturais*, segundo o tipo dos dados que compõe o seu domínio. Com relação ao tipo de dado, este trabalho distingui os conceitos de forma não exclusiva em: *(i) qualificados*, quando o dado representa um valor não numérico; *(ii) quantificados*, quando o dado representa um valor numérico; *(iii) constantes*, quando um valor do tipo qualificado ou quantificado se mostra constante em todas as instâncias XML; e *(iv) dispersos*, quando o dado é do tipo qualificado e apresenta um percentual de elementos distintos presentes nas instâncias XML, superior ao *percentual de dispersão máximo* (Tabela 5.1). Um percentual de dispersão de 70% indica que o conceito é considerado disperso quando possuir, em mais de 70% das instâncias XML em que estiver presente, um valor distinto.

As *propriedades de conteúdo* são definidas a partir da propriedade *hasValue* e suas sub-propriedades (Figura 5.4), que têm por função representar os valores cabíveis para o conceito, segundo dados presentes nas instâncias XML. A propriedade *hasValue* não é associada a um conceito (*hasValue* é definida como uma propriedade abstrata), mas sim uma de suas sub-propriedades: *(i) hasQualifiedValue*, representando um conceito com domínio qualificado; *(ii) hasQuantifiedValue*, representando um conceito com domínio quantificado; (*iii*) *hasConstantValue*, representando um conceito com

domínio constante; e (*iv*) *hasDispersedValue*, representando um conceito com domínio disperso.

(a) Propriedade hasQualifiedValue	**(b) Propriedade hasQuantifiedValue**
DatatypeProperty(b:hasQualifiedValue domain(b:leafNode) range(xsd:string))	DatatypeProperty(b:hasQuantifiedValue domain(unionOf(b:collectionNode b:leafNode)) range(xsd:double))
(c) Propriedade hasConstantValue	**(d) Propriedade hasDispersedValue**
DatatypeProperty(b:hasConstantValue domain(unionOf(b:collectionNode b:leafNode)) range(xsd:string)))	DatatypeProperty(b:hasDispersedValue domain(b:leafNode) range(oneOf("true"^^xsd:boolean)))

Fonte: Primária

Figura 5.4 - Sub-propriedades da propriedade hasValue.

As propriedades *hasConstantValue* e *hasDispersedValue* são atribuídas uma única vez a todo conceito definido como constante ou disperso, respectivamente. A propriedade *hasQualifiedValue* pode ser atribuída inúmeras vezes e tem como finalidade representar uma lista de valores possíveis para o conceito, segundo valores presentes nas instâncias XML. A propriedade *hasQuantifiedValue*, de forma similar a *hasQualifiedValue*, não possui restrição quanto à quantidade de associações a um conceito. No entanto, por se tratar de valores numéricos, eles devem ser discretizados a fim de diminuir a quantidade de valores a serem considerados nas próximas etapas da abordagem. O método de discretização adotado é detalhado na seção 5.1.2.

A seguir são detalhadas as classes de metadados da CDX-Tree, com suas respectivas propriedades.

- **treeElement.** Classe raiz de todas as demais classes da estrutura hierárquica. Não é instanciada, sendo definida como uma classe abstrata sem propriedades. A Figura 5.5(b) apresenta a classe *treeElement*.
- **nodeElement.** Super classe das classes que representam os nós da estrutura hierárquica: *collectionNode* e *leafNode*. Não é instanciada, sendo definida como uma classe abstrata sem propriedades. A Figura 5.5(a) apresenta a classe *nodeElement* em sintaxe abstrata.

(a) Classes nodeElement	(b) Classe treeElement
Class(b:nodeElement partial b:treeElement)	Class(b:treeElement partial)

Fonte: Primária

Figura 5.5 - Classes nodeElement e treeElement.

- **rootTree.** Classe que tem por finalidade representar o elemento raiz da estrutura hierárquica. Ela possui como propriedades: *(i) hasChild*; e *(ii) instancesAmount*. A Figura 5.6 apresenta a classe *rootTree*, enquanto a Figura 5.7 exemplifica uma instância da mesma, através do elemento raiz denominado *professor*.

```
Class(b:rootTree partial
 restriction(b:hasChild minCardinality(1))
 restriction(b:instancesAmount cardinality(1))
 b:treeElement)
```

Fonte: Primária

Figura 5.6 - Classe rootTree.

```
Individual(b:professor
 type(b:rootTree)
 value(b:hasChild b:completeName.firstName)
 value(b:hasChild b:degree)
 value(b:hasChild b:age)
 value(b:hasChild b:email)
 value(b:hasChild b:researchLines)
 value(b:hasChild b:completeName.lastName)
 value(b:hasChild b:courses)
 value(b:instancesAmount "40"^^xsd:int))
```

Fonte: Primária

Figura 5.7 - Exemplo de instância da classe rootTree.

- **collectionNode.** Classe que representa uma coleção de nós da estrutura hierárquica. Ela possui como propriedades: *(i) originalPath*; *(ii) relevanc*e; *(iii) hasParent*; *(iv) hasValue*; e *(v) hasChild*. A Figura 5.8 apresenta a classe *collectionNode*, enquanto a Figura 5.9 exemplifica uma instância da mesma através do elemento raiz denominado *researchLines*.

```
Class(b:collectionNode partial
  restriction(b:hasValue minCardinality(1))
  b:nodeElement
  restriction(b:hasParent cardinality(1))
  restriction(b:relevance cardinality(1))
  restriction(b:originalPath cardinality(1))
  restriction(b:hasChild minCardinality(1)))
```

Fonte: Primária

Figura 5.8 - Classe collectionNode.

```
Individual(b:researchLines
  type(b:collectionNode)
  value(b:hasParent b:professor)
  value(b:hasChild b:researchLine.name)
  value(b:relevance "30"^^xsd:int)
  value(b:hasQuantifiedValue "0"^^xsd:double)
  value(b:hasQuantifiedValue "4"^^xsd:double)
  value(b:hasQuantifiedValue "2"^^xsd:double)
  value(b:originalPath "/professor/researchLines"^^xsd:string))
```

Fonte: Primária

Figura 5.9 - Exemplo de instância da classe collectionNode.

- **leafNode.** Classe que representa um nó folha da estrutura hierárquica. Ela possui como propriedades: *(i) originalPath*; *(ii) relevance*; *(iii) hasParent*; e *(iv) hasValu*e. A Figura 5.10 abaixo apresenta a classe *leafNode*, enquanto a Figura 5.11 apresenta um exemplo de instância da mesma através do elemento *degree*.

```
  Class(b:leafNode partial
  restriction(b:hasValue minCardinality(1))
  b:nodeElement
  restriction(b:relevance cardinality(1))
  restriction(b:hasParent cardinality(1))
  restriction(b:originalPath cardinality(1)))
```

Fonte: Primária

Figura 5.10 - Classe leafNode.

```
Individual(b:degree
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "40"^^xsd:int)
  value(b:hasQualifiedValue "master"^^xsd:string)
  value(b:hasQualifiedValue "graduated"^^xsd:string)
  value(b:hasQualifiedValue "doctor"^^xsd:string)
  value(b:originalPath "/professor@degree"^^xsd:string))
```

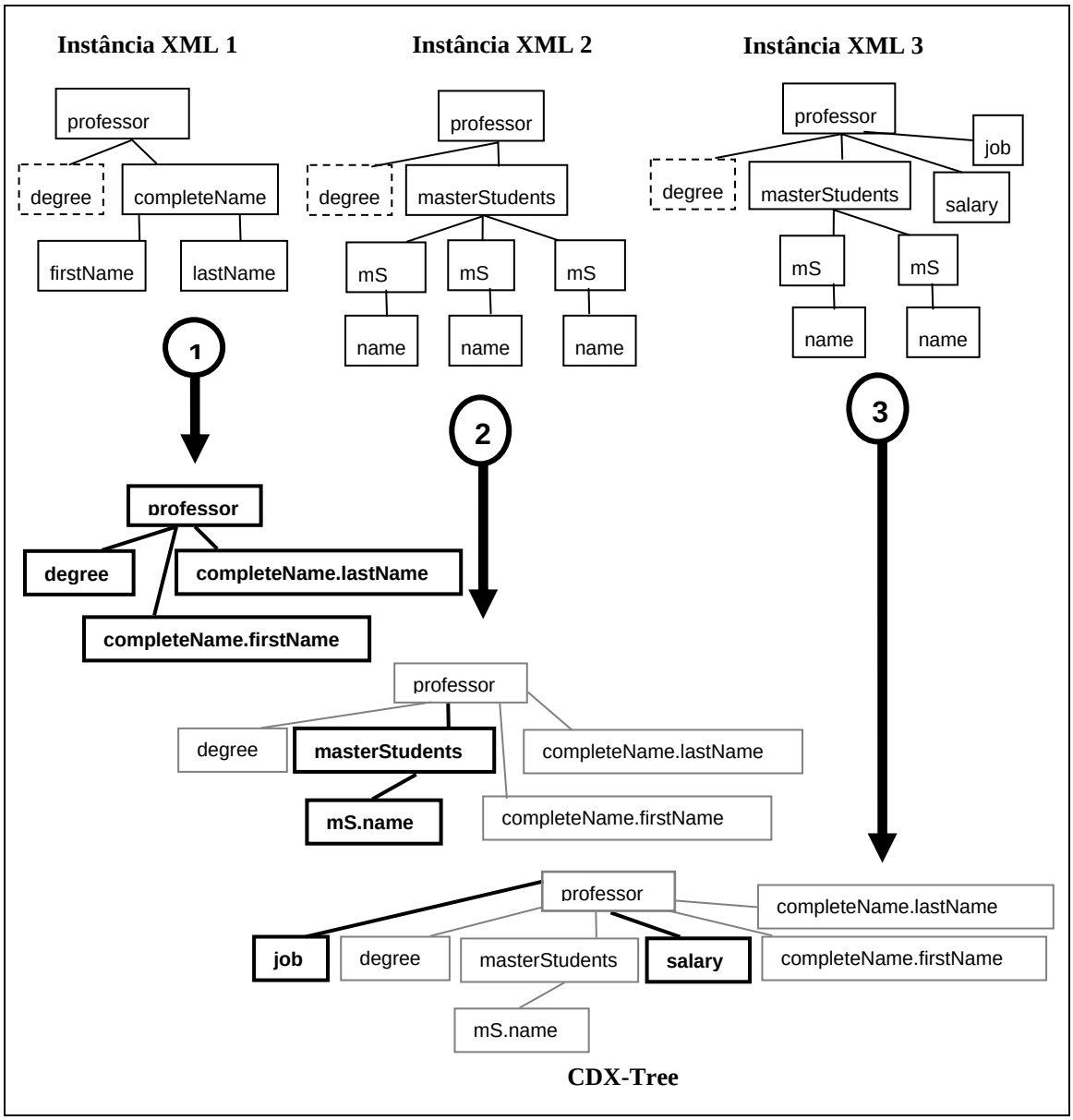
Fonte: Primária

Figura 5.11 - Exemplo de instância da classe leafNode.

A definição completa das classes do esquema conceitual hierárquico CDX-Tree é apresentada no Anexo 2.

5.1.2 Regras de Pré-processamento

Esta seção detalha as regras envolvidas na fase de Pré-processamento. O principal objetivo desta fase é uniformizar e simplificar a representação das instâncias XML, minimizando desta forma, a quantidade de dados a serem considerados na etapa de descoberta de RIS. A fase de pré-processamento se baseia em um conjunto de regras, distinguidas conforme sua função em duas etapas: *Construção de conceitos* e *Construção de conteúdo*. Ambas as etapas são vinculadas à construção da instância do esquema conceitual hierárquico CDX-Tree, descrito na seção anterior, que é o produto final desta fase.



Fonte: Primária

Figura 5.12 - Exemplo de construção da estrutura hierárquica da CDX-Tree.

A primeira etapa, chamada *Construção de conceitos,* foca na tradução dos atributos e elementos XML para instâncias das classes do esquema conceitual hierárquico CDX-Tree. Nesta etapa são atribuídas às instâncias das classes as propriedades *estruturais*, responsáveis por compor a estrutura hierárquica da CDX-Tree. Instâncias das classes CDX-Tree quando agregadas a suas propriedades estruturais são denominadas neste trabalho como *conceitos* do esquema conceitual CDX-Tree. A etapa de Construção inicia pela leitura das instâncias XML e incorpora à CDX-Tree a suas estruturas, a fim de criar uma instância de um esquema hierárquico que compreenda apenas a estrutura de elementos e atributos presentes nas instâncias

XML. A estrutura das instâncias XML é simplificada através da redução dos níveis hierárquicos da estrutura. A Figura 5.12 exemplifica a aplicação da etapa da fase de Pré-processamento, regras que compõem esta etapa, são as seguintes:

- **Regra 1 – Criação do elemento raiz**. O elemento raiz das instâncias XML, como o elemento *professor* na Figura 5.13(a), deve ser traduzido para uma instância do conceito *rootTree* da classe de metadados da CDX-Tree, conforme exemplificado na Figura 5.13(b);

(a) Exemplo de instância XML.

```
<professor degree =”doctor” >
    <age>30</age>
    <completeName>
      <firstName>Inacio</firstName>
      <lastName>Dorvantil</lastName>
    </completeName>
    <masterStudents>
      <masterStudent><name>Cristiane</name></...>
      <masterStudent><name>Marta G.</name></...>
      <masterStudent><name>Mathina</name></...>
    </masterStudents>
    <job>professor</job>
    <salary>5000,00</salary>
    <laboratory>
      <name>Database Laboratory</name>
    </laboratory>
...
 </professor>
```

(c) Exemplo de instância do conceito collectionNode.

```
Individual(b:masterStudents
  type(b:collectionNode)
  value(b:hasParent b:professor)
  value(b:hasChild b:masterStudent.name)
  value(b:relevance "30"^^xsd:int)
  value(b:hasQuantifiedValue "0"^^xsd:double)
  value(b:hasQuantifiedValue "2"^^xsd:double)
  value(b:hasQuantifiedValue "4"^^xsd:double)
  value(b:originalPath
"/professor/masterStudents"^^xsd:string))
```

(b) Trecho de uma instância da CDX-Tree.

```
...
Individual(b:professor
type(b:rootTree)
  value(b:hasChild b:degree))
Individual(b:degree
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "40"^^xsd:int)
  value(b:hasQualifiedValue "master"^^xsd:string)
  value(b:hasQualifiedValue "graduated"^^xsd:string)
  value(b:hasQualifiedValue "doctor"^^xsd:string)
  value(b:originalPath "/professor@degree"^^xsd:tring))
Individual(b:age
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "30"^^xsd:int)
  value(b:hasQuantifiedValue "30"^^xsd:double)
  value(b:hasQuantifiedValue "38"^^xsd:double)
  value(b:hasQuantifiedValue "46"^^xsd:double)
  value(b:originalPath "/professor/age"^^xsd:string))
Individual(b:completeName.firstName
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "40"^^xsd:int)
  value(b:hasDispersedValue "true"^^xsd:boolean)
  value(b:originalPath "/professor/completeName/firstName"^^xsd:string))
Individual(completeName.lastName
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "40"^^xsd:int)
  value(b:hasDispersedValue "true"^^xsd:boolean)
  value(b:originalPath "/professor/completeName/lastname"^^xsd:string))
Individual(job
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "40"^^xsd:int)
  value(b:hasConstantValue “professor”^^xsd:string)
  value(b:originalPath "/professor/job"^^xsd:string))
```

Fonte: Primária

Figura 5.13 - Instância XML (a), trecho da instância CDX-Tree correspondente (b) e instância collectionNode (c).

- **Regra 2 – Atributos para conceitos leafNode**. Atributos, como o atributo *degree* presente na Figura 5.13(a), devem ser traduzidos para uma instância da classe leafNode de mesmo nome, tal qual exemplificado na Figura 5.13(b);

- **Regra 3 – Elementos simples para conceitos leafNode**. Elementos simples, como o elemento *age* presente na Figura 5.13(a) devem ser traduzidos para instâncias da classe leafNode de mesmo nome, conforme exemplificado na

Figura 5.13(b). Elementos simples que estiverem contidos em elementos complexos serão tratados na regra 5 a seguir;

- **Regra 4 – Elementos que compõem coleções para conceitos collectionNode**. Elementos do tipo coleção são um caso especial de elemento XML que por si só não possuem informação semântica, atuando meramente como um contêiner para elementos de um mesmo tipo. O elemento *masterStudents,* presente na Figura 5.13(a), é um exemplo de elemento do tipo coleção, contêiner para os elementos *masterStudent*. Este tipo de elemento deve ser traduzido para uma instância da classe *collectionNode,* conforme apresentado na Figura 5.13(c);
- **Regra 5 – Elementos complexos para conceitos leafNode**. Ao traduzir elementos complexos, são gerados tantos conceitos *leafNode* quantos descendentes houverem. O nome do conceito traduzido deve ser composto pelo nome do elemento complexo, seguido pelos nomes dos elementos filhos, separados por *"."*. Os conceitos *completeName.firstName* e *completeName.lastName* demonstrados na Figura 5.13(b), são exemplos de elementos leafNode resultantes da tradução do elemento complexo XML *completeName* presente na Figura 5.13(a).

A etapa seguinte é chamada de etapa de *Construção de conteúdo.* Esta etapa foca na construção do domínio de dados dos conceitos criados no passo anterior. Isto se dá através da atribuição das sub-propriedades da classe de metadados da CDX-Tree *hasValue*. Nesta etapa, valores contínuos são transformados em valores discretos. A discretização se justifica, pois a mineração de RA's se torna uma tarefa onerosa quando aplicada sobre valores contínuos. Desta forma, objetiva-se reduzir a quantidade de valores numéricos a serem tratados através da aplicação de um método de discretização para simplificar o processo de mineração de RA's, aplicado na próxima fase.

O método de discretização empregado objetiva ser simples e baseia-se no *percentual de discretização* desejado*,* parâmetro este fornecido pelo usuário especialista para estabelecer intervalos o mais simetricamente distantes possível, dentre os valores existentes (*Equal Width Discretization*) (YANG, 2003). Vale comentar que, baseado em experimentos realizados no desenvolvimento da abordagem DIInCX, a escolha pelo método ideal é diretamente dependente do domínio de dados. No mais, questões como

desempenho do método, estão fora do escopo deste trabalho. Em DOUGHERTY et al. (1995) são discutidos diversos métodos de discretização.

O método adotado nos experimentos da abordagem DIInCX, elege um conjunto de valores dentre aqueles pertencentes ao domínio do elemento a ser discretizado, que estejam o mais simetricamente distantes possível. Por exemplo, partindo de um conjunto de valores *V*={20,30,40,50,60,70,80} pertencentes ao domínio de valores de um elemento denominado *age,* e tomando como percentual de discretização o valor de *50%*, têm-se como valores eleitos o conjunto *V'*={20,40,60,80}. Desta forma, a partir de um conjunto *V* de tamanho igual a sete e através da aplicação do método de discretização adotado, têm-se como resultado o conjunto *V'* de tamanho quatro.

Estes valores discretizados devem ser armazenados na estrutura CDX-Tree a fim de serem empregados na fase de Descoberta. Na fase de Descoberta, estes valores são definidos na forma de intervalos dois-a-dois, resultando no conjunto *V''*={*">=20To<=40"*, *">40To<=60"*, *">60To<=80"}*. Esses intervalos são adotados como dados de entrada, em substituição aos dados reais, no algoritmo de mineração de RA's.

Com relação a uma RIS e com base na observação de experimentos desenvolvidos na abordagem DIInCX, observa-se que o emprego de dados baseados em intervalos na descoberta de RIS's origina RIS com maior relevância do que quando baseadas diretamente sobre os valores discretizados.

As regras que compõem esta etapa são as seguintes:

- **Regra 6 – Criação do domínio de conceitos qualificados.** A leitura dos valores de elementos e atributos presentes nas instâncias XML é feita através da propriedade *originalPath,* presente nos conceitos recém criados, tal como o conceito *degree* presente na Figura 5.13(b). Ao definir o domínio do dado relativo ao conceito qualificado, através da associação de propriedades de domínio, devem ser consideradas as seguintes situações: *(i)* caso seja identificado um conceito constante, como o conceito *job,* presente na Figura 5.13(b), deve ser associada a propriedade *hasConstantValue*; *(ii)* caso seja identificado um conceito disperso, como o conceito *completeName.lastName*, deve ser associada a propriedade *hasDispersedValue*; e *(iii)* caso nenhum dos

casos anteriores ocorra, para cada valor distinto encontrado, é associada uma propriedade *hasQualifiedValue,* tal como as propriedades *hasQualifiedValue* presentes na Figura 5.13(b) que associam ao conceito *degree* os valores *"master"*, *"graduated"* e *"doctor"*.

- **Regra 7 – Criação do domínio de conceitos quantificados.** A leitura dos valores de elementos e atributos XML é feita de forma análoga à regra 6. Com relação aos conceitos quantificados do tipo coleção, como o conceito *masterStudents* (Figura 5.13(c)), traduzido a partir do elemento XML *masterStudents* (Figura 5.13(a)), são tomados como valor de entrada a quantidade de elementos filhos. Aos conceitos quantificados do tipo coleção deve ser acrescentado o valor "0", ao conjunto de valores distintos, caso o referido conceito não esteja presente em todas as instâncias XML. Na seqüência, o conjunto de valores distintos encontrados são discretizados segundo o método apresentado anteriormente. Ainda, para cada valor discretizado deve ser associada uma propriedade *hasQuantifiedValue,* tal como as propriedades hasQuantifiedValue presentes na Figura 5.13(b), que associam ao conceito *age* os valores *"30"*, *"38"* e *"46"*.

Após a aplicação desta última etapa, a saída desta fase é a instância do esquema conceitual CDX-Tree, representativa da estrutura e domínio de dados das instâncias XML tomadas como dados de entrada. Vale ressaltar que todas as regras de pré-processamento propostas, são aplicadas através da leitura das instâncias XML e posterior representação na CDX-Tree, ou seja, não há operações de atualização resultante de pré-processamento sobre as instâncias XML, o que poderia tornar o processo proibitivo quanto ao custo de processamento.

5.2 Fase de Descoberta

Esta seção detalha a segunda fase da abordagem DIInCX, chamada de fase de *Descoberta*. Esta fase tem como principal objetivo a aplicação de um algoritmo de mineração de RA's sobre as instâncias XML, que utiliza a instância do esquema conceitual hierárquico CDX-Tree, criada na fase anterior. A fase de Descoberta é composta por duas etapas independentes: *Descoberta de RIS simples* e *Mineração de RIS*. Na etapa de *Descoberta de RIS simples* é feita a análise da estrutura da CDX-Tree,

assim como dos dados presentes nela, para descobrir RIS's. A etapa de *Mineração de RIS,* através de um algoritmo de mineração de RA's com adaptações propostas, descobre RA's que possam ser traduzidas para RIS's válidas.

Há na literatura diversos algoritmos para mineração de RA's, alguns inclusive com foco em tipos de domínios de dados específicos. Esta área tem recebido crescente interesse da comunidade de pesquisa, se mostrando em constante evolução. A constante evolução na mineração de RA's motivou a escolha por manter a abordagem e as adaptações propostas a algoritmos de RA's, fracamente acopladas ao algoritmo adotado, possibilitando a adoção de novos algoritmos no futuro. Este objetivo é alcançado, mantendo o conjunto de dados de entrada para a etapa de Mineração de RIS, comuns a algoritmos de mineração de RA's. Como dado de entrada para a etapa de Mineração de RIS é dada uma matriz de transações com o auxílio da instância CDX-Tree, possuindo assim um formato de entrada comum a algoritmos de mineração de RA's. No mais uma análise de algoritmos de mineração de RA's se mostra como um trabalho futuro relevante, estando a avaliação de questões no que tange desempenho fora do escopo deste trabalho.

As adaptações propostas a algoritmos de mineração de RA's visam direcionar o algoritmo a localizar RA's que possam ser traduzidas para RIS's válidas e de menor complexidade de avaliação. RIS's de menor complexidade de avaliação são priorizadas através de um critério de restrição (ou critério de parada), que permite ordenar as RIS's conforme a quantidade de objetos envolvidos no componente condição restritiva e restringente da RIS. RIS's de baixa complexidade de avaliação podem ser facilmente incorporadas por sistemas de informação que pretendam dar suporte a elas. Baseado no estudo de linguagens de especificação de RIS's e sistemas de validação de RIS's, observou-se que a complexidade do sistema de gerenciamento de RIS's é diretamente proporcional à complexidade de avaliação das categorias de RIS's que estes sistemas pretendem dar suporte.

As adaptações propostas são as seguintes*: (i)* ordenar as RA's descobertas segundo a complexidade dos componentes antecedente e consequente respectivamente; e *(ii)* utilizar esta complexidade como critério de restrição às RA's descobertas. O uso da complexidade da RA, como critério de restrição, auxilia não só a restringir o tempo

de processamento, como também a focar na descoberta de RIS's que sejam compostas por condições restritivas de menor complexidade. De fato, RIS's com condições restritivas de menor complexidade são mais facilmente implementadas. Ao contrário, RIS's com condições restritivas muito complexas tendem a ser pouco realistas. Mesmo assim, embora o foco sejam RIS's com menor complexidade, RIS's complexas não são descartadas, ficando a critério do usuário especialista considerá-las ou não.

Uma vez que as adaptações se referem à complexidade das RA's descobertas (RA's são o produto comum de algoritmos de mineração de RA's), elas podem ser aplicadas, tanto na forma de adaptações ao algoritmo de mineração de RA's, quanto na forma de regras de pós-processamento. Assim sendo, as adaptações propostas consistem basicamente da análise da quantidade de elementos envolvidos nos componentes antecedente e conseqüente de uma RA. Componentes estes que serão traduzidos para os componentes condição restritiva e restringente de uma RIS. Tal constatação realça o caráter genérico das adaptações propostas, possibilitando sua aplicação sobre outros algoritmo de mineração de RA's, uma vez que as regras podem ser analisadas inclusive externamente ao algoritmo (sobre as RA's resultantes do algoritmo), ao invés de implementadas internamente a ele. Contudo, a implementação interna das adaptações propostas pode evitar a geração de RA's desnecessárias.

5.2.1 Etapa de Descoberta de RIS Simples

Esta seção descreve a etapa de *Descoberta de RIS Simples,* que foca na descoberta de RIS's de menor complexidade. Esta descoberta se dá através da análise da estrutura e dos dados presentes no esquema conceitual hierárquico CDX-Tree para descobrir as seguintes RIS's: *(i)* elementos ou atributos obrigatórios; *(ii)* elementos ou atributos restringidos a conjuntos de valores; *(iii)* elementos ou atributos constantes; *(iv)* cardinalidade relacionada a coleções de elementos; e (v) máximos e mínimos relacionados a valores de elementos ou atributos. As RIS's descobertas nesta etapa são especificadas em SWRL e armazenadas na instância do esquema conceitual hierárquico CDX-Tree (ver seção 5.3).

Com relação ao item *(i)*, uma restrição de obrigatoriedade é descoberta a partir da propriedade *relevance* (Figura 5.3-e), presente em instâncias das classes filhas da

classe *nodeElement* da CDX-Tree. A propriedade *relevance* tem por função representar a quantidade de vezes que o elemento ou atributo XML se faz presente no domínio de dados. Esta pode ser comparada com a propriedade *instancesAmount* (Figura 5.3-d), presente em instâncias da classe *rootTree* (Figura 5.6), que tem por função representar a quantidade de instâncias que compõem o domínio, a fim de descobrir RIS's envolvendo obrigatoriedade absoluta. Uma RIS que representa uma obrigatoriedade absoluta define que o elemento ou atributo deve estar presente em todas as instâncias do domínio. A RIS1 exemplificada na Figura 5.14 é um exemplo de obrigatoriedade absoluta do atributo *degree*[3]. Uma obrigatoriedade absoluta é definida através de uma implicação com antecedente vazio, pois a restrição que esta especifica deverá ser aplicada independente de condição restritiva. O conseqüente desta é definido através de uma instância da classe do conceito a ser restringido, no exemplo em questão a instância *degree* da classe *leafNode*. Também pode ser definida uma RIS que representa uma obrigatoriedade relativa ao elemento pai do elemento ou atributo em questão. Esta se dá comparando a propriedade *relevance* de instâncias de subclasses da classe *nodeElement* com suas respectivas instâncias de classe pai. A RIS2, exemplificada na Figura 5.14, é um exemplo de RIS representando uma obrigatoriedade do elemento Name ("/professor/masterStudents/masterStudent/name") relativa ao elemento *masterStudents*. Conforme demonstrado, uma obrigatoriedade relativa é representada na abordagem DIInCX como uma implicação do conceito que compõe a condição restritiva da RIS, para o conceito que compõe o componente restringido da RIS.

[3] Expressões com antecedente vazio são tratadas como verdadeiras em SWRL.

RIS1: → leafNode(degree)

RIS2: collectionNode(masterStudents) → leafNode(masterStudent.name)

RIS3: hasQualifiedValue(degree,?degreeValue)^
swrlb:listConcat(?degreeValueSet,”master”,”doctor”,”graduated”) →
swrlb:member(?degreeValue,?degreeValueSet)

RIS4: leafNode(job)→ hasConstantValue(job,”professor”)

RIS5: hasQuantifiedValue(age,?ageValue) →
swrlb:lessThanOrEqual(?ageValue,60)^swrlb:greaterThanOrEqual(?ageValue,20)

RIS6: hasQuantifiedValue(masterStudents,?masterStudentsAmount) →
swrlb:lessThanOrEqual(?masterStudentsAmount,6)
^swrlb:greaterThanOrEqual(?masterStudentsAmount,2)

Fonte: Primária.

Figura 5.14 – Exemplo de RIS’s especificadas em SWRL.

Na tradução de RIS’s que envolvam valores de conceitos adota-se a subclasse da propriedade *hasValue* correspondente a cada tipo de conceito, como a propriedade *hasQualifiedValue* para conceitos Qualificados (conceito *degree* RIS3 Figura 5.14) e *hasQuantifiedValue* para conceitos Quantificados (conceito *age* RIS5 Figura 5.14). Quanto ao nome de variáveis, como a variável *?degreeValue,* exemplificada na RIS3 da Figura 5.14, ele é determinado pelo nome do conceito ao qual a variável está associada, seguido da string *“Value”* para instâncias da classe *leafNode* e *“Amount”* para instâncias da classe *collectionNode.*

Com relação ao item *(ii)*, a tarefa de determinar se o domínio de um elemento ou atributo é restrito a um conjunto de valores se dá analisando a propriedade *hasQualifiedValue* (Figura 5.4-a) presente em conceitos qualificados na CDX-Tree. Esta propriedade compõe o conjunto de valores cabíveis ao conceito. A determinação de uma RIS na forma de uma enumeração para o conceito deve ser definida caso a quantidade de propriedades *hasQualifiedValue* seja menor ou igual ao parâmetro de entrada da abordagem DIInCX chamado *Quantidade máxima de itens para enumeração* (Tabela 5.1). Isto, porque enumerações na forma de RIS só se tornam relevantes quando compostas por conjuntos de baixa cardinalidade.

A RIS3 presente na Figura 5.14 exemplifica o item *(ii)*. A definição de uma RIS envolvendo enumeração, utiliza tipos primitivos da SWRL denominados *listConcat* e *member* a fim de atribuir um conjunto de valores a uma lista e determinar se um elemento faz parte de uma lista, respectivamente. A especificação da RIS, conforme

exemplificado pela RIS3, envolve a criação de uma lista composta pelo conjunto de elementos possíveis para o conceito em questão (*swrlb:listConcat(?degreeValueSet,"master","doctor","graduated")*), criação de uma variável relacionando esta ao valor do conceito através da propriedade *hasQualifiedValue* (*hasQualifiedValue(degree,?degreeValue)*) e posterior implicação de que esta variável representando o valor do conceito em questão seja membro da lista criada (*swrlb:member(?degreeValue,?degreeValueSet)*).

Com relação ao item *(iii)*, para determinar se um elemento ou atributo possui valor constante, verifica-se a propriedade *hasConstantValue* (Figura 5.4-c) presente em conceitos do tipo Constantes na estrutura CDX-Tree. Esta propriedade armazena o valor considerado constante do elemento ou atributo em questão. A RIS4 presente na Figura 5.14 é um exemplo de RIS sobre um conceito Constante. Ela estabelece o valor do elemento *job* como a string constante *"professor"*. A especificação deste tipo de RIS é similar a uma RIS envolvendo uma obrigatoriedade relativa. No entanto, a existência do conceito antecedente implica na propriedade *hasConstantValue* como conseqüente, atribuindo assim o valor constante ao conceito.

Com relação ao item *(iv)*, a determinação de cardinalidades máximas e mínimas para coleções de elementos traduzidos para instâncias da classe *collectionNode* na CDX-Tree se dá através da análise da propriedade *hasQuantifiedValue* (Figura 5.4-b). De forma semelhante é feita a determinação de máximos e mínimos valores para elementos e atributos, comentada no item *(v)*. Para ambos os casos, é tomado como máxima cardinalidade ou máximo valor, o maior valor desta propriedade e como mínima cardinalidade ou mínimo valor, o menor valor da propriedade. Na definição de uma RIS de máximos e mínimos valores ou cardinalidades, são adotados os tipos primitivos da SWRL denominados *lessThanOrEqual, lessThan, greaterThan* e *greaterThanOrEqual,* juntamente com a propriedade *hasQuantifiedValue* da CDX-Tree. A RIS5 exemplificada na Figura 5.14, mostra uma RIS que determina o máximo e mínimo valor para um elemento. Ela define que o valor do elemento *age* deve ser menor ou igual a *60* e maior ou igual a *20*. A RIS6 exemplificada na Figura 5.14, mostra uma RIS que determina a máxima e mínima cardinalidade para o elemento *masterStudents*.

Através desta RIS a quantidade de elementos filhos do elemento *masterStudents* é restrita a um valor menor ou igual a *6* e maior ou igual a *2*.

5.2.2 Etapa de Mineração de RIS

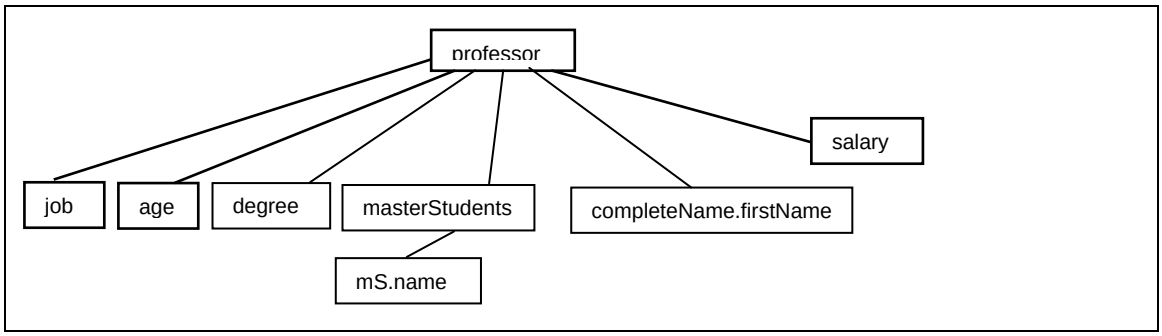
Esta seção detalha a etapa de *Mineração de RIS*. Esta etapa tem por objetivo aplicar um algoritmo de mineração de RA's com as adaptações propostas (apresentadas na seção 5.2) sobre um conjunto de instâncias XML, guiado pela instância do esquema conceitual hierárquico CDX-Tree criado na fase de Pré-processamento (seção 5.1).

A etapa inicia pela leitura da instância CDX-Tree a fim de coletar os dados de entrada para o algoritmo de mineração. A Tabela 5.2 apresenta como exemplo um conjunto de instâncias XML com seus respectivos valores de elementos e/ou atributos. A leitura dos dados de entrada através da CDX-Tree se dá percorrendo a estrutura hierárquica da mesma e acessando através da propriedade *originalPath* (Figura 5.3-c) o valor de elementos e atributos nas instâncias XML. A Figura 5.15 apresenta um exemplo de representação gráfica da estrutura hierárquica resumida da CDX-Tree correspondente às instâncias XML apresentadas na Tabela 5.2.

Tabela 5.2 - Conjunto de intâncias XML de entrada.

	completName/firstName	**Degree**	**Age**	**Salary**	**job**	**masterStudents**
i1	Cristiane	Doctor	60	5000	professor	3
i2	Inácio	Máster	40	2000	professor	0
i3	Mathina	Graduated	20	1000	professor	0
i4	Marta	Graduated	30	1000	professor	0
i5	Iolanda	Doctor	50	3000	professor	4
i6	Ivo	Doctor	55	4000	professor	6
i7	Ivani	Máster	35	2000	professor	0

Fonte: Primária.



Fonte: Primária.

Figura 5.15. Representação gráfica da estrutura hierárquica da CDX-Tree.

O objetivo da leitura das instâncias XML, a partir da CDX-Tree é construir uma *matriz de transações* que é usada como valor de entrada para o algoritmo de mineração. Ela é composta por atributos de entrada e um conjunto de transações *T*. *T* representa o conjunto de valores presentes nas instâncias XML, enquanto os atributos que compõem uma transação são definidos a partir de valores de subpropriedades da propriedade *hasValue* (*hasQuantifiedValue* e *hasQualifiedValue*) associadas a instâncias de subclasses da classe *nodeElement* presentes na CDX-Tree. Dentre estas, são consideradas relevantes para o algoritmo de mineração os conceitos *Qualificados,* como o conceito *degree,* e *Quantificados* como os conceitos *masterStudents*, *age* e *salary* exemplificados na Tabela 5.3. Conceitos *Dispersos* como o conceito *completeName.firstName* (Figura 5.15), assim como conceitos *Constantes* como o conceito *job* (Figura 5.15), são considerados irrelevantes na tarefa de mineração de RA's e desta forma não são considerados nesta etapa. Tanto conceitos *Dispersos* quanto *Constantes* não direcionam na descoberta de associações relevantes entre atributos, produzindo apenas uma quantidade maior de regras de associações irrelevantes.

Conceitos pertencentes a instâncias da classe *collectionNode* como o conceito *mS.name*, são considerados como pertencentes a uma coleção de subárvores da instância CDX-Tree. A fim de reduzir a quantidade de atributos de entrada para o algoritmo de mineração, coleções de subárvores são tratadas com uma nova iteração da etapa de Mineração de RIS. Cada subárvore pertencente à coleção de subárvores é considerada como uma nova transação $t_n \in T'$ (de forma análoga a uma instância XML). A definição das subárvores que devem ser consideradas é feita pela quantidade de atributos considerados relevantes para a etapa de Mineração de RIS (conceitos *Quantificados* e *Qualificados*) que a subárvore possui. O parâmetro *Quantidade mínima de conceitos*

para consideração de subárvores indica a quantidade mínima de conceitos considerados relevantes que uma subárvore deve possuir para ser considerada. Por exemplo, o conceito *mS.name* não seria levado em consideração, pois a coleção de subárvores com raiz no conceito *masterStudents* possui uma única subárvore com raiz no conceito filho *mS.name.* Assim, não havendo mais conceitos filhos para tomar como atributos de entrada para a etapa de *Mineração de RIS*, não é possível descobrir RA's para a subárvore *mS.name*.

A Tabela 5.3 apresenta um exemplo da matriz de transações gerada a partir das instâncias XML exemplificadas na Tabela 5.2 e da CDX-Tree da Figura 5.15.

Tabela 5.3 - Exemplo de matriz de transações originada de instâncias XML.

	degree	**age** **50% discretização**	**Salary** **50% discretização**	**masterStudents** **50% discretização**
t1	Doctor	>=20To<40	>=3000To<=5000	>=0To<4
t2	Master	>=40To<=60	>=1000To<3000	>=0To<4
t3	Graduated	>=20To<40	>=1000To<3000	>=0To<4
t4	Graduated	>=20To<40	>=1000To<3000	>=0To<4
t5	Doctor	>=40To<=60	>=3000To<=5000	>=4To<=6
t6	Doctor	>=40To<=60	>=3000To<=5000	>=4To<=6
t7	Master	>=20To<40	>=1000To<3000	>=0To<4

Fonte: Primária.

A busca de valores para inclusão na matriz de transações, quanto a conceitos *Qualificados*, se dá pela simples leitura de seus valores nas instâncias XML. No caso de uma transação, onde o referido conceito não esteja presente (conceitos com relevância menor que 100%) deve ser associado a string "none", como valor para o conceito na transação, significando sua ausência. Com relação aos conceitos *Quantificados,* o valor a ser tomado como dado de entrada é a correlação entre o valor do elemento na referida instância e o intervalo de dado estabelecido através de duas propriedades *hasQuantifiedValue* (em seqüência numérica), presentes no conceito em questão. Por exemplo, o elemento *salary* referente à instância XML *i1* exemplificada na Tabela 5.2, possuindo como valor *5000*, se enquadra no intervalo estabelecido entre o valor das propriedades *hasQuantifiedValue*: *3000* e *5000* respectivamente. Os operadores de comparação adotados em cada intervalo são definidos como ">=" e "<=" para o mínimo

e máximo valor do domínio respectivamente, enquanto aos demais intervalos de valores numéricos devem ser agregados os respectivos operadores “>=” ou “<” a fim de compô-los. Por exemplo: “>=n1To<n2”;“>=n2To<n3”;”>=n3To<=n4”. Desta forma, o valor a ser tomado como dado de entrada para o algoritmo de mineração para o exemplo acima comentado deve ser a string “*>=3000To<=5000”,* significando o referido intervalo conforme exemplificado na transação *t1* demonstrada na Tabela 5.3.

Após a matriz ter sido montada ela é passada como dado de entrada para o algoritmo de mineração de RA’s, que se somam os seguintes parâmetros: *(i) Relevância mínima* (suporte mínimo da RA), quantidade de vezes que uma RA acontece no conjunto de transações *T* que compõem o domínio de dados; *(ii) Complexidade máxima do componente restringido* (quantidade de elementos envolvidos no componente conseqüente da RA), complexidade máxima desejada para o componente restringido das RIS descobertas; *(iii) Complexidade máxima do componente condição restritiva* (quantidade de elementos envolvidos no componente antecedente da RA), complexidade máxima desejada para o componente condição restritiva das RIS descobertas; e *(iv) Quantidade máxima de RIS descobertas* (Quantidade de máxima de RA’s a serem descobertas pelo algoritmo de mineração de RA’s), a quantidade máxima de RIS que se deseja descobrir. Vale ressaltar que conforme comentado, a restrição através dos parâmetros *Complexidade máxima do componente restringido* e *Complexidade máxima do componente condição restritiva* podem ser aplicados tanto na forma de adaptações ao algoritmo de mineração de RA, como na forma pós-processamento, sendo aplicada sobre as RA’s descobertas pelo algoritmo. A Tabela 5.1 apresenta todos os parâmetros de entrada da abordagem DIInCX.

Nesta etapa são descobertas RA’s que são traduzidas, posteriomente, para RIS envolvendo elementos e/ou atributos em diferentes níveis hierárquicos nas instâncias XML. Também são descobertas RA’s envolvendo coleções de elementos (traduzidos para instâncias da classe collectonNode) e seus relacionamentos com outros elementos ou atributos (traduzidos para instâncias da classe leafNode). A Figura 5.16 apresenta um conjunto de RA’s descobertas nesta etapa. Na próxima etapa da abordagem estas RA’s descobertas são traduzidas para RIS’s especificadas em SWRL.

RA1: degree(“doctor”) → courses(“>=1To<3”)

RA2: salary(“>=1000To<3000”) → position(“none”)

RA3: publications(“>=0To<3”) → masterStudents(“>=0To<4”)

RA4: degree(“doctor”) → salary(“>=3000To<=5000”)

RA5: degree(“master”) → salary(“>=1000To<3000”)

Fonte: Primária.

Figura 5.16 – Exemplos de regras de associação descobertas.

5.3 Fase de Conversão

Esta seção detalha a fase final da abordagem DIInCX, chamada de fase de *Conversão*. A fase de Conversão recebe como dados de entrada as RA’s descobertas na fase de Descoberta e as traduz para RIS’s especificadas em uma linguagem de representação de regras de conhecimento chamada SWRL (SWRL, 2007). Ao final desta etapa as RIS’s especificadas em SWRL são adicionadas à instância do esquema conceitual hierárquico CDX-Tree.

A linguagem SWRL foi adotada por ser uma linguagem de alto nível, expressiva o suficiente para representação de RIS’s XML, além de possuir um mecanismo de inferência similar à definição de uma RA. O principal motivo do emprego da SWRL como linguagem de definição das RIS’s descobertas foi seu alto nível de abstração, o que facilita sua posterior tradução para linguagens específicas de definição de RIS’s (XCML, XCSL). Ainda com relação à SWRL, é importante observar que seu foco é prover regras que quando aplicadas sobre uma ontologia, possibilitam a geração de novo conhecimento sobre ela. Na abordagem DIInCX o papel da ontologia e seus indivíduos pode ser compreendido como representado pela estrutura CDX-Tree e sua instância, respectivamente. No entanto, no contexto deste trabalho, a SWRL é adotada como linguagem para especificação de RIS’s a fim de garantir a manutenção da integridade do conjunto de instâncias XML que a instância CDX-Tree representa. A tarefa de aplicar as RIS’s especificadas em SWRL pode ser realizada por uma ferramenta que aplique as RIS’s diretamente sobre as instâncias XML ou ainda que traduza as RIS’s para linguagens específicas de definição de RIS’s.

Além de traduzir RA's para RIS's, esta fase contempla as seguintes atividades: *(i)* unificar itens compostos por valores discretizados nas RA's descobertas; e *(ii)* descartar RA's redundantes em relação ao componente *conseqüente*.

O item *(i)* refere-se aos elementos ou atributos que foram traduzidos para conceitos Quantificados. RA's compostas por componentes conseqüente idênticos, quando diferenciadas apenas pelo valor de um conceito Quantificado em seu componente antecedente, devem ser unificadas se o intervalo entre estas não apresentar descontinuidade. A Figura 5.17 exemplifica o item *(i)*. A RA apresentada na Figura 5.17 RA1, estabelece que o conceito *courses,* possuindo como valor a string *">=1To<3",* implica no conceito *degree* com o valor *"doctor"*. A string *">=1To<3"* no conceito *courses* (instância da classe collectionNode) representa um intervalo definido a partir da quantidade de elementos filhos, discretizados pela *Regra 7* discutida na fase de Pré-processamento. Este mesmo raciocínio se aplica à RA2 presente na Figura 5.17.

Quando se traduzem RA's como as exemplificadas na Figura 5.17, que envolva conceitos *Quantificados,* são adotados os tipos primitivos da SWRL *lessThanOrEqual, lessThan, greaterThan* e *greaterThanOrEqual,* juntamente com a propriedade *hasQuantifiedValue* da CDX-Tree associando o conceito a uma variável. A associação de conceitos a variáveis deve ser representada no componente *antecedente* enquanto a restrição da referida variável deve ser representada no componente *conseqüente*. Conceitos *Qualificados,* quando presentes em uma RA, seja no componente *antecedente* ou *consequente*, são traduzidos utilizando a propriedade *hasQualifiedValue* conforme demonstrado através do conceito Qualificado *degree* presente na Figura 5.17 RIS1.

A RIS1 (Figura 5.17) representa a RIS resultante da união e posterior tradução das RA's RA1 e RA2. A fim de representar a restrição em questão, a variável *?coursesAmount*, estando limitada ao intervalo maior ou igual a 1 e menor que 5, através da propriedade *hasQuantifiedValue* quando é aplicada a conceitos *courses*, restringe o conceito *degree* a possuir o valor da propriedade *hasQualifiedValue* igual a *"doctor"*. Desta forma, as RA's RA1 e RA2 (Figura 5.17), quando traduzidas para a RIS1 apresentada, representam uma *Restrição Qualificada quanto ao tipo de limitação imposta*.

RA1: courses(“>=1To<3”) → degree(“doctor”)
RA2: courses(“>=3To<5”) → degree(“doctor”)

RIS1: hasQuantifiedValue(courses,?coursesAmount)^ swrlb:lessThan(?coursesAmount,5)
^ swrlb:greaterThanOrEqual(?coursesAmount,1) → hasQualifiedValue(degree,”doctor”)

Fonte: Primária.

Figura 5.17 – Tradução de regra de associação para RIS quantificada.

Com relação ao item *(ii)*, este trabalho define uma RA como redundante quando representa a mesma implicação que outro conjunto de RA’s ($X \rightarrow Y.Z = X \rightarrow Z \cup X \rightarrow Y$). Desta forma, a fim de eliminar RA’s redundantes, regras compostas por conseqüente complexo, cuja complexidade seja imediatamente inferior ao parâmetro *Complexidade máxima do componente restringido* fornecido pelo usuário especialista são priorizadas. Assim, RA’s redundantes compostas por conseqüente de complexidade inferior devem ser descartadas.

Na Figura 5.18 é apresentado um exemplo referente ao item *(ii)*. A RA1 define que o conceito *degree* possuindo o valor *“master”,* implica no conceito *salary* possuindo como valor a string *“>=800To<1200”* e o conceito *laboratory* possuindo o valor *“none”*. A RA2 define que o conceito *degree* possuindo o valor “master” implica no conceito *salary* possuindo como valor a string *“>=800To<1200”*. A definição de um intervalo a partir da string *“>=800To<1200”* se dá de forma análoga ao discutido no item *(i)*. A RA3 define que o conceito *degree* possuindo o valor *“master”* implica no conceito *laboratory* possuindo o valor *“none”*. Observa-se ainda que os componentes antecedente das RA1, RA2 e RA3 são idênticos e que o conseqüente da RA1 é equivalente à união dos componentes conseqüentes das RA2 e RA3. Desta forma, procede-se com o descarte das RA2 e RA3, sendo a RA1 traduzida para a RIS1 demonstrada na Figura 5.18. Embora tenha sido previamente comentado que RA’s de menor complexidade geram RIS’s de menor complexidade, mais diretas e fáceis de serem incorporadas por sistemas de informação, a abordagem DIInCX deixa a cargo do usuário especialista a decisão sobre a complexidade das RIS’s que devem ser descobertas através do parâmetro *Complexidade máxima do componente restringido*.

A RIS1 estabelece que a variável *?salaryValue,* representando qualquer valor possível de ser atribuído à propriedade *hasQuantifiedValue,* quando aplicada a

conceitos *salary*, é restringida a um valor maior ou igual a 800 e menor que 1200, quando o conceito *degree* possui a propriedade *hasQualifiedValue* com valor igual à string *"master" e o conceito laboratory* possui a propriedade *hasQualifiedValue* com valor igual à string *"none"*. No contexto das instâncias XML esta restrição restringe o valor do elemento XML *salary* a maior ou igual a 800 e menor que 1200, quando o atributo *degree* possuir o valor *"master"* e o elemento *laboratory* estiver ausente[4]. Desta forma, as RA's RA1, RA2 e RA3 (Figura 5.18), quando traduzidas para a RIS1 apresentada, representam uma *Restrição Quantificada e uma Qualificada quanto ao tipo de limitação imposta*.

RA1: degree("master") → laboratory("none")^salary(">=800To<1200")
RA2: degree("master") → salary(">=800To<1200")
RA3: degree("master") → laboratory("none")

RIS1: hasQualifiedValue(degree,"master")^hasQuantifiedValue(salary,?salaryValue)→
hasQualifiedValue(laboratory,"none")
^swrlb:lessThan(?salaryValue,1200)^swrlb:greaterThanOrEqual(?salaryValue,800)

Fonte: Primária.

Figura 5.18 – Tradução de regras de associação para RIS em SWRL.

Após a tradução de todas as RA's para RIS's especificadas em SWRL, elas são agregadas ao esquema conceitual hierárquico CDX-Tree. O usuário especialista é chamado a descartar as RIS's que julgar pouco relevantes e ou desnecessárias. Por fim, a abordagem DIInCX produz como produtos finais, uma estrutura conceitual hierárquica definida em OWL chamada CDX-Tree com RIS's explícitas agregadas. Estas RIS's descobertas, a partir de informação implícita, se encontram especificadas em SWRL e representam a principal contribuição da abordagem. A partir deste ponto, qualquer sistema que pretenda dar suporte a RIS's, pode incorporar as RIS's descobertas e especificadas em uma linguagem com alto nível de abstração.

5.4 Categorias de RIS's XML descobertas

Como forma de definir o escopo da abordagem é possível definir, segundo a taxionomia para RIS's XML proposta neste trabalho, as categorias de RIS capazes de serem descobertas pela abordagem DIInCX.

[4] Conforme discutido na seção 5.2.2, a ausência de um elemento e/ou atributo é representada pela associação deste à string "none".

Analisando o aspecto *Quanto ao tipo de limitação imposta* é possível observar que a abordagem DIInCX é capaz de identificar apenas RIS pertencentes à categoria chamada *estado*. Isto porque a análise das instâncias XML é atemporal. Dentre as RIS's de *estado* é possível identificar RIS's *estáticas*. Embora observando conjuntos de RIS's seja possível identificar comportamento dinâmico, por questões de simplicidade opta-se na abordagem DIInCX, representar as RIS's em um formato mais simples. Por exemplo: Quando o valor do atributo *degree* for igual a *"master"* o valor do elemento *salary* deveria ser igual a *">=5000To<=6000"*, enquanto, quando o valor do atributo *degree* for igual a *"doctor",* o valor do elemento *salary* deveria ser igual a *">=6000To<=7000"*. Na abordagem DIInCX esta RIS seria representada de forma desmembrada em duas RIS's como segue: *degree("master")* → *salary(">=5000To<=6000")* e *degree("doctor")* → *salary(">=6000To<=7000")*. Ainda, quanto às RIS estáticas é possível identificar RIS: qualificadas, quantificadas ou de existência dependente. Conforme previamente discutido, os três tipos de categorias (qualificadas, quantificadas ou de existência dependente) podem ser percebidos implicitamente na semântica das RA's descobertas que posteriormente são traduzidas para RIS's especificadas em SWRL.

Com relação ao aspecto *Quanto ao alcance*, é possível distinguir RIS's em todas as categorias: *item de dado, tupla, elemento* e *documento*. Isto porque através das regras de pré-processamento propostas, todos os tipos de elementos XML são abordados de igual forma, inclusive tipos de elementos com comportamento especial como os elementos XML que compõem coleções de outros elementos. Elementos que compõem coleções por si só não possuem informação semântica relevante. Entretanto, DIInCX é capaz de capturar relacionamentos entre a quantidade de elementos filhos destes e os demais elementos e/atributos que compõem o domínio de dados XML.

O aspecto *Quanto à forma* aborda a notação empregada na representação da RIS, ou seja, a forma como é especificada. Desta forma, objetivando simplicidade e fácil incorporação por sistemas de informação, representa-se as RIS's na forma de expressões booleanas. No mais, é possível distinguir dentre as RIS descobertas tanto RIS's *baseadas em expressões booleanas simples* quanto *compostas*.

A análise do aspecto *Quanto ao momento da verificação* é extremamente dependente do sistema de gerenciamento adotado em SGBD's. Assim sendo, na abordagem DIInCX, todas as RIS são especificadas como pertencentes à categoria *imediata*.

Quanto ao aspecto *Quanto à ação a ser executada*, todas as RIS's descobertas e representadas pela abordagem proposta pertencem à categoria denominada *informativa*. A escolha por representar as RIS's descobertas como pertencentes à categoria *informativa*, é representar as RIS's em um formato tão simples quanto possível, uma vez que RIS's *informativas* são mais simples que RIS's *ativas*.

6 ESTUDO DE CASO

Este capítulo apresenta um experimento prático aplicado à abordagem *DIInCX* proposta neste trabalho. O experimento tem por objetivo demonstrar que a abordagem é adequada para descobrir e representar RIS's XML.

O algoritmo de mineração adotado nos experimentos foi baseado no clássico algoritmo de mineração de RA chamado ***Apriori*** (AGRAWAL et al, 1993). A escolha deste algoritmo foi motivada pelo fato de ser um algoritmo já amplamente discutido e com diversas implementações disponíveis na literatura. Entretanto, conforme discutido anteriormente, o objetivo proposto é manter tanto a abordagem, quanto as adaptações propostas ao algoritmo de mineração de RA's, genéricas o suficiente, para serem aplicadas com o suporte de diversos outros algoritmos (HAN et al., 2000; BRAGA et al., 2002; GYORÖDI, 2003; BAYARDO, 2004) ou otimizações do próprio algoritmo Apriori (AGRAWAL & SRIKANT, 1994; SRIKANT et al., 1997).

Diversos algoritmos para mineração de RA se encontram à disposição na literatura. Entretanto, as principais motivações para as demais abordagens são principalmente duas: desempenho e melhor resultado quando aplicados sobre tipos de domínios de dados específicos. Trabalhos como de HIPP et al. (2000), demonstram as semelhanças entre abordagens para mineração de RA's, traçando comparativos entre estas. Contudo, questões envolvendo desempenho estão fora do escopo desta versão da abordagem proposta. O objetivo da abordagem DIInCX é ser adaptável a qualquer domínio de dado. Contudo, o interesse da comunidade de pesquisa e o crescente desenvolvimento nesta área sugerem que manter a abordagem e as adaptações propostas ao algoritmo de mineração de RA's genéricas são uma boa escolha, pois possibilita a adoção de algoritmos específicos para cada tipo de domínio, além de evoluções e otimizações destes. Uma análise da incorporação de demais algoritmos de mineração de RA com foco em desempenho é um trabalho futuro relevante.

A fim de viabilizar a experimentação da abordagem DIInCX, um conjunto de tarefas, tais como aquelas propostas na fase de Conversão, foram implementadas no ambiente Java. Todas as tarefas implementadas atuam de forma independente entre si (fracamente acopladas), propiciando a observação isolada dos resultados de sua

aplicação. O editor de ontologias *PROTÉGÉ* (PROTÉGÉ, 2007) foi utilizado na modelagem do esquema conceitual hierárquico CDX-Tree. Com relação ao algoritmo de mineração de RA`s, foram adotadas duas implementações nos testes: uma implementação de domínio público do algoritmo *Apriori* (DATAMINING, 2007) e a implementação do algoritmo *Apriori* disponível na biblioteca *WEKA* (WEKA, 2007). Testes foram feitos com as adaptações propostas ao algoritmo de mineração de RA`s, tanto internamente ao algoritmo como na forma de regras de pós-processamento, a fim de comprovar sua aplicabilidade.

O domínio adotado para o experimento são professores universitários, sendo composto por 40 instâncias XML. O conjunto de instâncias XML de entrada foi gerado artificialmente, seguindo um conjunto de premissas apresentadas na Tabela 6.1.

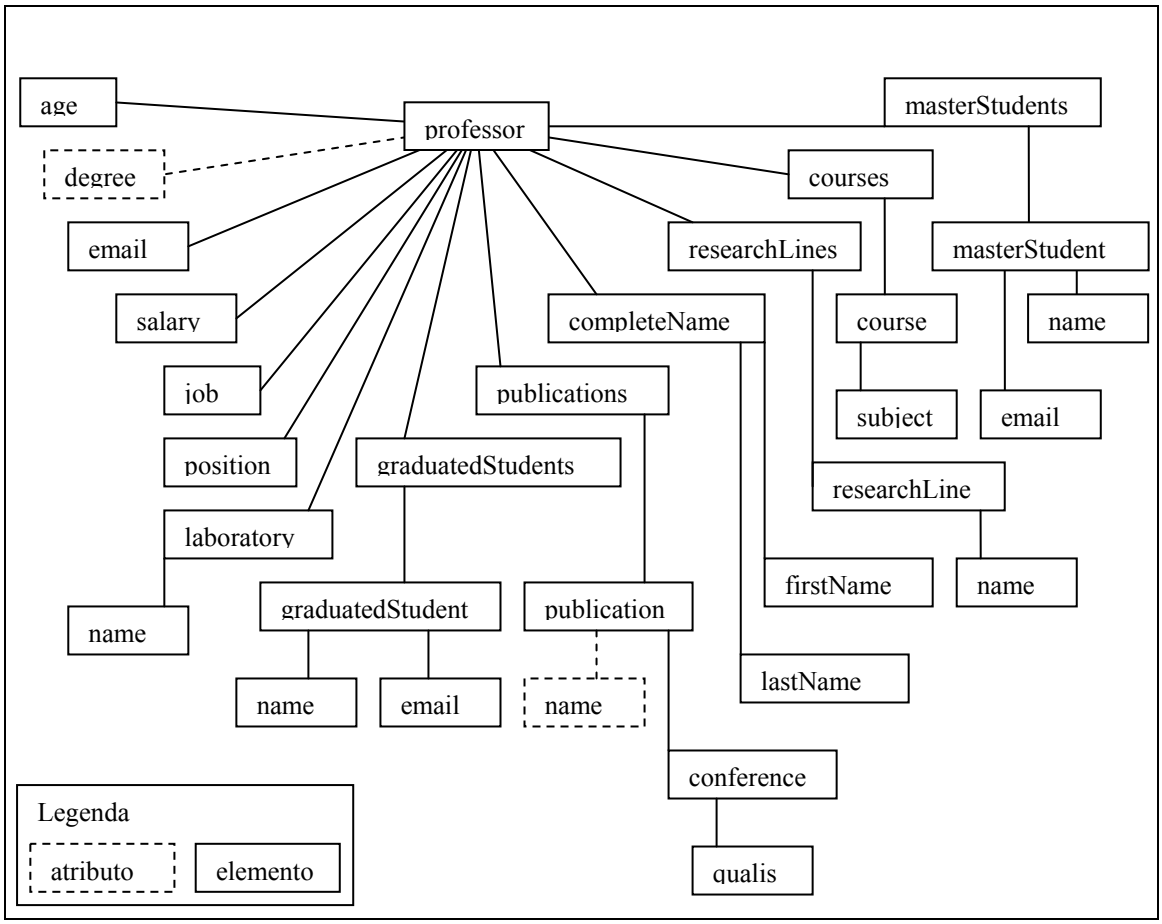
Tabela 6.1 – Premissas adotadas na construção do domínio de dados.

P1	Professores apenas com titulação de graduado não orientam alunos de mestrado e não orientam alunos de graduação.
P2	Professores apenas com titulação de mestre orientam poucos alunos de graduação e não orientam alunos de mestrado.
P3	Professores apenas com titulação de doutor orientam vários alunos de graduação e vários alunos de mestrado.
P4	Professores apenas com titulação de graduado possuem poucas publicações ou nenhuma.
P5	Professores com titulação de mestre possuem ao menos uma publicação.
P6	Professores com titulação de doutor possuem várias publicações.
P7	O número de publicações é diretamente proporcional ao número de alunos de mestrado que um professor orienta.
P8	O salário de um professor é diretamente proporcional à sua titulação.
P9	Professores com uma posição de coordenador têm maior salário.
P10	A quantidade de disciplinas que um professor leciona é inversamente proporcional à titulação do mesmo.
P11	Apenas professores com titulação de doutor possuem um laboratório de pesquisa.
P12	Ums disciplina só pode ser lecionada por um professor.
P13	Uma professor deve possuir duas linhas de pesquisa.
P14	Vários professores podem trabalhar em um mesmo laboratório.
P15	Um professor deve lecionar ao menos uma disciplina.

Fonte: Primária.

A Figura 6.1 apresenta uma representação gráfica da estrutura completa das instâncias XML, enquanto a Figura 6.2 exemplifica uma instância XML. O Anexo 1 apresenta uma instância XML completa, ou seja, composta por todos os elementos e

atributos apresentados na Figura 6.1, pretencentes ao domínio de dados. A Tabela 6.2 apresenta os valores dos parâmetros de entrada adotados no experimento em questão. Cada parâmetro é discutido dentro do seu contexto neste capítulo.



Fonte: Primária.

Figura 6.1 – Estrutura completa do domínio de instâncias XML.

A DIInCX, conforme discutido previamente, é composta por três fases. A primeira fase é responsável por construir uma instância do esquema conceitual hierárquico CDX-Tree através de um conjunto de regras propostas. As classes da CDX-Tree são apresentadas no Anexo 2, enquanto a instância completa CDX-Tree resultante do experimento apresentado neste capítulo é apresentada no Anexo 3. As RIS's descobertas agregadas, à instância CDX-Tree apresentada no Anexo 3, são apresentadas em separado no Anexo 4.

Tabela 6.2 - Valores dos parâmetros de entrada utilizados no experimento.

Parâmetro	**Valor**
Relevância mínima	60%
Percentual de discretização	80%
Percentual de dispersão máximo	70%
Complexidade máxima do componente restringido	1
Complexidade máxima do componente condição restritiva	1
Quantidade máxima de RIS descobertas	100
Quantidade máxima de itens para enumeração	3
Quantidade mínima de conceitos para consideração de subárvores	3

Fonte: Primária.

```
<professor degree =”master” >
   <age>60</age>
   <email>inacio.doval@gmail.com</email>
   <completeName>
      <firstName>Inacio</firstName>
      <lastName>Dorvantil</lastName>
    </completeName>
    <courses>
      <course>
         <subject>Introduction to Database Systems</subject>
      </course>
   </courses>
   <researchLines>
      <researchLine>
         <name>Database</name>
      </researchLine>
   <researchLines>
   <graduatedStudents>
      <graduatedStudent>
         <name>Iolanda Rezende</name>
         <e-mail>iolanda@gmail.com</e-mail>
      </graduatedStudent>
   </graduatedStudents>
   <publications>
      <publication name=”CRIS-XML”>
         <conference>DEXA
            <qualis>B</qualis>
         </conference>
      </publication>
   </publications>
   <job>Professor</job>
   <salary>3000,00</salary>
</professor>
```

Fonte: Primária.

Figura 6.2 – Exemplo de instância XML.

A fase de pré-processamento consiste na leitura das instâncias XML e aplicação de um conjunto de regras com a finalidade de construir o esquema conceitual

hierárquico CDX-Tree. As regras de pré-processamento são divididas em duas etapas: Construção de conceitos e Construção de conteúdo. A etapa de Construção de conceitos é responsável por traduzir os elementos e atributos presentes nas instâncias XML para instâncias das classes de metadados da CDX-Tree.

A regra 1 (Criação do elemento raiz) consiste na criação do elemento raiz da estrutura hierárquica da CDX-Tree. No caso do domínio exemplificado, o elemento raiz *professor* (Figura 6.2) deve ser traduzido para uma instância da classe *rootTree* (Figura 6.3-a). Para o conceito *professor* criado, deve ser adicionada a propriedade *instancesAmount*, representando a quantidade de instâncias XML tomadas como dados de entrada (no caso do experimento, são 40 instâncias).

(a) Instância da classe rootTree

```
Individual(b:professor
type(b:rootTree)
  value(b:hasChild b:degree)
  value(b:hasChild b:age)
  value(b:hasChild b:email)
  value(b:hasChild b:salary)
  value(b:hasChild b:job)
  value(b:hasChild b:position)
  value(b:hasChild b:laboratory)
  value(b:hasChild b:graduatedStudents)
  value(b:hasChild b:publications)
  value(b:hasChild b:completeName)
  value(b:hasChild b:researchLines)
  value(b:hasChild b:courses)
  value(b:hasChild b:masterStudents)
  value(b:instancesAmount "40"^^xsd:int))
```

(b) Instância da classe leafNode

```
Individual(b:degree
  type(b:leafNode)
  value(b:hasParent b:professor)
  value(b:relevance "40"^^xsd:int)
  value(b:hasQualifiedValue "master"^^xsd:string)
  value(b:hasQualifiedValue "graduated"^^xsd:string)
  value(b:hasQualifiedValue "doctor"^^xsd:string)
  value(b:originalPath "/professor@degree"^^xsd:string))
```

(c) Instância da classe collectonNode.

```
Individual(b:masterStudents
  type(b:collectionNode)
  value(b:hasParent b:professor)
  value(b:hasChild b:masterStudent.name)
  value(b:relevance "30"^^xsd:int)
  value(b:hasQuantifiedValue "0"^^xsd:double)
  value(b:hasQuantifiedValue "3"^^xsd:double)
  value(b:hasQuantifiedValue "6"^^xsd:double)
  value(b:originalPath "/professor/masterStudents"^^xsd:string))
```

Fonte: Primária

Figura 6.3 - Trecho da instância CDX-Tree.

A regra 2 (Atributos para conceitos *leafNode*) consiste na tradução de todos os atributos presentes nas instâncias XML, como o atributo *degree* (Figura 6.2), para instâncias da classe *leafNode* como o conceito *degree* demonstrado na Figura 6.3(b). Vale observar que embora o conceito *degree* demonstrado na Figura 6.3(b) esteja completo, no momento de sua criação, ele possui apenas as propriedades estruturais: *hasParent*, *relevance e originalPath*. As demais propriedades de conteúdo são adicionadas na etapa de Construção de conteúdo. Quanto à propriedade *hasParent*, deve ser atribuído o elemento imediatamente acima na árvore. No caso do conceito *degree,* o conceito imediatamente acima é o próprio conceito *professor*. À medida que novos conceitos vão sendo criados através das propriedades *hasChild* e *hasParent* a estrutura hierárquica vai sendo construída.

Na regra 3 (Elementos simples para conceitos leafNode), elementos simples como *age, email, job, salary* e *position* (Figura 6.1) são traduzidos para os conceitos *leafNode* de mesmo nome. Elementos simples que estiverem contidos em elementos complexos são tratados a seguir. A Tabela 6.3 apresenta todos os elementos tomados como entrada e os respectivos iconceitos da instância CDX-Tree para as quais foram traduzidos. A Tabela 6.3 também demonstra a regra onde cada elemento foi criado.

Tabela 6.3 – Tradução de elementos/atributos para conceitos da CDX-Tree.

Regra	**Objeto XML**	**Tipo de objeto XML**	**Conceito traduzido**	**Tipo do conceito traduzido**
1	/professor	Elemento complexo	professor	rootTree
2	/professor@Degree	Atributo	degree	Qualified LeafNode
3	/professor/Age	Elemento simples	age	Quantified LeafNode
3	/professor/email	Elemento simples	email	Dispersed LeafNode
3	/professor/Salary	Elemento simples	salary	Quantified LeafNode
3	/professor/job	Elemento simples	Job	Constant LeafNode
	/professor/completeName	Elemento complexo	-	-
5	/professor/completeName/firstName	Elemento simples	completeName.firstName	Dispersed LeafNode
5	/professor/completeName/lastName	Elemento Simples	completeName.lastName	Dispersed LeafNode
3	/professor/position	Elemento simples	position	Qualified LeafNode
	/professor/laboratory	Elemento complexo	-	-
5	/professor/laboratory/name	Elemento simples	laboratory.name	Qualified LeafNode
4	/professor/courses	Elemento complexo	courses	Quantified CollectionNode
	/professor/courses/course	Elemento complexo	-	-
5	/professor/courses/course/subject	Elemento simples	course.subject	Dispersed LeafNode
4	/professor/researchLines	Elemento complexo	researchLines	Constant CollectionNode
	/professor/researchLines/ researchLine	Elemento complexo	-	-
5	/professor/researchLines/researchLine/name	Elemento simples	researchLine.name	Qualified LeafNode
4	/professor/masterStudents	Elemento complexo	masterStudents	Quantified CollectionNode
	/professor/masterStudents/masterStudent	Elemento complexo	-	-
	/professor/masterStudents/masterStudent/name	Elemento simples	masterStudent.name	Dispersed LeafNode
	/professor/masterStudents/masterStudent/email	Elemento simples	masterStudent.email	Dispersed LeafNode
4	/professor/graduatedStudents	Elemento complexo	graduatedStudents	Quantified CollectionNode
	/professor/graduatedStudents/graduatedStudent	Elemento complexo	-	-
5	/professor/graduatedStudents/graduatedStudent /name	Elemento simples	graduatedStudent.name	Dispersed LeafNode
5	/professor/graduatedStudents/graduatedStudent /email	Elemento simples	graduatedStudent.email	Dispersed LeafNode
4	/professor/publications	Elemento complexo	publications	Quantified CollectionNode
	/professor/publications/publication	Elemento complexo	-	-
2	/professor/publications/publication@name	Atributo	publication.name	Dispersed LeafNode
5	/professor/publications/publication/conference	Elemento Misto	publication.conference	Qualified LeafNode
5	/professor/publications/publication/conference/ qualis	Elemento Simples	publication.conference.qualis	Qualified LeafNode

A regra 4 (Elementos que compõem coleções para conceitos *collectionNode*) compreende a tradução de elementos complexos XML compostos exclusivamente por elementos de um mesmo tipo, em quantidade maior que 1. Este tipo de elemento XML por si só não possui informação relevante, mas serve de contêiner para seus elementos

filhos. Exemplos de elementos deste tipo são: *masterStudents*, *graduatedStudents*, *courses*, *publications* e *researchLines* (Figura 6.1 e Figura 6.2). Estes tipos de elementos são traduzidos para instâncias da classe *collectionNode,* conforme exemplificado na Figura 6.3(c) através da tradução do elemento XML *masterStudents* para o conceito de mesmo nome.

A regra 5 (Elementos complexos para conceitos *leafNode*), compreende a tradução de elementos complexos XML para instâncias da classe *leafNode*. Neste caso elementos complexos como *completeName, laboratory, course, researchLine, masterStudent, graduatedStudent* e *publication* (Figura 6.1 e Figura 6.2), são incorporados a seus elementos filhos originando um novo conceito para cada elemento simples filho que ele possuir. O elemento *completeName,* por exemplo, possui dois elementos filhos. Desta forma, o resultado de sua tradução são os conceitos *completeName.firstName* e *completeName.lastName*, resultantes da incorporação do nome do elemento complexo como prefixo para os elementos filhos *firstName* e *lastName* (Tabela 6.3). Os elementos mistos XML, compostos tanto por elementos simples quanto por texto, são tratados nesta regra. No caso do elemento *conference*, por exemplo, contido no elemento complexo *publication*, por possuir valor próprio, é traduzido diretamente para uma instância da classe *leafNode* (*publication.conference* Tabela 6.3), além de incorporado aos seus elementos filhos como o conceito resultante *publication.conference.qualis*, demonstrado na Tabela 6.3

A partir da regra 5, a etapa de construção das propriedades estruturais e conseqüentemente a estrutura hierárquica da instância do esquema conceitual CDX-Tree está completa. As regras 6 e 7 a seguir compreendem a etapa de Construção do conteúdo da fase de Pré-processamento. Esta etapa foca na atribuição de sub-propriedades da propriedade *hasValue* (Figura 5.4), a fim de delimitar o tipo de dado que cada conceito representa (*qualificado, quantificado, disperso, constante*). Nesta etapa conceitos quantificados também têm seus valores contínuos transformados em valores discretos, a fim de tornar a descoberta de RIS mais direta e menos onerosa. Para fins de exemplificação, no restante deste capítulo é tomada como base a tabela 6.3.

A regra 6 (Criação do domínio de elementos qualificados) compreende a atribuição das sub-propriedades *hasConstantValue*, *hasDispersedValue* e

hasQualifiedValue a conceitos classificados como constantes, dispersos e qualificados, respectivamente. O conceito *job* é um exemplo de conceito constante, instância da classe *leafNode*. O conceito *job* é classificado como constante, pois todo professor de uma universidade no dado domínio possui o mesmo emprego: "professor". Quanto aos conceitos dispersos, uma vez adotado um percentual de dispersão máximo de 70% os seguintes conceitos foram classificados como dispersos: *email*, *completeName.firstName*, *completeName.lastName*, *course.subject*, *masterStudent.name*, *masterStudent.email*, *graduatedStudent.name*, *graduatedStudent.email* e *publication.name.* Quanto aos conceitos qualificados, é construída, através da leitura das instâncias XML, uma lista de valores cabíveis a estes conceitos, sendo cada valor associado ao conceito através da propriedade *hasQualifiedValue*. No experimento, os seguintes conceitos foram classificados como qualificados: *degree*, *position*, *laboratory.name*, *researchLine.name*, *publication.conference*, *publication.conference.qualis*. A estes conceitos foram atribuídas tantas propriedades *hasQualifiedValue* quantos valores distintos pertencentes a estes[5]. Por exemplo, ao conceito *degree*, foram associados os valores *graduated*, *master* e *doctor*, conforme exemplificado na Figura 6.4(a).

(a) Conceito Qualified leafNode	**(b) Conceito Quantified leafNode**
Individual(b:degree type(b:leafNode) value(b:hasParent b:professor) value(b:relevance "40"^^xsd:int) value(b:hasQualifiedValue "master"^^xsd:string) value(b:hasQualifiedValue "graduated"^^xsd:string) value(b:hasQualifiedValue "doctor"^^xsd:string) value(b:originalPath "/professor@degree"^^xsd:string))	Individual(b:salary type(b:leafNode) value(b:hasParent b:professor) value(b:relevance "40"^^xsd:int) value(b:hasQuantifiedValue "1000"^^xsd:double) value(b:hasQuantifiedValue "3050"^^xsd:double) value(b:hasQuantifiedValue "5000"^^xsd:double) value(b:originalPath "/professor/salary"^^xsd:string))

Fonte: Primária

Figura 6.4 - Exemplos de conceitos leafNode.

A regra 7 (Criação do domínio de elementos quantificados), consiste na classificação de conceitos como quantificados ou constantes através da atribuição das propriedades *hasQuantifiedValue* ou *hasConstantValue* respectivamente. Os seguintes conceitos, instâncias da classe *leafNode*, foram classificados como quantificados: *age* e *salary*. Quanto a instâncias da classe *collectionNode*, estas são classificadas como

[5] A criação de conceitos qualificados com muitas propriedades *hasQualifiedValue* associadas é minimizada pela definição destes como conceitos dispersos através do parâmetro de entrada *Percentual de Dispersão Máximo*.

conceitos quantificados sempre que não forem classificadas como conceitos constantes. O conceito *researchLines* é um exemplo de conceito constante instância da classe *collectionNode*. Este conceito é constante, pois em todas as instâncias XML cada professor é associado a duas linhas de pesquisa (conforme determinado pela premissa P13 – Tabela 6.1). No mais, foram classificados como quantificados os seguintes conceitos: *courses*, *masterStudents*, *graduatedStudents* e *publications*. Quanto aos valores de conceitos quantificados, estes são associados aos conceitos através da propriedade *hasQuantifiedValue*. No entanto, os valores distintos presentes nas instâncias XML devem ser previamente discretizados, antes de serem associados aos conceitos. Na seqüência, para cada valor discretizado deve ser associada ao conceito, uma propriedade *hasQuantifiedValue* tal como as propriedades *hasQuantifiedValue* presentes na Figura 6.4(b), que associam ao elemento *salary* os valores *"1000"*, *"3050"* e *"5000"*.

O método de discretização adotado foi discutido no capítulo 5. A Tabela 6.4 detalha os conceitos quantificados presentes no experimento. Esta tabela apresenta os valores máximos, mínimos, quantidade de valores distintos pertencente ao domínio de cada conceito, além do conjunto de valores resultantes do processo de discretização. A Tabela 6.5 apresenta a *relevância* de cada conceito, ou seja, a quantidade de vezes que ele se faz presente dentre as instâncias XML que compõem o domínio.

Tabela 6.4 – Valores discretizados de conceitos Quantificados.

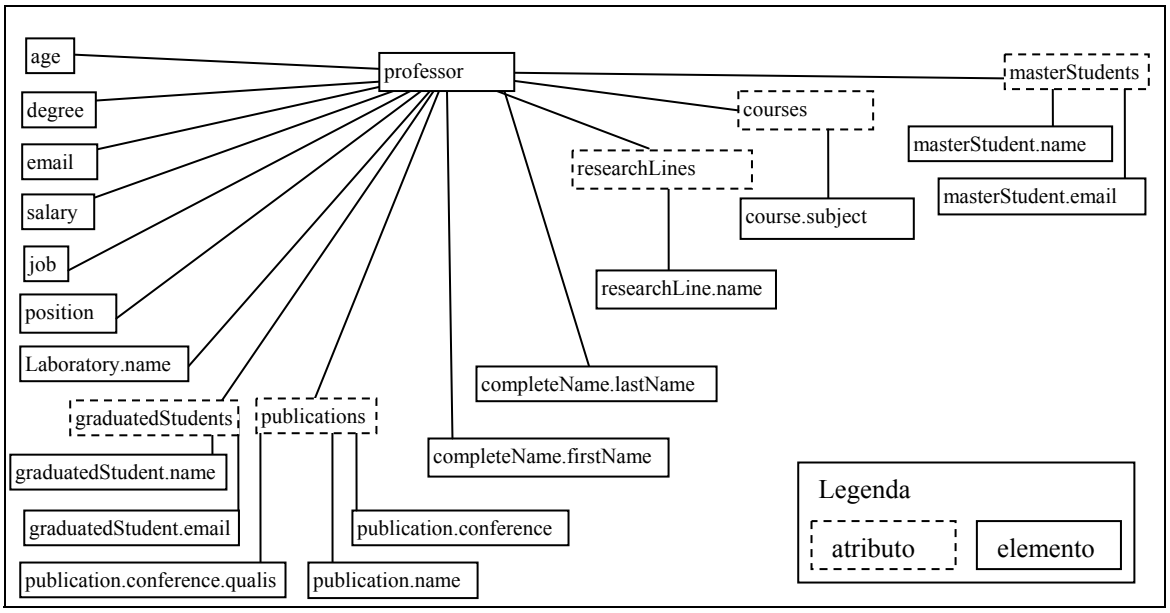
Conceitos	**Mínimo**	**Máximo**	**Quantidade de valores distintos**	**Conjunto de valores Discretizados**
Age	30	60	30	{30, 36, 42, 48, 54, 60}
Salary	1000,00	5000,00	15	{1000,3050, 5000}
Courses	1	6	6	{1,3,6}
masterStudents	0	6	7	{0,3,6}
graduatedStudents	0	10	11	{0,5,10}
Publications	0	20	21	{0,6,14,20}

Após a regra 7, a fase de Pré-processamento está completa, sendo iniciada a fase de Descoberta. A representação gráfica da estrutura da instância CDX-Tree completa resultante do final da fase de pré-processamento é apresentada na Figura 6.5. Vale observar que todas as regras de pré-processamento propostas foram aplicadas através da leitura das instâncias XML e posterior representação na instância CDX-Tree, ou seja, conforme previamente comentado, não há operações de atualização sobre as instâncias XML. A fase de Descoberta tem por objetivo a descoberta de algumas RIS de menor

complexidade e RA's que são traduzidas na próxima fase para RIS's. Ela é dividida em duas etapas: *Descoberta de RIS simples* e *Mineração de RIS*.

Tabela 6.5 – Relevância dos conceitos presentes na CDX-Tree.

Conceito	**Relevância Qtd - %**	**Conceito**	**Relevância Qtd - %**
		researchLines	40 - 100%
Degree	40 - 100%	researchLine.name	40 - 100%
Age	40 - 100%	masterStudents	15 – 37,5%
Email	20 - 50%	masterStudent.name	15 – 37,5%
Salary	40 - 100%	masterStudent.email	11 – 27,5%
Job	40 - 100%	graduatedStudents	29 – 72,5%
completeName.firstName	40 - 100%	graduatedStudent.name	29 – 72,5%
completeName.lastName	40 - 100%	graduatedStudent.email	25 – 62,5%
Position	4 - 10%	Publications	17 – 42,5%
laboratory.name	23 - 57,5%	publication.name	17 – 42,5%
Courses	40 - 100%	publication.conference	17 – 42,5%
course.subject	40 - 100%	publication.conference.qualis	9 – 22,5%



Fonte: Primária.

Figura 6.5 – Representação gráfica da estrutura hierárquica da CDX-Tree.

A etapa de Descoberta de RIS simples, através da análise da estrutura e dos dados presentes na instância CDX-Tree, objetiva descobrir RIS relacionadas a: *(i)* Elementos ou atributos obrigatórios; *(ii)* Elementos ou atributos restringidos a conjuntos de valores; *(iii)* Elementos ou atributos constantes; e *(iv)* Cardinalidade de coleções de elementos.

Com relação ao item *(i)*, através da propriedade *relevance* presente em cada conceito (Tabela 6.5) e da propriedade *instancesAmount,* presente no conceito raiz da CDX-Tree, foram identificados como obrigatórios no contexto do domínio de dados os

seguintes elementos: *professor*, *degree*, *age*, *salary*, *job*, *completeName.firstName*, *completeName.lastName*, *courses* e *researchLines*. Entretanto, os conceitos filhos de conceitos obrigatórios não devem ser considerados como obrigatórios no contexto do domínio de dados, pois seu caráter obrigatório é tratado como relativo ao conceito pai. Por conseguinte, foram identificados como conceitos com obrigatoriedade relativa: *course.subject*, *researchLine.name*, *masterStudent.name*, *graduatedStudent.name*, *publication.name* e *publication.conference*. As RIS descobertas relacionadas à obrigatoriedade são apresentadas na Figura 6.6.

(a) RIS de obrigatoriedade absoluta	**(b) RIS de obrigatoriedade relativa**
RIS1: → leafNode(degree) RIS2: → leafNode(age) RIS3: → leafNode(salary) RIS4: → leafNode(job) RIS5: → leafNode(completeName.firstName) RIS6: → leafNode(completeName.lastName) RIS7: → collectionNode(courses) RIS8: → collectionNode(researchLines)	RIS9: collectionNode(researchLines) → leafNode(researchLine.name) RIS10: collectionNode(courses) → leafNode(course.subject) RIS11: collectionNode(masterStudents) → leafNode(masterStudent.name) RIS12: collectionNode(graduatedStudents) → leafNode(graduatedStudent.name) RIS13: collectionNode(publications) → leafNode(publication.name) RIS14: collectionNode(publications) → leafNode(publication.conference)

Fonte: Primária.

Figura 6.6 – RIS descobertas relacionadas à obrigatoriedade.

Com relação ao item *(ii),* conceitos são restringidos a conjuntos enumerados caso a quantidade de propriedades *hasQualifiedValue* presentes em conceitos qualificados seja menor ou igual ao parâmetro de entrada *Quantidade máxima de itens para enumeração*. Uma vez adotado o valor 3 no experimento para o referido parâmetro de entrada, foram restringidos a enumerações os conceitos: *degree*, *publication.conference.qualis* e *position*. A Figura 6.7 apresenta as RIS's RIS1, RIS2 e RIS3 definidas para os conceitos *degree*, *publication.conference.qualis* e *position,* respectivamente.

RIS1: hasQualifiedValue(degree,?degreeValue)^
swrlb:listConcat(?degreeValueSet,"master","doctor","graduated") →
swrlb:member(?degreeValue,?degreeValueSet)

RIS2: hasQualifiedValue(publication.conference.qualis,?publicationConferenceQualisValue)^
swrlb:listConcat(?publicationConferenceQualisValueSet,"A","B","C") →
swrlb:member(?publicationConferenceQualisValue,?publicationConferenceQualisValueSet)

RIS3: hasQualifiedValue(position,?positionValue)^

swrlb:listConcat(?positionValueSet,”Coordinator”,”Director”) →
swrlb:member(?positionValue,?positionValueSet)

Fonte: Primária.

Figura 6.7 – RIS descobertas relacionadas a conceitos enumerados.

Com relação ao item *(iii)*, foram os seguintes conceitos constantes identificados: *job* e *researchLines*. As RIS definidas a partir destes conceitos são apresentadas na Figura 6.8. As RIS’s descobertas no experimento, relacionadas ao item *(iv)* são apresentadas na Figura 6.9. RIS’s relacionadas à cardinalidade máxima e mínima são descobertas a partir das propriedades *hasQuantifiedValue* com máximo e mínimo valor, presentes em conceitos quantificados como: *age*, *salary*, *courses*, *masterStudents*, *graduatedStudents* e *publications*.

Todas as RIS descobertas nesta etapa (Figura 6.6, Figura 6.7, Figura 6.8 e Figura 6.9) são acrescentadas à instância do esquema conceitual CDX-Tree, compondo assim o primeiro conjunto de RIS descobertas através da abordagem DIInCX.

A etapa de Mineração de RIS, inicia pela leitura dos dados presentes nas instâncias XML, a fim de compor a *matriz de transações*. A leitura dos dados, se dá através da navegação pela estrutura hierárquica da instância CDX-Tree, acessando os elementos e atributos das instâncias XML com o auxílio da propriedade *originalPath* presente nos conceitos da instância CDX-Tree. Entretanto, na composição apenas conceitos julgados relevantes são incorporados à da matriz de transações. A Tabela 6.6 apresenta um exemplo parcial da matriz de transações com alguns dos conceitos julgados relevantes para fins de exemplificação.

RIS1: leafNode(job)→ hasConstantValue(job,”professor”)

RIS2: collectionNode(researchLines)→ hasConstantValue(researchLines,2)

Fonte: Primária.

Figura 6.8 – RIS descobertas relacionadas a conceitos constantes.

RIS1: hasQuantifiedValue(age,?ageValue) →
swrlb:lessThanOrEqual(?ageValue,60)^swrlb:greaterThanOrEqual(?ageValue,30)

RIS2: hasQuantifiedValue(salary,?salaryValue) →
swrlb:lessThanOrEqual(?salaryValue,5000)^swrlb:greaterThanOrEqual(?salaryValue,1000)

RIS3: hasQuantifiedValue(courses,?coursesAmount) →
swrlb:lessThanOrEqual(?coursesAmount,6)^swrlb:greaterThanOrEqual(?coursesAmount,0)

RIS4: hasQuantifiedValue(masterStudents,?masterStudentsAmount) →
swrlb:lessThanOrEqual(?masterStudentsAmount,6)^swrlb:greaterThanOrEqual(?masterStudentsAmount,0)

RIS5: hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) →
swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^swrlb:greaterThanOrEqual(?graduatedStudentsAmount,0)

RIS6: hasQuantifiedValue(publications,?publicationsAmount) →
swrlb:lessThanOrEqual(?publicationsAmount,20)^swrlb:greaterThanOrEqual(?publicationsAmount,0)

Fonte: Primária.

Figura 6.9 – RIS descobertas relacionadas à cardinalidade.

Dentre os conceitos da instância CDX-Tree que não foram considerados relevantes na composição da matriz de transações, estão os conceitos classificados como constantes, dispersos e pertencentes a subárvores. Conforme discutido previamente, subárvores serão consideradas em matrizes de transações separadas. Contudo, apenas subárvores relevantes para mineração de RA’s são consideradas, com base no parâmetro de entrada *Quantidade mínima de conceitos para consideração de subárvores*.

Assim, a partir da comparação dos valores cabíveis de conceitos qualificados (propriedade *hasQualifiedValue*) e intervalos de valores de conceitos quantificados (propriedade *hasQuantifiedValue*) com as instâncias XML tomadas como dados de entrada, é montada a matriz de transações que é utilizada como dado de entrada para o algoritmo de mineração de RA’s. Desta forma, cada transação t_n pertencente ao conjunto T da matriz de transações representa uma instância XML.

No experimento em questão, conforme demonstrado na Tabela 6.2, é adotado o valor 3 para o parâmetro *Quantidade mínima de conceitos para consideração de subárvores*. Por conseguinte, são identificadas as seguintes subárvores com conceitos relevantes: *researchLines*, composto pelo conceito *researchLine.name,* e *publications* composto pelos conceitos *publication.conference* e *publication.conference.qualis.*

Entretanto, estas subárvores são desconsideradas, pois sua cardinalidade de conceitos relevantes (1 e 2 respectivamente) é inferior ao parâmetro anteriormente comentado.

Tabela 6.6 - Matriz de transações parcial.

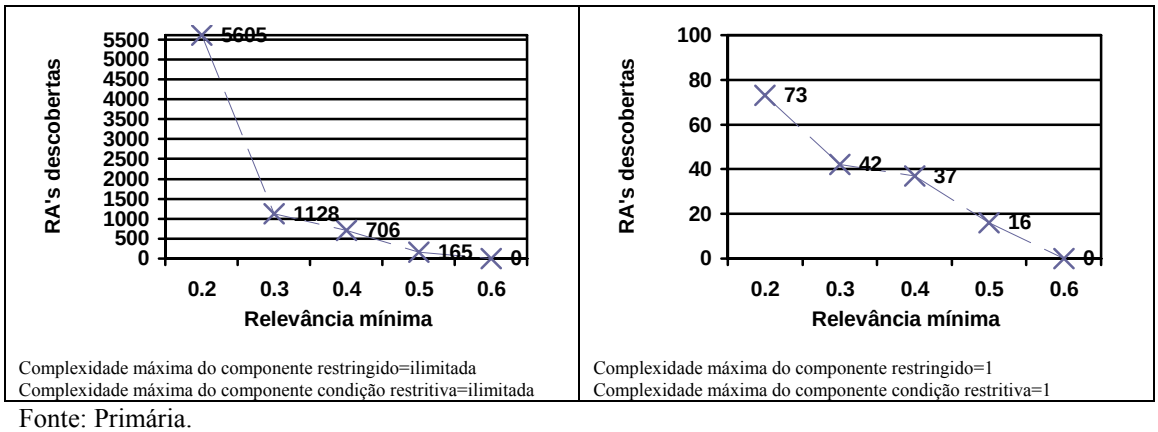
	degree	age	Salary	position	Courses	masterStudents	Publications
t1	graduated	">=30To<36"	“>=1000To<3050”	none	">=3To<=6"	">=0To<3"	">=0To<6"
t2	graduated	">=30To<36"	">=1000To<3050"	none	">=3To<=6"	">=0To<3"	">=0To<6"
t3	master	">=30To<36"	">=1000To<3050"	none	">=3To<=6"	">=0To<3"	">=0To<6"
t4	master	">=30To<36"	">=1000To<3050"	none	">=3To<=6"	">=0To<3"	">=0To<6"
t5	máster	">=30To<36"	">=1000To<3050"	none	">=3To<=6"	">=0To<3"	">=0To<6"
t6	master	">=42To<48"	">=3050To<=5000"	none	">=3To<=6"	">=3To<=6"	">=14To<=20"
t7	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t8	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t9	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t10	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t11	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t12	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t13	doctor	">=42To<48"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t14	doctor	">=48To<54"	">=3050To<=5000"	none	">=1To<3"	">=3To<=6"	">=14To<=20"
t15	doctor	">=54To<=60"	">=3050To<=5000"	coordinator	">=1To<3"	">=3To<=6"	">=14To<=20"
t16	doctor	">=54To<=60"	">=3050To<=5000"	director	">=1To<3"	">=3To<=6"	">=14To<=20"

Fonte: Primária.

Quanto aos parâmetros para o algoritmo de mineração de RA’s, experimentos demonstraram que a determinação do valor ideal para os parâmetros de entrada *Relevância Mínima (RM)*, *Complexidade Máxima do componente Restringido (CMR)* e *Complexidade Máxima do componente Condição Restritiva (CMCR)* é dependente do domínio de dados e sua distribuição entre conceitos quantificados, qualificados, bem como a variedade de valores cabíveis a estes. Com relação à distribuição dos dados, no domínio adotado, a matriz de transações se mostrou composta por 50% de valores qualificados e quantificados, estando igualmente distribuídos. Assim, embora seja proposto o parâmetro *Quantidade máxima de RIS descobertas* (parâmetro de parada comum a algoritmos de mineração de RA’s) como forma de automatizar o processo,

observa-se, que a qualidade do resultado obtido para a abordagem é dependente de experimentação.

A fim de determinar os valores ideiais para os parâmetros comentados, experimentos são iniciados com valores altos de RM e valores baixos de CMR e CMCR. Valores altos de RM originam RIS's de maior relevância, cuja implicação ocorreu com maior frequência no domínio, enquanto valores baixos de CMR e CMCR originam RIS's de menor complexidade. Caso o usuário julgue necessário descobrir uma quantidade maior de RIS's, deve incrementar os valores de CMR e CMCR e decrementar o valor do parâmetro RM. A Figura 6.10 apresenta um quadro comparativo entre o valor fornecido de RM e a quantidade de RIS's descobertas. A Figura 6.10(a) demonstra a quantidade de RA's descobertas com complexidade ilimitada. Enquanto a Figura 6.10(b) apresenta a quantidade de RA's descobertas com complexidade igual a 1. A partir destes experimentos, por intervenção do usuário especialista, adota-se que a quantidade de 16 regras descobertas para o valor 50% para o parâmetro RM e 1 para ambos os parâmetros CMR e CMCR seria adequada. A Figura 6.11 apresenta o conjunto de RA's descobertas como produto final desta etapa que encerra a fase de descoberta.



Complexidade máxima do componente restringido=ilimitada
Complexidade máxima do componente condição restritiva=ilimitada

Complexidade máxima do componente restringido=1
Complexidade máxima do componente condição restritiva=1

Fonte: Primária.

Figura 6.10 – Quantidade de RA's descobertas por Relevância mínima.

A fase de Conversão é responsável por traduzir as RA's descobertas para RIS's especificadas em SWRL. A Figura 6.12 apresenta as RIS's resultantes da tradução das RA's apresentadas na Figura 6.11. Ao final desta fase o usuário especialista é chamado a avaliar as RIS's descobertas, desconsiderando as que não julgar relevantes ou criando novas caso julgue necessário. Por conseguinte, as RIS's descobertas são agregadas à

instância do esquema conceitual hierárquico CDX-Tree encerrando a fase de Conversão e por conseqüência, a abordagem DIInCX.

RA1: courses=">=1To<3" → degree="doctor"
RA2: courses=">=1To<3" → graduatedStudents=">=5To<=10"
RA3: courses=">=1To<3" → masterStudents=">=3To<=6"
RA4: courses=">=1To<3" → salary=">=3050To<=5000"
RA5: degree="doctor" → courses=">=1To<3"
RA6: degree="doctor" → graduatedStudents=">=5To<=10"
RA7: degree="doctor" → masterStudents=">=3To<=6"
RA8: degree="doctor" → salary=">=3050To<=5000"
RA9: graduatedStudents=">=5To<=10" → courses=">=1To<3"
RA10: graduatedStudents=">=5To<=10" → degree="doctor"
RA11: graduatedStudents=">=5To<=10" → masterStudents=">=3To<=6"
RA12: graduatedStudents=">=5To<=10" → salary=">=3050To<=5000"
RA13: salary=">=3050To<=5000" → courses=">=1To<3"
RA14: salary=">=3050To<=5000" → degree="doctor"
RA15: salary=">=3050To<=5000" → graduatedStudents=">=5To<=10"
RA16: salary=">=3050To<=5000" → masterStudents=">=3To<=6"

Fonte: Primária.

Figura 6.11 – RA's descobertas com Relevância Mínima 50%.

Pretende-se desta forma que o esquema conceitual CDX-Tree, sua respectiva instância e as RIS's XML agregadas, componham uma ferramenta útil para sistemas de informação que pretendam dar suporte a RIS's. O Anexo 3 apresenta a instância do esquema conceitual CDX-Tree completa, enquanto o Anexo 4 apresenta as RIS's XML descobertas e agregadas à instância CDX-Tree.

RIS1: hasQuantifiedValue(courses,?coursesAmount) ^ swrlb:lessThan(?coursesAmount,3)
^ swrlb:greaterThanOrEqual(?coursesAmount,1) → hasQualifiedValue(degree,”doctor”)

RIS2: hasQuantifiedValue(courses,?coursesAmount) ^ swrlb:lessThan(?coursesAmount,3) ^ swrlb:greaterThanOrEqual(?coursesAmount,1)
^ hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) → swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5)

RIS3: hasQuantifiedValue(courses,?coursesAmount) ^ swrlb:lessThan(?coursesAmount,3) ^ swrlb:greaterThanOrEqual(?coursesAmount,1)
^ hasQuantifiedValue(masterStudents,?masterStudentsAmount) → swrlb:lessThanOrEqual(?masterStudentsAmount,6)
^ swrlb:greaterThanOrEqual(?masterStudentsAmount,3)

RIS4: hasQuantifiedValue(courses,?coursesAmount) ^ swrlb:lessThan(?coursesAmount,3) ^ swrlb:greaterThanOrEqual(?coursesAmount,1)
^ hasQuantifiedValue(salary,?salaryValue) → swrlb:lessThanOrEqual(?salaryValue,5000) ^ swrlb:greaterThanOrEqual(?salaryValue,3050)

RIS5: hasQualifiedValue(degree,”doctor”) ^ hasQuantifiedValue(courses,?coursesAmount) → swrlb:lessThan(?coursesAmount,3)
^ swrlb:greaterThanOrEqual(?coursesAmount,1)

RIS6: hasQualifiedValue(degree,”doctor”) ^ hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) →
swrlb:lessThanOrEqual(?graduatedStudentsAmount,10) ^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5)

RIS7: hasQualifiedValue(degree,”doctor”) ^ hasQuantifiedValue(masterStudents,?masterStudentsAmount) →
swrlb:lessThanOrEqual(?masterStudentsAmount,6) ^ swrlb:greaterThanOrEqual(?masterStudentsAmount,3)

RIS8: hasQualifiedValue(degree,”doctor”) ^ hasQuantifiedValue(salary,?salaryValue) → swrlb:lessThanOrEqual(?salaryValue,5000)
^ swrlb:greaterThanOrEqual(?salaryValue,3050)

RIS9: hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) ^ swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5) ^ hasQuantifiedValue(courses,?coursesAmount) → swrlb:lessThan(?coursesAmount,3)
^ swrlb:greaterThanOrEqual(?coursesAmount,1)

RIS10: hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) ^ swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5) → hasQualifiedValue(degree,”doctor”)

RIS11: hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) ^ swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5) ^ hasQuantifiedValue(masterStudents,?masterStudentsAmount) →
swrlb:lessThanOrEqual(?masterStudentsAmount,6) ^ swrlb:greaterThanOrEqual(?masterStudentsAmount,3)

RIS12: hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) ^ swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5) ^ hasQuantifiedValue(salary,?salaryValue) →
swrlb:lessThanOrEqual(?salaryValue,5000) ^ swrlb:greaterThanOrEqual(?salaryValue,3050)

RIS13: hasQuantifiedValue(salary,?salaryValue) ^ swrlb:lessThanOrEqual(?salaryValue,5000) ^ swrlb:greaterThanOrEqual(?salaryValue,3050)
^ hasQuantifiedValue(courses,?coursesAmount) → swrlb:lessThan(?coursesAmount,3) ^ swrlb:greaterThanOrEqual(?coursesAmount,1)

RIS14: hasQuantifiedValue(salary,?salaryValue) ^ swrlb:lessThanOrEqual(?salaryValue,5000) ^ swrlb:greaterThanOrEqual(?salaryValue,3050)
→ hasQualifiedValue(degree,”doctor”)

RIS15: hasQuantifiedValue(salary,?salaryValue) ^ swrlb:lessThanOrEqual(?salaryValue,5000) ^ swrlb:greaterThanOrEqual(?salaryValue,3050)
^ hasQuantifiedValue(graduatedStudents,?graduatedStudentsAmount) → swrlb:lessThanOrEqual(?graduatedStudentsAmount,10)
^ swrlb:greaterThanOrEqual(?graduatedStudentsAmount,5)

RIS16: hasQuantifiedValue(salary,?salaryValue) ^ swrlb:lessThanOrEqual(?salaryValue,5000) ^ swrlb:greaterThanOrEqual(?salaryValue,3050)
^ hasQuantifiedValue(masterStudents,?masterStudentsAmount) → swrlb:lessThanOrEqual(?masterStudentsAmount,6)
^ swrlb:greaterThanOrEqual(?masterStudentsAmount,3)

Fonte: Primária.

Figura 6.12 – RIS’s descobertas com Relevância Mínima 50%.

7 CONCLUSÃO

Este trabalho apresenta uma abordagem semi-automática para descoberta de RIS's XML implícitas a partir de instâncias XML chamada *DIInCX*. A abordagem proposta tem por foco instâncias XML ao invés de esquemas XML, pois a representação de RIS's através de esquemas XML é usualmente incompleta, seja pela grande quantidade de RIS's que podem ter de ser especificadas para um dado domínio, ou limitações das linguagens de esquema (LEE & CHU, 2000; FAN, 2005).

A abordagem DIInCX é composta por um conjunto de regras de pré-processamento que tem por intenção uniformizar e melhorar o processo de descoberta, adaptações propostas a algoritmos de mineração de RA's e um conjunto de regras para tradução das RA's descobertas para RIS especificadas em SWRL. O processo de mineração de RIS's é guiado pelo esquema conceitual hierárquico chamado CDX-Tree, o qual representa a estrutura simplificada das instâncias XML. Embora em experimentos tenha sido empregado o algoritmo clássico *Apriori*, argumenta-se que a abordagem proposta é genérica o suficiente para ser aplicada com o auxílio de outros algoritmos de mineração de (HAN et al., 2000; BRAGA et al., 2002; GYORÖDI, 2003; BAYARDO, 2004) e/ou otimizações do algoritmo Apriori (AGRAWAL & SRIKANT, 1994; SRIKANT et al., 1997)

As adaptações propostas a algoritmos de mineração de RA tem por objetivo dar prioridade a RA's que podem ser traduzidas para RIS's de menor complexidade. Mesmo assim, RIS's de maior complexidade não são descartadas e podem ser descobertas conforme a necessidade do usuário especialista.

Este capítulo apresenta as conclusões sobre a abordagem DIInCX. A seção 7.1 destaca as principais contribuições. Na seqüência, a seção 7.2 destaca os trabalhos futuros mais significativos. Por último a seção 7.3. apresenta as considerações finais deste trabalho.

7.1 Contribuições

A principal contribuição deste trabalho é propor uma abordagem capaz de descobrir informação implícita a fim de complementar fontes de dados pré-existentes com a agregação de semântica na forma de RIS's. Desta forma, acredita-se estar contribuindo para a pesquisa sobre RIS's no contexto do modelo da dados XML e mais especificamente em sistemas como SII's e SGBD's XML. A incorporação de RIS nestes sistemas pode ser empregada na otimização de diversos processos, como: *(i)* uma maior consistência do modelo de dados XML; *(ii)* sistemas de consultas mais robustos; e *(iii)* maior precisão na integração de instâncias e esquemas quando os dados estiverem acompanhados de RIS's.

Dentre as demais contribuições da abordagem DIInCX destacam-se:

- **Uma abordagem para descoberta de RIS's implícitas com foco em regras de negócio e não apenas RI's simples como tipos de dados primitivos.** Com base nos trabalhos relacionados presentes na literatura para o modelo de dados XML, observa-se que a descoberta de RIS é abordada principalmente dentre aqueles que focam na descoberta de esquemas (NESTOROV et al., 1998; HACID et al., 2000; CASTANO et al., 2002; CHIDLOVSKII, 2002; LIU et al., 2004; HEGEWALD et al., 2006). No entanto, estes focam em RIS's de caráter simples, tais como, tipos de dados primitivos. Isto porque, ao representar o conhecimento descoberto em linguagens de definição de RIS's específicas (com baixo nível de abstração e/ou baixo poder de expressão), trabalhos que focam na descoberta de esquemas estão limitados a descobrir o conhecimento capaz de ser representado através de suas linguagens (LEE & CHU, 2000; DODDS, 2001; JACINTO et al., 2002a; JACINTO et al., 2002b; BUNEMAN, 2003; HU & TAO, 2004; LAZZARETTI & MELLO, 2005). A abordagem proposta define um processo que é capaz de descobrir RIS's através de uma técnica de mineração de dados conhecida como mineração de RA's. Esta técnica permite a descoberta de padrões de co-ocorrência e relacionamentos entre objetos de dados. Desta forma, não apenas tipos de dados primitivos podem ser descobertos, mas padrões complexos (padrões de relacionamentos estruturais e de conteúdo) e estes por sua vez, podem ser traduzidos para RIS's. No entanto,

algumas RIS's de menor complexidade também são abordadas através da análise da instância do esquema conceitual hierárquico CDX-Tree, definido a partir da estrutura das instâncias XML;

- **As RIS's descobertas são representadas em uma linguagem para representação de regras de conhecimento de alto nível.** A partir do estudo sobre os principais trabalhos presentes na literatura, a respeito de linguagens para especificações de RIS's, é possível constatar que há um crescente interesse da comunidade de pesquisa e diversas propostas têm surgido (CLARK & MAKOTO, 2001; DODDS, 2001; JACINTO et al., 2002a; KLARLUND et al., 2002; HU & TAO, 2004; LAZZARETTI & MELLO, 2005; NENTWICH, 2005). Entretanto, observa-se que não há ainda uma linguagem consolidada, principalmente pela falta de expressividade da XML Schema (DODDS, 2001; JACINTO et al., 2002a; BUNEMAN, 2003; HU & TAO, 2004), que seria a adoção natural para especificação de RIS. Desta forma, buscando representar as RIS's descobertas na forma mais abstrata possível, é adotada a SWRL como linguagem para especificação de RIS. A SWRL é uma linguagem para representação de regras de conhecimento. Uma vez que a SWRL é uma linguagem com alto nível de abstração e alto poder de expressão, ela facilita a tradução para linguagens específicas/proprietárias de definição de RIS's;

- **Uma taxionomia para RIS XML como forma de mensurar o escopo da abordagem proposta e que possa auxiliar na avaliação de expressividade de linguagens de especificação de RIS.** A fim de mensurar a abordagem proposta definindo claramente as categorias de RIS's capazes de serem descobertas por ela, é apresentada uma taxionomia para classificar as RIS's segundo os componentes que a compõe definidos em SANTOS (1980). Uma taxionomia analisada a partir dos componentes de uma RI se mostra uma boa escolha, pois permite distinguir as RIS's com relação a suas principais propriedades. Propriedades estas, que sistemas que pretendam dar suporte a RIS's devem considerar. Ainda, esta taxionomia teve como base o estudo sobre RIS realizado para o modelo de dados relacional e XML. O referido estudo demonstra que poucos trabalhos presentes na literatura apresentam uma taxionomia adequada

para classificar RIS XML (PAVLOVA et al., 2000; JACINTO et al., 2002a; HU & TAO, 2004; LAZZARETTI & MELLO, 2005), embora alguns apresentem poucas categorias (BUNEMAN et al., 2001; ARENAS et al., 2002a; KLARLUND et al., 2002; DEUTSCH & TANNEN, 2003; FAN & SIMÉON, 2003; NENTWICH, 2005). Contudo, eles se apresentam heterogêneos, não apresentam justificativa para as categorias propostas ou não analisam os componentes que compõem uma RI. Desta forma, se demonstram pouco abrangentes e inadequados tanto para avaliar a expressividade de linguagens de especificação de RIS's quanto para avaliar sistemas que dêem suporte a elas. As principais contribuições desta proposta de taxionomia são: *(i)* flexíbilidade, pois é especificada como uma classificação facetada; *(ii)* maior abrangência em relação ao universo de RIS's para documentos XML; *(iii)* apresenta justificativa para as categorias da taxionomia (baseado em propostas do modelo relacional com adaptações para o modelo XML e nas propostas existentes na literatura para o modelo de dados XML); e *(iv)* permite a avaliação da expressividade de linguagens de especificação de RIS's XML e conseqüente comparação entre estas, além de se mostrar adequada a avaliar a expressividade de sistemas que dêem suporte a RIS's quanto a estas.

7.2 Trabalhos futuros

Trabalhos futuros relacionados à abordagem DIInCX incluem um estudo da incorporação de outros algoritmos de mineração, avaliando sua aplicação sobre domínios específicos de dados. Considerando o uso das RIS's XML descobertas sobre domínios compostos por mais instâncias do que os tomados como dados de entrada, se torna necessária a análise de outros algoritmos de discretização. Uma possível aplicação da abordagem é a geração de RIS's através de um grupo de instâncias XML de treinamento e posterior aplicação sobre um domínio de dados composto por mais instâncias, embora na abordagem atual os percentuais de discretização adotados teriam de ser altos. Um estudo da incorporação de outros métodos de discretização tornaria a abordagem mais flexível (BAY, 2000). A análise da estrutura da instância CDX-Tree usando métodos de mineração, tal como a mineração de árvores freqüentes (CHI et al., 2005), é considerado um trabalho futuro relevante. No mais, a incorporação de uma

ontologia como forma de validar previamento os conceitos relevantes à aplicação da abordagem, descartando conceitos irrelevantes, otimizaria o processo de descoberta. Inclusive a adoção de uma ontologia poderia proporcionar a descoberta de RIS's mais precisas em relação ao resultado esperado.

Finalmente, pretende-se aplicar a abordagem DIInCX no contexto do SII XML *BInXS* (MELLO & HEUSER, 2005), o qual se encontra em desenvolvimento pelo Grupo de Banco de Dados da Universidade Federal de Santa Catarina[6].

7.3 Considerações finais

A relevância deste trabalho está na evolução de sistemas de informação no contexto do modelo de dados XML. A pesquisa em torno destes sistemas têm sido ampla e a abordagem proposta se mostra como um incentivo, propondo um meio de minimizar a árdua tarefa de definir todo o conjunto de RIS's pertencentes ao domínio de dados a fim de mantê-lo íntegro. A decisão de abordar instâncias XML se baseia, conforme previamente discutido, no fato de que a especificação de RIS's através de esquemas é incompleta, seja pela falta de conhecimento do usuário especialista ou pela falta de expressividade das linguagens de especificação de RIS.

Através da análise de experimentos conclui-se que todas as RIS's descobertas são válidas dentro do domínio de dados adotado para descoberta. No mais, a adoção de uma confiança de 100% para a mineração de RA's garante que a implicação é verdadeira para todo componente antecedente único. Por conseguinte, todas as RIS's descobertas puderam ser classificadas segundo a taxionomia para RIS's XML proposta. Desta forma, considera-se que a abordagem proposta pode auxiliar a atividade de diversos sistemas de informação como SGBD's XML e SII's XML, complementando a semântica de seus dados através da descoberta de RIS's que poderiam não ser percebidas por um usuário especialista.

A taxionomia proposta se mostrou adequada como forma de mensurar as RIS's XML descobertas. Entretanto, espera-se que ela contribua também para o avanço da pesquisa de RIS's XML, facilitando a avaliação da expressividade e definição de

---

[6] *http://www.grupobd.inf.ufsc.br*

escopo de trabalhos no contexto de RIS's XML. A escolha pelo modelo relacional como base para algumas categorias se deu devido ao seu caráter consolidado e já amplamente discutido.

BIBLIOGRAFIA

AGRAWAL, R.; SRIKANT, R. Fast Algorithms for Mining Association Rules. Very Large Database Conference (VLDB), Chile, 1994.

AGRAWAL, R.; IMIELINSKI, T.; SWAMI, A. Mining Association Rules Between Sets of Items in Large Databases. Em: ACM SIGMOD Conference on Management of Data, E.U.A., 1993. p. 207–216.

ALANI, H. et al. Automatic Ontology-Based Knowledge Extraction from Web Documents. Em: IEEE Intelligent Systems, v. 18, 2003. p. 14-21.

ALIMOHAMMADZADEH, R.; SOLTAN, S.; RAHGOZAR, M. Template guided association rule mining from XML documents. Em: International World Wide Web Conference (WWW), 2006. p. 963-964.

AMO, S. Técnicas de Mineração de Dados. Congresso da Sociedade Brasileira de Computação (SBC), Jornada de Atualização em Informática, Salvador, Bahia, Brasil, 2004.

ARENAS, M.; FAN, W.; LIBKIN, L. On Verifying Consistency of XML Specifications. Em: Symposium on Principles of Database Systems, E.U.A., 2002. p. 259-270.

___. What's Hard about XML Schema Constraints. Em: International Conference on Database and Expert Systems, 2002.

BAY, S.D. Multivariate discretization of continuous variables for set mining. Em: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000., p. 315-319.

BAYARDO, R.J. The Hows, Whys, and Whens of Constraints in Itemset and Rule Discovery. Em: Constraint-Based Mining and Inductive Databases, 2004. p 1-13.

BENSLIMANE, S.M.; BENSLIMANE, D.; MALKI, M. Acquiring OWL Ontologies from Data-Intensive Web Sites. Em: International Conference on Web Engineering, Palo Alto, California, E.U.A., 2006. p. 361-368.

BRAGA, D. et al. A Tool for Extracting XML Association Rules. Em: IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2002.

BUNEMAN, P. et al. Constraints for Semistructured Data and XML. Em: SIGMOD Record, v. 30, ed. 1, 2001.

___. Keys for XML. Em: International World Wide Web Conference (WWW). Hong Kong, China, 2001. p. 201-210.

BÜCHNER A.G. et al. Data Mining and XML: Current and Future Issues. Em: International Conference on Web Information Systems Engineering, v. 2, 2000. p. 2131.

CALI, A. et al. Data Integration Under Integrity Constraints. Em: Conference on Advanced Information Systems Engineering (CAISE), 2002. p. 262-279.

CASTANO, S. et al. Semi-Automated Extraction of Ontological Knowledge from XML Datasources. Em: DEXA Workshops, 2002. p 852-860.

CHI, Y. et al. Frequent Subtree Mining : An Overview. Em: Fundamenta Informaticae, 66(1-2), 2005. p. 161-198.

CHIDLOVSKII, B. Schema Extraction from XML Collections. Em: ACM/IEEE-CS joint conference on Digital libraries, ACM Press, 2002. p. 291-292.

CLARK, J.; MAKOTO, M. RELAX NG Specification. Technical report. Em: Organization for the Advancement of Structured Information Standards (OASIS), 2001.

CODD, E.F. Data Models in Database Management. Em: International Conference on Management of Data, E.U.A., 1980. p. 112-114.

CRUZ, F.; XIAO, H.; HSU, F. An Ontology-Based Framework for XML Semantic Integration. Em: International Database Engineering and Applications Symposium (IDEAS), Portugal, 2004. p. 217-226.

DAM+OIL, Reference Description. Disponível em: http://www.w3.org/TR/daml+oil-reference. Último acesso em novembro de 2007.

DATAMINING, Michael Holler`s web site. Disponível em: http://www.helsinki.fi/~holler/datamining/. Último acesso em novembro de 2007.

DATE, C.J. An Introduction to Database Systems. Ed. Addison Wesley, 2003. 8a edição.

DELOBEL, C. et al. Semantic Integration in Xyleme: a Uniform Tree-Based Approach. Em: Data & Knowledge Engineering, 2003. p. 267–298.

DEUTSCH, A.; TANNEN, V. Reformulation of XML Queries and Constraints. Em: International Conference on Database Theory, 2003.

DEUTSCH, A.; FERNANDEZ, M.; SUCIU. D. Storing semistructured data with stored. Em: ACM SIGMOD Conference on Management of Data, Junho.1999.

DODDS, L. Schematron: Validating XML Using XSLT. Em: XSLT UK Conference, Inglaterra, 2001.

DOUGHERTY, J.; KOHAVI, R.; SAHAMI, M. Supervised and Unsupervised Discretization of Continuous Features. Em: International Conference on Machine Learning, E.U.A., 1995. p. 194-202.

DTD, Guide to the W3C XML Specification. Disponível em: http://www.w3.org/XML/1998/06/xmlspec-report.htm. Último acesso em novembro de 2007.

ELMASRI, R.; NAVATHE, S.B. Fundamentals of Database Systems. Ed. Addison Wesley, 2003. 4a edição.

EMBLEY, D.W. et al. Conceptual Model-Based Data Extraction of Information from Multi-Record Web Documents. Em: Data & Knowledge Engineering, v. 31, n. 3, 1999. p. 227-251.

ERDMANN, M.; STUDER, R. How to Structure and Access XML Documents with Ontologies. Em: Data and Knowledge Engineering, v. 36, 2001. p. 317-335.

FAN, W.; SIMÉON, J. Integrity Constraints for XML. Em: Journal of Computer and System Sciences, 2003. p. 254-291.

FAN, W. XML Constraints: Specification, Analysis, and Applications. Em: International Workshop on Database and Expert Systems Applications, 2005. p. 805-809.

FENG, L.; DILLON, T.S. Mining Interesting XML-Enabled Association Rules with Templates. Em: International Workshop on Knowledge Discovery in Inductive Databases (KDID), Pisa, Itália, 2004. p. 66-88.

GARBONI, C.; MASSEGLIA, F.; TROUSSE, B. Sequential Pattern Mining for Structure-Based XML Document Classification. Em: Initiative for the Evaluation of XML Retrieval (INEX), 2006.

GAROFALAKIS, M.N. et al. Data Mining and the Web: Past, Present and Future. Em: Workshop on Web Information and Data Management, E.U.A., 1999. p. 43-47.

GYORÖDI, R. A Comparative Study of Iterative Algorithms in Association Rules Mining. Em: Studies in Informatics and Control Journal, v. 12, n. 3, 2003. p. 205-215.

HACID, M.-S.; SOUALMIA, F.; TOUMANI, F.. Schema Extraction for Semi-Structured Data. Em: Description Logics, 2000. p. 133-142.

HAN, J.; PEI, J.; YIN, Y. Mining frequent patterns without candidate generation. Em: ACM SIGMOD International Conference on Management of Data, v. 29, n. 2, 2000. p. 1-13.

HEGEWALD, J.; NAUMANN, F.; WEIS, M. XStruct: Efficient Schema Extraction from Multiple and Large XML Documents. Em: International Conference on Data Engineering Workshops, E.U.A., 2006.

HIPP, J.; GUNTZER, U.; NAKAEIZADEH, G. Algorithms for Association Rule Mining - A General Survey and Comparison. Em: Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000.

HU, J.; TAO, L. An Extensible Constraint Markup Language: Specification, Modeling, and Processing. Em: XML Conference and Exhibition, E.U.A., 2004.

JACINTO, M.H. et al. XCSL: Constraint Specification Language. Em: Latin American Conference on Informatics, 2002.

___. Constraint Specification Languages: Comparing XCSL, Schematron and XML-Schemas. Em: XML EUROPE, Espanha, 2002.

KLARLUND, N.; MOLLER, A.; SCHWARTZBACH, I.M. The DSD Schema Language. Em: Automated Software Engineering, v. 9, 2002. p. 285-319.

KIMBALL, R. The data warehouse toolkit: practical techniques for building dimensional data warehouses. Ed. John Wiley & Sons, New York, NY, E.U.A, 1996.

LAENDER, A. et al. A Brief Survey of Web Data Extraction Tools. Em: SIGMOD Record, v. 31, n.2, 2002.

LAZZARETTI, A.T.; MELLO, R.S. A Domain Integrity Constraint Control for XML Documents. Em: Simpósio Brasileiro de Banco de Dados (SBBD), Brasil, 2005.

LAZZARETTI, A.T. XDC: Uma Proposta de Controle de Restrições de Integridade de Domínio em Documentos XML. Dissertação de Mestrado, PPGCC – UFSC, Florianópolis, Santa Catarina, 2005.

LEE, D.; CHU, W.W. Comparative Analysis of Six XML Schema Languages. Em: ACM SIGMOD Record, v. 29, 2000. p. 76-87.

LETHI, P.; FANKHAUSE, P. XML Data Integration with OWL: Experiences & Challenges. Em: International Symposium on Applications and the Internet (SAINT), Japão, 2004. p. 160-170.

LIU, Y.; ZHONG, H.; WANG, Y. Capturing XML Constraints with Relational Schema. Em: International Conference on Computer and Information Technology (CIT), v. 0, 2004. p. 309-314.

MA, H.; SCHEWE, K.-D. Fragmentation of XML Documents. Em: Simpósio Brasileiro de Banco de Dados (SBBD), Brasil, 2003.

YANG, Y. Discretization for naive-Bayes learning. Tese de Doutorado em Ciência da Computação, School of Computer Science and Software Engineering of Monash University, Melbourne, 2003.

MEIER, W. eXist: An Open Source Native XML Database. Em: Lecture Notes in Computer Science, v. 2593, Alemanha, 2002. p. 169-183.

MELLO, R.S.; HEUSER, C.A. BInXS: A Process for Integration of XML Schemata. Em: Conference on Advanced Information Systems Engineering (CAiSE), 2005. p. 151-166.

MURATA, M. et al. Taxonomy of XML Schema Languages Using Formal Language Theory. Em: ACM Transactions on Internet Technology (TOIT), v. 5, 2005. p. 660-704.

NAYAK, R.; WITT, R.; TONEV, A. Data Mining and XML documents. Em: International Conference on Internet Computing, Nevada, E.U.A., 2002. p. 660-667.

NENTWICH, C. CLiX - A Validation Rule Language for XML. Em Rule Languages for Interoperability, E.U.A., 2005.

NESTOROV, S.; ABITEBOUL, S.; MOTWANI, R. Extracting Schema from Semistructured Data. Em: International Conference on Management of Data, E.U.A., 1998.

NOY, N.F. Semantic Integration: A Survey of Ontology-Based Approaches. Em: ACM SIGMOD Record, v. 33, ed. 4, 2004. p. 65-70.

OWL, Ontology Web Language. Disponível em: http://www.w3.org/TR/owl-features/. Último acesso em novembro de 2007.

PAVLOVA, E.; NEKRESTYANOV, I.; NOVIKOV, B. Constraints for Semistructured Data. Em: Russian Conference on Digital Libraries, Russia, 2000.

PROTÉGÉ, Disponível em: http://protege.stanford.edu/. Último acesso em novembro de 2007.

RANGANATHAN, S.R. Prolegomena to Library Classification. Asian Publishing House, India, 1967.

REYNAUD, C.; SIROT, J.; VODISLAV, D. Semantic Integration of XML Heterogeneous Data Sources. Em: International Database Engineering & Applications Symposium (IDEAS), França, 2001. p. 199-208.

RODRIGUES, K.R.; MELLO, R.S. DIInCX: An Approach to Discovery of Implicit Integrity Constraints from XML Data. Em: IEEE International Conference on Information Reuse and Integration (IRI), Las Vegas, E.U.A., 2007. p. 606-611.

___. A Faceted Taxonomy of Semantic Integrity Constraints for the XML Data Model. Em: International Conference in Database and Expert Systems Applications (DEXA), Springer Berlin, v. 4653, 2007. p. 65-74.

___. Um Estudo e Proposta de Abordagem para Extração Semi-Automática de Restrições de Integridade Semântica de Dados XML Em: Escola Regional de Banco de Dados (ERBD), Caxias do Sul, Brasil, 2007. p. 194-203.

SANTOS, C.S. et al. Towards Constructive Axiomatic Specifications. Em: International Conference on Management of Data, 1980. p. 183-185.

SANTOS, C.S. Caracterização Sistemática de Restrições de Integridade em Bancos de Dados. Tese de Doutorado, D.I-PUC, Rio de Janeiro, Rio de Janeiro. 1980.

SCHÖNING, H. Tamino - A DBMS Designed for XML. Em: International Conference on Data Engineering, 2001. p. 149-154.

SILBERSCHATZ, A.; KORTH, H.F.; SUDARSHAN, S. Database System Concepts. McGraw-Hill, 2005. 5a edição.

SMITH, D.; LOPEZ, M. Information Extraction for Semi-structured Documents. Em: Workshop on Management of Semistructured Data, PODS/SIGMOD., E.U.A., 1997.

SNOUSSI, H.; MAGNIN, L.; NIE, J.-Y. Toward an Ontology-based Web Data Extraction. Em: Workshop on Business Agents and the Semantic Web, Canadian Conference on Artificial Intelligence (AI), Calgary, Alberta, Canada, 2002.

SRIKANT, R.; VU, Q.; Agrawal, R. Mining Association Rules with Item Constraints. Em: International Conference on Knowledge Discovery and Data Mining, E.U.A., 1997. p. 67-73.

SVÁTEK, V.; RAUCH, J.; RALBOVSKÝ, M. Ontology-Enhanced Association Mining. Em: Semantics, Web and Mining, Ed. Springer Berlin / Heidelberg, v. 4289, 2006. p. 163-179 .

SWRL, Semantic Web Rule Language. Disponível em: http://www.w3.org/submission/swrl. Último acesso em novembro de 2007.

TATARINOV, I. et al. Updating XML. Em: International Conference on Management of Data, 2001. p. 413-424.

WAN, J.W.W.; DOBBIE, G. Mining association rules from XML data using XQuery. Em: ACM International Conference Proceeding Series v. 54, ed. Australian Computer Society, Dunedin, Nova Zelândia, 2004. p. 169-174.

WEKA, Disponível em: http://www.cs.waikato.ac.nz/ml/weka. Último acesso em novembro de 2007.

WIWATWATTANA, N.; WU, Y.; YU, C. TIMBER: A Native System for Querying XML. Em: SIGMOD Conference, 2003. p. 672.

XML SCHEMA, Disponível em: http://www.w3.org/XML/Schema. Último acesso em novembro de 2007.

XML, Extensible Markup Language. Disponível em: http://www.w3.org/xml. Último acesso em novembro de 2007.

ANEXOS

ANEXO 1 – EXEMPLO DE INSTÂNCIA XML

```
<professor degree =”doctor” >
        <age>60</age>
        <email>inácio.dorvantil@gmail.com</email>
        <completeName>
            <firstName=“Inacio”/>
            <lastName=”Dorvantil“/>
         </completeName>
        <courses>
            <course>
                <subject>Introduction to Database Systems</subject>
            </course>
            <course>
                <subject>Principles of Software Engineering</subject>
            </course>
        </courses>
        <researchLines>
            <researchLine>
                <name>Database</name>
            </researchLine>
            <researchLine>
                <name> Software Engineering </name>
            </researchLine>
        <researchLines>
        <masterStudents>
            <masterStudent>
                <name>Cristiane Girardi</name>
                <email>cristianegi@gmail.com</email>
            </masterStudent>
            <masterStudent>
                <name>Marta Grecia</name>
            </masterStudent>
            <masterStudent>
                <name>Mathina Rodrigues</name>
                <email>mathina@gmail.com</email>
            </masterStudent>
        </masterStudents>
        <graduatedStudents>
            <graduatedStudent>
                <name>Iolanda Rezende</name>
                <email>iolanda@gmail.com</email>
            </graduatedStudent>
            <graduatedStudent>
                <name>Ivo Callado</name>
            </graduatedStudent>
        </graduatedStudents>
        <publications>
            <publication name=”CRIS-XML”>
                <conference>
                    DEXA
                    <qualis>B</qualis>
                </conference>
            </publication>
            <publication name=”RIS-XML”>
                <conference>
                    SBBD
                    <qualis>A</qualis>
                </conference>
```

```
            </publication>
        </publications>
        <job>Teacher</job>
        <salary>5000,00</salary>
        <laboratory>
            <name>Database Laboratory</name>
        </laboratory>
        <position>Coordinator</position>
</professor>
```

ANEXO 2 – CLASSES DA CDX-TREE EM OWL.

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns="http://www.inf.ufsc.br/~khaue/diincx/cdx_tree_diincx.owl#"
   xml:base="http://www.inf.ufsc.br/~khaue/diincx/cdx_tree_diincx.owl">
   <owl:Ontology rdf:about="">
   <owl:imports rdf:resource="http://www.w3.org/2003/11/swrl"/>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
   CDX-TREE is part of DIInCX approach (Discovery of Integrity Constraint from XML)
   </rdfs:comment>
   <owl:versionInfo rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      1.0
   </owl:versionInfo>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      CDX-TREE
   </rdfs:label>
   <owl:imports rdf:resource="http://www.w3.org/2003/11/swrlb"/>
   </owl:Ontology>

  <owl:Class rdf:ID="leafNode">
   <owl:disjointWith>
     <owl:Class rdf:ID="collectionNode"/>
   </owl:disjointWith>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
       >1</owl:minCardinality>
       <owl:onProperty>
         <owl:DatatypeProperty rdf:ID="hasValue"/>
       </owl:onProperty>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty>
         <owl:ObjectProperty rdf:ID="hasParent"/>
       </owl:onProperty>
       <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
       >1</owl:cardinality>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
       >1</owl:cardinality>
```

```
    <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="relevance"/>
    </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="originalPath"/>
    </owl:onProperty>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
    >1</owl:cardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Class rdf:ID="nodeElement"/>
 </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="treeElement">
 <owl:disjointWith rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#List"/>
 <owl:disjointWith>
  <rdf:Description rdf:about="http://www.w3.org/2003/11/swrl#Atom">
    <owl:disjointWith rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#List"/>
    <owl:disjointWith>
      <rdf:Description rdf:about="http://www.w3.org/2003/11/swrl#Builtin">
        <owl:disjointWith rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#List"/>
        <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Atom"/>
        <owl:disjointWith>
          <rdf:Description rdf:about="http://www.w3.org/2003/11/swrl#Imp">
            <owl:disjointWith rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#List"/>
            <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Atom"/>
            <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Builtin"/>
            <owl:disjointWith>
              <rdf:Description rdf:about="http://www.w3.org/2003/11/swrl#Variable">
                <owl:disjointWith rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#List"/>
                <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Atom"/>
                <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Builtin"/>
                <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Imp"/>
                <owl:disjointWith rdf:resource="#treeElement"/>
              </rdf:Description>
            </owl:disjointWith>
            <owl:disjointWith rdf:resource="#treeElement"/>
          </rdf:Description>
        </owl:disjointWith>
        <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
        <owl:disjointWith rdf:resource="#treeElement"/>
      </rdf:Description>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Imp"/>
    <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
    <owl:disjointWith rdf:resource="#treeElement"/>
  </rdf:Description>
 </owl:disjointWith>
 <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Builtin"/>
 <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Imp"/>
 <owl:disjointWith rdf:resource="http://www.w3.org/2003/11/swrl#Variable"/>
</owl:Class>

<owl:Class rdf:ID="rootTree">
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
```

```
    >1</owl:cardinality>
    <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="instancesAmount"/>
    </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="hasChild"/>
    </owl:onProperty>
    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
    >1</owl:minCardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf rdf:resource="#treeElement"/>
 <owl:disjointWith>
  <owl:Class rdf:about="#nodeElement"/>
 </owl:disjointWith>
</owl:Class>
<owl:Class rdf:about="#nodeElement">
 <rdfs:subClassOf rdf:resource="#treeElement"/>
 <owl:disjointWith rdf:resource="#rootTree"/>
</owl:Class>

<owl:Class rdf:about="#collectionNode">
 <owl:disjointWith rdf:resource="#leafNode"/>
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:about="#hasChild"/>
    </owl:onProperty>
    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</owl:minCardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:DatatypeProperty rdf:about="#hasValue"/>
    </owl:onProperty>
    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</owl:minCardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:about="#hasParent"/>
    </owl:onProperty>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
    >1</owl:cardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
    >1</owl:cardinality>
    <owl:onProperty>
      <owl:DatatypeProperty rdf:about="#relevance"/>
    </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
```

```
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
    >1</owl:cardinality>
    <owl:onProperty>
     <owl:DatatypeProperty rdf:about="#originalPath"/>
    </owl:onProperty>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#nodeElement"/>
</owl:Class>

<owl:ObjectProperty rdf:about="http://www.w3.org/2003/11/swrl#argument2"/>

<owl:ObjectProperty rdf:about="#hasChild">
  <owl:inverseOf>
   <owl:ObjectProperty rdf:about="#hasParent"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#nodeElement"/>
  <rdfs:domain rdf:resource="#treeElement"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#hasParent">
  <rdfs:domain rdf:resource="#nodeElement"/>
  <rdfs:range rdf:resource="#treeElement"/>
  <owl:inverseOf rdf:resource="#hasChild"/>
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:about="#relevance">
  <rdfs:domain rdf:resource="#nodeElement"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#instancesAmount">
  <rdfs:domain rdf:resource="#rootTree"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#hasValue">
  <rdfs:domain>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#leafNode"/>
     <owl:Class rdf:about="#collectionNode"/>
    </owl:unionOf>
   </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="hasQuantifiedValue">
  <rdfs:subPropertyOf rdf:resource="#hasValue"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#double"/>
  <rdfs:domain>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#collectionNode"/>
     <owl:Class rdf:about="#leafNode"/>
    </owl:unionOf>
   </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#originalPath">
```

```
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#nodeElement"/>
 </owl:DatatypeProperty>

 <owl:DatatypeProperty rdf:ID="hasQualifiedValue">
  <rdfs:subPropertyOf rdf:resource="#hasValue"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#leafNode"/>
 </owl:DatatypeProperty>

 <owl:FunctionalProperty rdf:ID="hasConstantValue">
  <rdfs:domain>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#collectionNode"/>
     <owl:Class rdf:about="#leafNode"/>
    </owl:unionOf>
   </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:subPropertyOf rdf:resource="#hasValue"/>
 </owl:FunctionalProperty>

 <owl:FunctionalProperty rdf:ID="hasDispersedValue">
  <rdfs:domain rdf:resource="#leafNode"/>
  <rdfs:range>
   <owl:DataRange>
    <owl:oneOf rdf:parseType="Resource">
     <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
     >true</rdf:first>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </owl:oneOf>
   </owl:DataRange>
  </rdfs:range>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:subPropertyOf rdf:resource="#hasValue"/>
 </owl:FunctionalProperty>

</rdf:RDF>
```

ANEXO 3 – INSTÂNCIA DO ESQUEMA CONCEITUAL CDX-TREE

```
 <rootTree rdf:ID="professor">
        <instancesAmount rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >40</instancesAmount>
        <hasChild>
         <collectionNode rdf:ID="graduatedStudents">
          <hasParent rdf:resource="#professor"/>
          <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >/professor/graduatedStudents</originalPath>
          <hasChild>
           <leafNode rdf:ID="graduatedStudent.email">
            <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
            >true</hasDispersedValue>
            <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >/professor/graduatedStudents/graduatedStudent/email</originalPath>
            <hasParent rdf:resource="#graduatedStudents"/>
            <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >25</relevance>
           </leafNode>
```

```
    </hasChild>
    <hasChild>
     <leafNode rdf:ID="graduatedStudent.name">
      <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >29</relevance>
      <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >/professor/graduatedStudents/graduatedStudent/name</originalPath>
      <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
      >true</hasDispersedValue>
      <hasParent rdf:resource="#graduatedStudents"/>
     </leafNode>
    </hasChild>
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >10</hasQuantifiedValue>
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >5</hasQuantifiedValue>
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >0</hasQuantifiedValue>
    <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >29</relevance>
   </collectionNode>
  </hasChild>
  <hasChild>
   <leafNode rdf:ID="completeName.firstName">
    <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
    >true</hasDispersedValue>
    <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >/professor/completeName/firstName</originalPath>
    <hasParent rdf:resource="#professor"/>
    <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >40</relevance>
   </leafNode>
  </hasChild>
  <hasChild>
   <leafNode rdf:ID="position">
    <hasParent rdf:resource="#professor"/>
    <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >/professor/position</originalPath>
    <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >4</relevance>
    <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Coordinator</hasQualifiedValue>
    <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Director</hasQualifiedValue>
   </leafNode>
  </hasChild>
  <hasChild>
   <collectionNode rdf:ID="publications">
    <hasChild>
     <leafNode rdf:ID="publication.conference.qualis">
      <hasParent rdf:resource="#publications"/>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >A</hasQualifiedValue>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >B</hasQualifiedValue>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >C</hasQualifiedValue>
      <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >/professor/publications/publication/conference/qualis</originalPath>
      <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >9</relevance>
     </leafNode>
    </hasChild>
```

```
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >20</hasQuantifiedValue>
    <hasParent rdf:resource="#professor"/>
    <hasChild>
     <leafNode rdf:ID="publication.name">
      <hasParent rdf:resource="#publications"/>
      <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
      >true</hasDispersedValue>
      <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >17</relevance>
      <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >/professor//publications/publication@name</originalPath>
     </leafNode>
    </hasChild>
    <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >17</relevance>
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >0</hasQuantifiedValue>
    <hasChild>
     <leafNode rdf:ID="publication.conference">
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >CAiSE</hasQualifiedValue>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >ER</hasQualifiedValue>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >SBBD</hasQualifiedValue>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >DEXA</hasQualifiedValue>
      <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >17</relevance>
      <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >/professor/publications/publication/conference</originalPath>
      <hasParent rdf:resource="#publications"/>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >ERBD</hasQualifiedValue>
      <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >IRI</hasQualifiedValue>
     </leafNode>
    </hasChild>
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >6</hasQuantifiedValue>
    <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >14</hasQuantifiedValue>
    <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >/professor/publications</originalPath>
   </collectionNode>
  </hasChild>
  <hasChild>
   <leafNode rdf:ID="degree">
    <hasParent rdf:resource="#professor"/>
    <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >40</relevance>
    <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >doctor</hasQualifiedValue>
    <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >master</hasQualifiedValue>
    <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >graduated</hasQualifiedValue>
    <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >/professor@degree</originalPath>
   </leafNode>
  </hasChild>
  <hasChild>
```

```
  <collectionNode rdf:ID="researchLines">
   <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >40</relevance>
   <hasParent rdf:resource="#professor"/>
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/researchLines</originalPath>
   <hasChild>
    <leafNode rdf:ID="researchLine.name">
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Software Engineering and Databases</hasQualifiedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Design Automation of Embedded Computing Systems</hasQualifiedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Computational Intelligence</hasQualifiedValue>
     <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >/professor/researchLines/researchLine/name</originalPath>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Security in Computational Systems</hasQualifiedValue>
     <hasParent rdf:resource="#researchLines"/>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Computer Networks</hasQualifiedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Parallel and Distributed Computing</hasQualifiedValue>
     <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >40</relevance>
    </leafNode>
   </hasChild>
   <hasConstantValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >2</hasConstantValue>
  </collectionNode>
 </hasChild>
 <hasChild>
  <leafNode rdf:ID="age">
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >36</hasQuantifiedValue>
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/age</originalPath>
   <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >40</relevance>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >54</hasQuantifiedValue>
   <hasParent rdf:resource="#professor"/>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >42</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >48</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >30</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >60</hasQuantifiedValue>
  </leafNode>
 </hasChild>
 <hasChild>
  <collectionNode rdf:ID="courses">
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >6</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >3</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >1</hasQuantifiedValue>
   <hasParent rdf:resource="#professor"/>
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/courses</originalPath>
```

```
  <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >40</relevance>
   <hasChild>
    <leafNode rdf:ID="course.subject">
     <hasParent rdf:resource="#courses"/>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Software Engineering</hasQualifiedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Database</hasQualifiedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Distributed Systems</hasQualifiedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Operational Systems</hasQualifiedValue>
     <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
     >true</hasDispersedValue>
     <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Formal Language and Compilers</hasQualifiedValue>
     <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >/professor/courses/course/subject</originalPath>
     <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >40</relevance>
    </leafNode>
   </hasChild>
  </collectionNode>
 </hasChild>
 <hasChild>
  <leafNode rdf:ID="email">
   <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
   >true</hasDispersedValue>
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/email</originalPath>
   <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >20</relevance>
   <hasParent rdf:resource="#professor"/>
  </leafNode>
 </hasChild>
 <hasChild>
  <leafNode rdf:ID="job">
   <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >40</relevance>
   <hasParent rdf:resource="#professor"/>
   <hasConstantValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >professor</hasConstantValue>
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/job</originalPath>
  </leafNode>
 </hasChild>
 <hasChild>
  <leafNode rdf:ID="laboratory.name">
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/laboratory/name</originalPath>
   <hasParent rdf:resource="#professor"/>
   <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >23</relevance>
   <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Database Laboratory</hasQualifiedValue>
   <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Distrbuted Systems Laboratory</hasQualifiedValue>
   <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Software Engineering Laboratory</hasQualifiedValue>
   <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Operational Systems Laboratory</hasQualifiedValue>
   <hasQualifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
```

```
      >Multimedia Laboratory </hasQualifiedValue>
     </leafNode>
    </hasChild>
    <hasChild>
     <leafNode rdf:ID="salary">
      <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >40</relevance>
      <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
      >5000</hasQuantifiedValue>
      <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
      >3050</hasQuantifiedValue>
      <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
      >1000</hasQuantifiedValue>
      <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >/professor/salary</originalPath>
      <hasParent rdf:resource="#professor"/>
     </leafNode>
    </hasChild>
    <hasChild>
     <leafNode rdf:ID="completeName.lastName">
      <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
      >true</hasDispersedValue>
      <hasParent rdf:resource="#professor"/>
      <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >/professor/completeName/lastname</originalPath>
      <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >40</relevance>
     </leafNode>
    </hasChild>
    <hasChild>
     <collectionNode rdf:ID="masterStudents">
      <hasChild>
       <leafNode rdf:ID="masterStudent.name">
        <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >/professor/masterStudents/masterStudent/name</originalPath>
        <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
       >15</relevance>
        <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
        >true</hasDispersedValue>
      <hasParent rdf:resource="#masterStudents"/>
       </leafNode>
      </hasChild>
     <hasChild>
  <leafNode rdf:ID="masterStudent.email">
   <hasDispersedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
   >true</hasDispersedValue>
   <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >/professor/masterStudents/masterStudent/email</originalPath>
   <hasParent rdf:resource="#masterStudents"/>
   <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
   >11</relevance>
  </leafNode>
 </hasChild>
  <relevance rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
  >15</relevance>
  <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >3</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
   >6</hasQuantifiedValue>
   <hasQuantifiedValue rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
    >0</hasQuantifiedValue>
    <originalPath rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >/professor/masterStudents</originalPath>
```

```
            <hasParent rdf:resource="#professor"/>
          </collectionNode>
         </hasChild>
       </rootTree>
```

ANEXO 4 – RIS's ESPECIFICADAS EM SWRL.

```
<swrl:Imp rdf:ID="RIS22">
   <swrl:head>
    <swrl:AtomList>
     <rdf:rest>
      <swrl:AtomList>
       <rdf:first>
        <swrl:BuiltinAtom>
         <swrl:arguments>
          <rdf:List>
           <rdf:rest>
            <rdf:List>
             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >0</rdf:first>
             <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            </rdf:List>
           </rdf:rest>
           <rdf:first>
            <swrl:Variable rdf:ID="coursesValue"/>
           </rdf:first>
          </rdf:List>
         </swrl:arguments>
         <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
        </swrl:BuiltinAtom>
       </rdf:first>
       <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      </swrl:AtomList>
     </rdf:rest>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:arguments>
        <rdf:List>
         <rdf:rest>
          <rdf:List>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >6</rdf:first>
          </rdf:List>
         </rdf:rest>
         <rdf:first rdf:resource="#coursesValue"/>
        </rdf:List>
       </swrl:arguments>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
      </swrl:BuiltinAtom>
     </rdf:first>
    </swrl:AtomList>
   </swrl:head>
   <swrl:body>
    <swrl:AtomList>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     <rdf:first>
      <swrl:DatavaluedPropertyAtom>
       <swrl:argument1 rdf:resource="#courses"/>
       <swrl:argument2 rdf:resource="#coursesValue"/>
       <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
      </swrl:DatavaluedPropertyAtom>
```

```
    </rdf:first>
   </swrl:AtomList>
  </swrl:body>
</swrl:Imp>
<swrl:Variable rdf:ID="salaryValue"/>
<swrl:Imp rdf:ID="RIS15">
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   <rdf:first>
    <swrl:BuiltinAtom>
     <swrl:arguments>
      <rdf:List>
       <rdf:first>
        <swrl:Variable rdf:ID="degreeValue"/>
       </rdf:first>
       <rdf:rest>
        <rdf:List>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         <rdf:first>
          <swrl:Variable rdf:ID="degreeValueSet"/>
         </rdf:first>
        </rdf:List>
       </rdf:rest>
      </rdf:List>
     </swrl:arguments>
     <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#member"/>
    </swrl:BuiltinAtom>
   </rdf:first>
  </swrl:AtomList>
 </swrl:head>
 <swrl:body>
  <swrl:AtomList>
   <rdf:rest>
    <swrl:AtomList>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#listConcat"/>
       <swrl:arguments>
        <rdf:List>
         <rdf:first rdf:resource="#degreeValueSet"/>
         <rdf:rest>
          <rdf:List>
           <rdf:rest>
            <rdf:List>
             <rdf:rest>
              <rdf:List>
               <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
               <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
               >graduated</rdf:first>
              </rdf:List>
             </rdf:rest>
             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
             >doctor</rdf:first>
            </rdf:List>
           </rdf:rest>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
           >master</rdf:first>
          </rdf:List>
         </rdf:rest>
        </rdf:List>
       </swrl:arguments>
```

```
          </swrl:BuiltinAtom>
         </rdf:first>
        </swrl:AtomList>
       </rdf:rest>
       <rdf:first>
        <swrl:DatavaluedPropertyAtom>
         <swrl:argument1 rdf:resource="#degree"/>
         <swrl:argument2 rdf:resource="#degreeValue"/>
         <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
        </swrl:DatavaluedPropertyAtom>
       </rdf:first>
      </swrl:AtomList>
     </swrl:body>
    </swrl:Imp>
    <swrl:Imp rdf:ID="RIS8">
     <swrl:head>
      <swrl:AtomList>
       <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       <rdf:first>
        <swrl:ClassAtom>
         <swrl:classPredicate rdf:resource="#collectionNode"/>
         <swrl:argument1 rdf:resource="#researchLines"/>
        </swrl:ClassAtom>
       </rdf:first>
      </swrl:AtomList>
     </swrl:head>
     <swrl:body>
      <swrl:AtomList/>
     </swrl:body>
    </swrl:Imp>
    <swrl:Variable rdf:ID="publicationConferenceQualisValueSet"/>
    <swrl:Imp rdf:ID="RIS33">
     <swrl:head>
      <swrl:AtomList>
       <rdf:first>
        <swrl:BuiltinAtom>
         <swrl:arguments>
          <rdf:List>
           <rdf:rest>
            <rdf:List>
             <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
             >5000</rdf:first>
            </rdf:List>
           </rdf:rest>
           <rdf:first rdf:resource="#salaryValue"/>
          </rdf:List>
         </swrl:arguments>
         <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
        </swrl:BuiltinAtom>
       </rdf:first>
       <rdf:rest>
        <swrl:AtomList>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         <rdf:first>
          <swrl:BuiltinAtom>
           <swrl:arguments>
            <rdf:List>
             <rdf:first rdf:resource="#salaryValue"/>
             <rdf:rest>
              <rdf:List>
               <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
               >3050</rdf:first>
```

```
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
               </rdf:List>
              </rdf:rest>
             </rdf:List>
            </swrl:arguments>
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
           </swrl:BuiltinAtom>
          </rdf:first>
         </swrl:AtomList>
        </rdf:rest>
       </swrl:AtomList>
      </swrl:head>
      <swrl:body>
       <swrl:AtomList>
        <rdf:rest>
         <swrl:AtomList>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first>
           <swrl:DatavaluedPropertyAtom>
            <swrl:argument2 rdf:resource="#salaryValue"/>
            <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
            <swrl:argument1 rdf:resource="#salary"/>
           </swrl:DatavaluedPropertyAtom>
          </rdf:first>
         </swrl:AtomList>
        </rdf:rest>
        <rdf:first>
         <swrl:DatavaluedPropertyAtom>
          <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >doctor</swrl:argument2>
          <swrl:argument1 rdf:resource="#degree"/>
          <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
         </swrl:DatavaluedPropertyAtom>
        </rdf:first>
       </swrl:AtomList>
      </swrl:body>
     </swrl:Imp>
     <swrl:Imp rdf:ID="RIS40">
      <swrl:head>
       <swrl:AtomList>
        <rdf:rest>
         <swrl:AtomList>
          <rdf:first>
           <swrl:BuiltinAtom>
            <swrl:arguments>
             <rdf:List>
              <rdf:first>
               <swrl:Variable rdf:ID="graduatedStudentsAmount"/>
              </rdf:first>
              <rdf:rest>
               <rdf:List>
                <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >5</rdf:first>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
               </rdf:List>
              </rdf:rest>
             </rdf:List>
            </swrl:arguments>
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
           </swrl:BuiltinAtom>
          </rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </swrl:AtomList>
```

```
    </rdf:rest>
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
      <swrl:arguments>
       <rdf:List>
        <rdf:first rdf:resource="#graduatedStudentsAmount"/>
        <rdf:rest>
         <rdf:List>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >10</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </rdf:List>
        </rdf:rest>
       </rdf:List>
      </swrl:arguments>
     </swrl:BuiltinAtom>
    </rdf:first>
   </swrl:AtomList>
  </swrl:head>
  <swrl:body>
   <swrl:AtomList>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:arguments>
         <rdf:List>
          <rdf:rest>
           <rdf:List>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >5000</rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#salaryValue"/>
         </rdf:List>
        </swrl:arguments>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:arguments>
           <rdf:List>
            <rdf:first rdf:resource="#salaryValue"/>
            <rdf:rest>
             <rdf:List>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >3050</rdf:first>
             </rdf:List>
            </rdf:rest>
           </rdf:List>
          </swrl:arguments>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
         </swrl:BuiltinAtom>
        </rdf:first>
        <rdf:rest>
         <swrl:AtomList>
          <rdf:first>
```

```
              <swrl:DatavaluedPropertyAtom>
                <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
                <swrl:argument1 rdf:resource="#graduatedStudents"/>
                <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
              </swrl:DatavaluedPropertyAtom>
             </rdf:first>
             <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            </swrl:AtomList>
           </rdf:rest>
          </swrl:AtomList>
         </rdf:rest>
        </swrl:AtomList>
       </rdf:rest>
       <rdf:first>
        <swrl:DatavaluedPropertyAtom>
         <swrl:argument1 rdf:resource="#salary"/>
         <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
         <swrl:argument2 rdf:resource="#salaryValue"/>
        </swrl:DatavaluedPropertyAtom>
       </rdf:first>
      </swrl:AtomList>
     </swrl:body>
    </swrl:Imp>
    <swrl:Imp rdf:ID="RIS13">
     <swrl:body>
      <swrl:AtomList>
       <rdf:first>
        <swrl:ClassAtom>
         <swrl:argument1 rdf:resource="#publications"/>
         <swrl:classPredicate rdf:resource="#collectionNode"/>
        </swrl:ClassAtom>
       </rdf:first>
       <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      </swrl:AtomList>
     </swrl:body>
     <swrl:head>
      <swrl:AtomList>
       <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       <rdf:first>
        <swrl:ClassAtom>
         <swrl:classPredicate rdf:resource="#leafNode"/>
         <swrl:argument1 rdf:resource="#publication.name"/>
        </swrl:ClassAtom>
       </rdf:first>
      </swrl:AtomList>
     </swrl:head>
    </swrl:Imp>
    <swrl:Imp rdf:ID="RIS39">
     <swrl:body>
      <swrl:AtomList>
       <rdf:first>
        <swrl:DatavaluedPropertyAtom>
         <swrl:argument1 rdf:resource="#salary"/>
         <swrl:argument2 rdf:resource="#salaryValue"/>
         <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
        </swrl:DatavaluedPropertyAtom>
       </rdf:first>
       <rdf:rest>
        <swrl:AtomList>
         <rdf:first>
          <swrl:BuiltinAtom>
           <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
           <swrl:arguments>
```

```
            <rdf:List>
              <rdf:rest>
                <rdf:List>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                  >5000</rdf:first>
                </rdf:List>
              </rdf:rest>
              <rdf:first rdf:resource="#salaryValue"/>
            </rdf:List>
          </swrl:arguments>
        </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
        <swrl:AtomList>
          <rdf:first>
            <swrl:BuiltinAtom>
              <swrl:arguments>
                <rdf:List>
                  <rdf:first rdf:resource="#salaryValue"/>
                  <rdf:rest>
                    <rdf:List>
                      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                      >3050</rdf:first>
                    </rdf:List>
                  </rdf:rest>
                </rdf:List>
              </swrl:arguments>
              <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
            </swrl:BuiltinAtom>
          </rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </swrl:AtomList>
      </rdf:rest>
    </swrl:AtomList>
  </rdf:rest>
</swrl:AtomList>
</swrl:body>
<swrl:head>
  <swrl:AtomList>
    <rdf:first>
      <swrl:DatavaluedPropertyAtom>
        <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
        <swrl:argument1 rdf:resource="#degree"/>
        <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >doctor</swrl:argument2>
      </swrl:DatavaluedPropertyAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
</swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS7">
<swrl:head>
  <swrl:AtomList>
    <rdf:first>
      <swrl:ClassAtom>
        <swrl:classPredicate rdf:resource="#collectionNode"/>
        <swrl:argument1 rdf:resource="#courses"/>
      </swrl:ClassAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
```

```
    </swrl:AtomList>
  </swrl:head>
  <swrl:body>
    <swrl:AtomList/>
  </swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS4">
  <swrl:body>
    <swrl:AtomList/>
  </swrl:body>
  <swrl:head>
    <swrl:AtomList>
      <rdf:first>
        <swrl:ClassAtom>
          <swrl:classPredicate rdf:resource="#leafNode"/>
          <swrl:argument1 rdf:resource="#job"/>
        </swrl:ClassAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </swrl:AtomList>
  </swrl:head>
</swrl:Imp>
<swrl:Variable rdf:ID="positionValueSet"/>
<swrl:Variable rdf:ID="ageValue"/>
<swrl:Variable rdf:ID="coursesAmount"/>
<swrl:Imp rdf:ID="RIS36">
  <swrl:head>
    <swrl:AtomList>
      <rdf:first>
        <swrl:BuiltinAtom>
          <swrl:arguments>
            <rdf:List>
              <rdf:rest>
                <rdf:List>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                  >5000</rdf:first>
                </rdf:List>
              </rdf:rest>
              <rdf:first rdf:resource="#salaryValue"/>
            </rdf:List>
          </swrl:arguments>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
        </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
        <swrl:AtomList>
          <rdf:first>
            <swrl:BuiltinAtom>
              <swrl:arguments>
                <rdf:List>
                  <rdf:first rdf:resource="#salaryValue"/>
                  <rdf:rest>
                    <rdf:List>
                      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                      >3050</rdf:first>
                    </rdf:List>
                  </rdf:rest>
                </rdf:List>
              </swrl:arguments>
              <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
            </swrl:BuiltinAtom>
```

```
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     </swrl:AtomList>
    </rdf:rest>
   </swrl:AtomList>
</swrl:head>
<swrl:body>
  <swrl:AtomList>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:arguments>
         <rdf:List>
          <rdf:rest>
           <rdf:List>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >10</rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#graduatedStudentsAmount"/>
         </rdf:List>
        </swrl:arguments>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
          <swrl:arguments>
           <rdf:List>
            <rdf:first rdf:resource="#graduatedStudentsAmount"/>
            <rdf:rest>
             <rdf:List>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >5</rdf:first>
             </rdf:List>
            </rdf:rest>
           </rdf:List>
          </swrl:arguments>
         </swrl:BuiltinAtom>
        </rdf:first>
        <rdf:rest>
         <swrl:AtomList>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first>
           <swrl:DatavaluedPropertyAtom>
            <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
            <swrl:argument2 rdf:resource="#salaryValue"/>
            <swrl:argument1 rdf:resource="#salary"/>
           </swrl:DatavaluedPropertyAtom>
          </rdf:first>
         </swrl:AtomList>
        </rdf:rest>
       </swrl:AtomList>
      </rdf:rest>
     </swrl:AtomList>
    </rdf:rest>
    <rdf:first>
```

```
      <swrl:DatavaluedPropertyAtom>
       <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
       <swrl:argument1 rdf:resource="#graduatedStudents"/>
       <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
      </swrl:DatavaluedPropertyAtom>
     </rdf:first>
    </swrl:AtomList>
   </swrl:body>
  </swrl:Imp>
  <swrl:Imp rdf:ID="RIS26">
   <swrl:body>
    <swrl:AtomList>
     <rdf:rest>
      <swrl:AtomList>
       <rdf:rest>
        <swrl:AtomList>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         <rdf:first>
          <swrl:BuiltinAtom>
           <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
           <swrl:arguments>
            <rdf:List>
             <rdf:rest>
              <rdf:List>
               <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
               >1</rdf:first>
               <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              </rdf:List>
             </rdf:rest>
             <rdf:first rdf:resource="#coursesAmount"/>
            </rdf:List>
           </swrl:arguments>
          </swrl:BuiltinAtom>
         </rdf:first>
        </swrl:AtomList>
       </rdf:rest>
       <rdf:first>
        <swrl:BuiltinAtom>
         <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
         <swrl:arguments>
          <rdf:List>
           <rdf:first rdf:resource="#coursesAmount"/>
           <rdf:rest>
            <rdf:List>
             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
             >3</rdf:first>
             <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            </rdf:List>
           </rdf:rest>
          </rdf:List>
         </swrl:arguments>
        </swrl:BuiltinAtom>
       </rdf:first>
      </swrl:AtomList>
     </rdf:rest>
     <rdf:first>
      <swrl:DatavaluedPropertyAtom>
       <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
       <swrl:argument1 rdf:resource="#courses"/>
       <swrl:argument2 rdf:resource="#coursesAmount"/>
      </swrl:DatavaluedPropertyAtom>
     </rdf:first>
    </swrl:AtomList>
```

```
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   <rdf:first>
    <swrl:DatavaluedPropertyAtom>
     <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >doctor</swrl:argument2>
     <swrl:argument1 rdf:resource="#degree"/>
     <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
    </swrl:DatavaluedPropertyAtom>
   </rdf:first>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS23">
 <swrl:head>
  <swrl:AtomList>
   <rdf:first>
    <swrl:BuiltinAtom>
     <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     <swrl:arguments>
      <rdf:List>
       <rdf:rest>
        <rdf:List>
         <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
         >6</rdf:first>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:List>
       </rdf:rest>
       <rdf:first>
        <swrl:Variable rdf:ID="masterStudentsValue"/>
       </rdf:first>
      </rdf:List>
     </swrl:arguments>
    </swrl:BuiltinAtom>
   </rdf:first>
   <rdf:rest>
    <swrl:AtomList>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
       <swrl:arguments>
        <rdf:List>
         <rdf:first rdf:resource="#masterStudentsValue"/>
         <rdf:rest>
          <rdf:List>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
           >0</rdf:first>
          </rdf:List>
         </rdf:rest>
        </rdf:List>
       </swrl:arguments>
      </swrl:BuiltinAtom>
     </rdf:first>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </swrl:AtomList>
   </rdf:rest>
  </swrl:AtomList>
 </swrl:head>
 <swrl:body>
  <swrl:AtomList>
```

```
      <rdf:first>
        <swrl:DatavaluedPropertyAtom>
          <swrl:argument2 rdf:resource="#masterStudentsValue"/>
          <swrl:argument1 rdf:resource="#masterStudents"/>
          <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
        </swrl:DatavaluedPropertyAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </swrl:AtomList>
  </swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS18">
  <swrl:head>
    <swrl:AtomList>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      <rdf:first>
        <swrl:DatavaluedPropertyAtom>
          <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Universitary professor</swrl:argument2>
          <swrl:argument1 rdf:resource="#job"/>
          <swrl:propertyPredicate rdf:resource="#hasConstantValue"/>
        </swrl:DatavaluedPropertyAtom>
      </rdf:first>
    </swrl:AtomList>
  </swrl:head>
  <swrl:body>
    <swrl:AtomList>
      <rdf:first>
        <swrl:ClassAtom>
          <swrl:classPredicate rdf:resource="#leafNode"/>
          <swrl:argument1 rdf:resource="#job"/>
        </swrl:ClassAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </swrl:AtomList>
  </swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS37">
  <swrl:head>
    <swrl:AtomList>
      <rdf:first>
        <swrl:DatavaluedPropertyAtom>
          <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
          <swrl:argument1 rdf:resource="#degree"/>
          <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >doctor</swrl:argument2>
        </swrl:DatavaluedPropertyAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </swrl:AtomList>
  </swrl:head>
  <swrl:body>
    <swrl:AtomList>
      <rdf:first>
        <swrl:DatavaluedPropertyAtom>
          <swrl:argument1 rdf:resource="#graduatedStudents"/>
          <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
          <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
        </swrl:DatavaluedPropertyAtom>
      </rdf:first>
      <rdf:rest>
        <swrl:AtomList>
          <rdf:rest>
```

```
        <swrl:AtomList>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first>
            <swrl:BuiltinAtom>
              <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
              <swrl:arguments>
                <rdf:List>
                  <rdf:rest>
                    <rdf:List>
                      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                      >5</rdf:first>
                      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                    </rdf:List>
                  </rdf:rest>
                  <rdf:first rdf:resource="#graduatedStudentsAmount"/>
                </rdf:List>
              </swrl:arguments>
            </swrl:BuiltinAtom>
          </rdf:first>
        </swrl:AtomList>
      </rdf:rest>
      <rdf:first>
        <swrl:BuiltinAtom>
          <swrl:arguments>
            <rdf:List>
              <rdf:first rdf:resource="#graduatedStudentsAmount"/>
              <rdf:rest>
                <rdf:List>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                  >10</rdf:first>
                </rdf:List>
              </rdf:rest>
            </rdf:List>
          </swrl:arguments>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
        </swrl:BuiltinAtom>
      </rdf:first>
    </swrl:AtomList>
  </rdf:rest>
 </swrl:AtomList>
</swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS27">
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest>
    <swrl:AtomList>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:arguments>
        <rdf:List>
         <rdf:first rdf:resource="#graduatedStudentsAmount"/>
         <rdf:rest>
          <rdf:List>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
           >5</rdf:first>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          </rdf:List>
         </rdf:rest>
        </rdf:List>
       </swrl:arguments>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
```

```
        </swrl:BuiltinAtom>
       </rdf:first>
       <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      </swrl:AtomList>
     </rdf:rest>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
       <swrl:arguments>
        <rdf:List>
         <rdf:first rdf:resource="#graduatedStudentsAmount"/>
         <rdf:rest>
          <rdf:List>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
           >10</rdf:first>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          </rdf:List>
         </rdf:rest>
        </rdf:List>
       </swrl:arguments>
      </swrl:BuiltinAtom>
     </rdf:first>
    </swrl:AtomList>
  </swrl:head>
  <swrl:body>
   <swrl:AtomList>
    <rdf:first>
     <swrl:DatavaluedPropertyAtom>
      <swrl:argument2 rdf:resource="#coursesAmount"/>
      <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
      <swrl:argument1 rdf:resource="#courses"/>
     </swrl:DatavaluedPropertyAtom>
    </rdf:first>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:rest>
         <swrl:AtomList>
          <rdf:first>
           <swrl:DatavaluedPropertyAtom>
            <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
            <swrl:argument1 rdf:resource="#graduatedStudents"/>
            <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
           </swrl:DatavaluedPropertyAtom>
          </rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </swrl:AtomList>
        </rdf:rest>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:arguments>
           <rdf:List>
            <rdf:rest>
             <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >1</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#coursesAmount"/>
           </rdf:List>
          </swrl:arguments>
```

```
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
           </swrl:BuiltinAtom>
          </rdf:first>
         </swrl:AtomList>
        </rdf:rest>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
          <swrl:arguments>
           <rdf:List>
            <rdf:first rdf:resource="#coursesAmount"/>
            <rdf:rest>
             <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >3</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             </rdf:List>
            </rdf:rest>
           </rdf:List>
          </swrl:arguments>
         </swrl:BuiltinAtom>
        </rdf:first>
       </swrl:AtomList>
      </rdf:rest>
     </swrl:AtomList>
    </swrl:body>
   </swrl:Imp>
   <swrl:Imp rdf:ID="RIS14">
    <swrl:head>
     <swrl:AtomList>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      <rdf:first>
       <swrl:ClassAtom>
        <swrl:argument1 rdf:resource="#publication.conference"/>
        <swrl:classPredicate rdf:resource="#leafNode"/>
       </swrl:ClassAtom>
      </rdf:first>
     </swrl:AtomList>
    </swrl:head>
    <swrl:body>
     <swrl:AtomList>
      <rdf:first>
       <swrl:ClassAtom>
        <swrl:classPredicate rdf:resource="#collectionNode"/>
        <swrl:argument1 rdf:resource="#publications"/>
       </swrl:ClassAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     </swrl:AtomList>
    </swrl:body>
   </swrl:Imp>
   <swrl:Imp rdf:ID="RIS12">
    <swrl:body>
     <swrl:AtomList>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      <rdf:first>
       <swrl:ClassAtom>
        <swrl:classPredicate rdf:resource="#collectionNode"/>
        <swrl:argument1 rdf:resource="#graduatedStudents"/>
       </swrl:ClassAtom>
      </rdf:first>
     </swrl:AtomList>
    </swrl:body>
```

```
  <swrl:head>
   <swrl:AtomList>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    <rdf:first>
     <swrl:ClassAtom>
      <swrl:argument1 rdf:resource="#graduatedStudent.name"/>
      <swrl:classPredicate rdf:resource="#leafNode"/>
     </swrl:ClassAtom>
    </rdf:first>
   </swrl:AtomList>
  </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS6">
  <swrl:head>
   <swrl:AtomList>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    <rdf:first>
     <swrl:ClassAtom>
      <swrl:classPredicate rdf:resource="#leafNode"/>
      <swrl:argument1 rdf:resource="#completeName.lastName"/>
     </swrl:ClassAtom>
    </rdf:first>
   </swrl:AtomList>
  </swrl:head>
  <swrl:body>
   <swrl:AtomList/>
  </swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS41">
  <swrl:body>
   <swrl:AtomList>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
        <swrl:arguments>
         <rdf:List>
          <rdf:first rdf:resource="#salaryValue"/>
          <rdf:rest>
           <rdf:List>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >5000</rdf:first>
           </rdf:List>
          </rdf:rest>
         </rdf:List>
        </swrl:arguments>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:rest>
         <swrl:AtomList>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first>
           <swrl:DatavaluedPropertyAtom>
            <swrl:argument2>
             <swrl:Variable rdf:ID="masterStudentsAmount"/>
            </swrl:argument2>
            <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
            <swrl:argument1 rdf:resource="#masterStudents"/>
           </swrl:DatavaluedPropertyAtom>
```

```
            </rdf:first>
           </swrl:AtomList>
          </rdf:rest>
          <rdf:first>
           <swrl:BuiltinAtom>
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
            <swrl:arguments>
             <rdf:List>
              <rdf:first rdf:resource="#salaryValue"/>
              <rdf:rest>
               <rdf:List>
                <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >3050</rdf:first>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
               </rdf:List>
              </rdf:rest>
             </rdf:List>
            </swrl:arguments>
           </swrl:BuiltinAtom>
          </rdf:first>
         </swrl:AtomList>
        </rdf:rest>
       </swrl:AtomList>
      </rdf:rest>
      <rdf:first>
       <swrl:DatavaluedPropertyAtom>
        <swrl:argument2 rdf:resource="#salaryValue"/>
        <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
        <swrl:argument1 rdf:resource="#salary"/>
       </swrl:DatavaluedPropertyAtom>
      </rdf:first>
     </swrl:AtomList>
    </swrl:body>
    <swrl:head>
     <swrl:AtomList>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
          <swrl:arguments>
           <rdf:List>
            <rdf:first rdf:resource="#masterStudentsAmount"/>
            <rdf:rest>
             <rdf:List>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >3</rdf:first>
             </rdf:List>
            </rdf:rest>
           </rdf:List>
          </swrl:arguments>
         </swrl:BuiltinAtom>
        </rdf:first>
       </swrl:AtomList>
      </rdf:rest>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
        <swrl:arguments>
         <rdf:List>
          <rdf:first rdf:resource="#masterStudentsAmount"/>
```

```
          <rdf:rest>
           <rdf:List>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >6</rdf:first>
           </rdf:List>
          </rdf:rest>
         </rdf:List>
        </swrl:arguments>
       </swrl:BuiltinAtom>
      </rdf:first>
    </swrl:AtomList>
  </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS31">
  <swrl:body>
    <swrl:AtomList>
      <rdf:first>
        <swrl:DatavaluedPropertyAtom>
          <swrl:argument1 rdf:resource="#degree"/>
          <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
          <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >doctor</swrl:argument2>
        </swrl:DatavaluedPropertyAtom>
      </rdf:first>
      <rdf:rest>
        <swrl:AtomList>
          <rdf:first>
            <swrl:DatavaluedPropertyAtom>
              <swrl:argument1 rdf:resource="#graduatedStudents"/>
              <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
              <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
            </swrl:DatavaluedPropertyAtom>
          </rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </swrl:AtomList>
      </rdf:rest>
    </swrl:AtomList>
  </swrl:body>
  <swrl:head>
    <swrl:AtomList>
      <rdf:rest>
        <swrl:AtomList>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first>
            <swrl:BuiltinAtom>
              <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
              <swrl:arguments>
                <rdf:List>
                  <rdf:rest>
                    <rdf:List>
                      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                      >5</rdf:first>
                      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                    </rdf:List>
                  </rdf:rest>
                  <rdf:first rdf:resource="#graduatedStudentsAmount"/>
                </rdf:List>
              </swrl:arguments>
            </swrl:BuiltinAtom>
          </rdf:first>
        </swrl:AtomList>
      </rdf:rest>
```

```
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:arguments>
       <rdf:List>
        <rdf:first rdf:resource="#graduatedStudentsAmount"/>
        <rdf:rest>
         <rdf:List>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >10</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </rdf:List>
        </rdf:rest>
       </rdf:List>
      </swrl:arguments>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     </swrl:BuiltinAtom>
    </rdf:first>
   </swrl:AtomList>
  </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS21">
  <swrl:head>
   <swrl:AtomList>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
        <swrl:arguments>
         <rdf:List>
          <rdf:rest>
           <rdf:List>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >1000</rdf:first>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#salaryValue"/>
         </rdf:List>
        </swrl:arguments>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     </swrl:AtomList>
    </rdf:rest>
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:arguments>
       <rdf:List>
        <rdf:rest>
         <rdf:List>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >5000</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </rdf:List>
        </rdf:rest>
        <rdf:first rdf:resource="#salaryValue"/>
       </rdf:List>
      </swrl:arguments>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     </swrl:BuiltinAtom>
    </rdf:first>
   </swrl:AtomList>
```

```
  </swrl:head>
  <swrl:body>
   <swrl:AtomList>
    <rdf:first>
     <swrl:DatavaluedPropertyAtom>
      <swrl:argument2 rdf:resource="#salaryValue"/>
      <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
      <swrl:argument1 rdf:resource="#salary"/>
     </swrl:DatavaluedPropertyAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   </swrl:AtomList>
  </swrl:body>
 </swrl:Imp>
 <swrl:Imp rdf:ID="RIS3">
  <swrl:head>
   <swrl:AtomList>
    <rdf:first>
     <swrl:ClassAtom>
      <swrl:argument1 rdf:resource="#salary"/>
      <swrl:classPredicate rdf:resource="#leafNode"/>
     </swrl:ClassAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   </swrl:AtomList>
  </swrl:head>
  <swrl:body>
   <swrl:AtomList/>
  </swrl:body>
 </swrl:Imp>
 <swrl:Variable rdf:ID="publicationConferenceQualisValue"/>
 <swrl:Imp rdf:ID="RIS19">
  <swrl:body>
   <swrl:AtomList>
    <rdf:first>
     <swrl:ClassAtom>
      <swrl:classPredicate rdf:resource="#collectionNode"/>
      <swrl:argument1 rdf:resource="#researchLines"/>
     </swrl:ClassAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   </swrl:AtomList>
  </swrl:body>
  <swrl:head>
   <swrl:AtomList>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    <rdf:first>
     <swrl:DatavaluedPropertyAtom>
      <swrl:propertyPredicate rdf:resource="#hasConstantValue"/>
      <swrl:argument1 rdf:resource="#researchLines"/>
      <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >2</swrl:argument2>
     </swrl:DatavaluedPropertyAtom>
    </rdf:first>
   </swrl:AtomList>
  </swrl:head>
 </swrl:Imp>
 <swrl:Imp rdf:ID="RIS28">
  <swrl:head>
   <swrl:AtomList>
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:arguments>
```

```
        <rdf:List>
          <rdf:rest>
            <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >6</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#masterStudentsAmount"/>
        </rdf:List>
      </swrl:arguments>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
    </swrl:BuiltinAtom>
  </rdf:first>
  <rdf:rest>
    <swrl:AtomList>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      <rdf:first>
        <swrl:BuiltinAtom>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
          <swrl:arguments>
            <rdf:List>
              <rdf:rest>
                <rdf:List>
                  <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                  >3</rdf:first>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                </rdf:List>
              </rdf:rest>
              <rdf:first rdf:resource="#masterStudentsAmount"/>
            </rdf:List>
          </swrl:arguments>
        </swrl:BuiltinAtom>
      </rdf:first>
    </swrl:AtomList>
  </rdf:rest>
</swrl:AtomList>
</swrl:head>
<swrl:body>
  <swrl:AtomList>
    <rdf:rest>
      <swrl:AtomList>
        <rdf:rest>
          <swrl:AtomList>
            <rdf:rest>
              <swrl:AtomList>
                <rdf:first>
                  <swrl:DatavaluedPropertyAtom>
                    <swrl:argument2 rdf:resource="#masterStudentsAmount"/>
                    <swrl:argument1 rdf:resource="#masterStudents"/>
                    <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
                  </swrl:DatavaluedPropertyAtom>
                </rdf:first>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              </swrl:AtomList>
            </rdf:rest>
            <rdf:first>
              <swrl:BuiltinAtom>
                <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
                <swrl:arguments>
                  <rdf:List>
                    <rdf:rest>
                      <rdf:List>
```

```
                    <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                    >1</rdf:first>
                    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                   </rdf:List>
                  </rdf:rest>
                  <rdf:first rdf:resource="#coursesAmount"/>
                 </rdf:List>
                </swrl:arguments>
               </swrl:BuiltinAtom>
              </rdf:first>
             </swrl:AtomList>
            </rdf:rest>
            <rdf:first>
             <swrl:BuiltinAtom>
              <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
              <swrl:arguments>
               <rdf:List>
                <rdf:rest>
                 <rdf:List>
                  <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                  >3</rdf:first>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                 </rdf:List>
                </rdf:rest>
                <rdf:first rdf:resource="#coursesAmount"/>
               </rdf:List>
              </swrl:arguments>
             </swrl:BuiltinAtom>
            </rdf:first>
           </swrl:AtomList>
          </rdf:rest>
          <rdf:first>
           <swrl:DatavaluedPropertyAtom>
            <swrl:argument2 rdf:resource="#coursesAmount"/>
            <swrl:argument1 rdf:resource="#courses"/>
            <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
           </swrl:DatavaluedPropertyAtom>
          </rdf:first>
         </swrl:AtomList>
        </swrl:body>
       </swrl:Imp>
       <swrl:Imp rdf:ID="RIS34">
        <swrl:head>
         <swrl:AtomList>
          <rdf:first>
           <swrl:BuiltinAtom>
            <swrl:arguments>
             <rdf:List>
              <rdf:rest>
               <rdf:List>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >3</rdf:first>
               </rdf:List>
              </rdf:rest>
              <rdf:first rdf:resource="#coursesAmount"/>
             </rdf:List>
            </swrl:arguments>
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
           </swrl:BuiltinAtom>
          </rdf:first>
          <rdf:rest>
           <swrl:AtomList>
```

```
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:arguments>
         <rdf:List>
          <rdf:first rdf:resource="#coursesAmount"/>
          <rdf:rest>
           <rdf:List>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >1</rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           </rdf:List>
          </rdf:rest>
         </rdf:List>
        </swrl:arguments>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     </swrl:AtomList>
    </rdf:rest>
   </swrl:AtomList>
  </swrl:head>
  <swrl:body>
   <swrl:AtomList>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:arguments>
         <rdf:List>
          <rdf:rest>
           <rdf:List>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >10</rdf:first>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#graduatedStudentsAmount"/>
         </rdf:List>
        </swrl:arguments>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:arguments>
           <rdf:List>
            <rdf:rest>
             <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >5</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#graduatedStudentsAmount"/>
           </rdf:List>
          </swrl:arguments>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
         </swrl:BuiltinAtom>
        </rdf:first>
        <rdf:rest>
```

```
            <swrl:AtomList>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              <rdf:first>
                <swrl:DatavaluedPropertyAtom>
                  <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
                  <swrl:argument2 rdf:resource="#coursesAmount"/>
                  <swrl:argument1 rdf:resource="#courses"/>
                </swrl:DatavaluedPropertyAtom>
              </rdf:first>
            </swrl:AtomList>
          </rdf:rest>
        </swrl:AtomList>
      </rdf:rest>
    </swrl:AtomList>
  </rdf:rest>
  <rdf:first>
    <swrl:DatavaluedPropertyAtom>
      <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
      <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
      <swrl:argument1 rdf:resource="#graduatedStudents"/>
    </swrl:DatavaluedPropertyAtom>
  </rdf:first>
 </swrl:AtomList>
</swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS38">
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest>
    <swrl:AtomList>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
       <swrl:arguments>
        <rdf:List>
         <rdf:rest>
          <rdf:List>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
           >1</rdf:first>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          </rdf:List>
         </rdf:rest>
         <rdf:first rdf:resource="#coursesAmount"/>
        </rdf:List>
       </swrl:arguments>
      </swrl:BuiltinAtom>
     </rdf:first>
    </swrl:AtomList>
   </rdf:rest>
   <rdf:first>
    <swrl:BuiltinAtom>
     <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
     <swrl:arguments>
      <rdf:List>
       <rdf:first rdf:resource="#coursesAmount"/>
       <rdf:rest>
        <rdf:List>
         <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
         >3</rdf:first>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:List>
       </rdf:rest>
```

```
        </rdf:List>
       </swrl:arguments>
      </swrl:BuiltinAtom>
    </rdf:first>
  </swrl:AtomList>
</swrl:head>
<swrl:body>
  <swrl:AtomList>
    <rdf:first>
      <swrl:DatavaluedPropertyAtom>
        <swrl:argument1 rdf:resource="#salary"/>
        <swrl:argument2 rdf:resource="#salaryValue"/>
        <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
      </swrl:DatavaluedPropertyAtom>
    </rdf:first>
    <rdf:rest>
      <swrl:AtomList>
        <rdf:first>
          <swrl:BuiltinAtom>
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
            <swrl:arguments>
              <rdf:List>
                <rdf:rest>
                  <rdf:List>
                    <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                    >5000</rdf:first>
                    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  </rdf:List>
                </rdf:rest>
                <rdf:first rdf:resource="#salaryValue"/>
              </rdf:List>
            </swrl:arguments>
          </swrl:BuiltinAtom>
        </rdf:first>
        <rdf:rest>
          <swrl:AtomList>
            <rdf:first>
              <swrl:BuiltinAtom>
                <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
                <swrl:arguments>
                  <rdf:List>
                    <rdf:rest>
                      <rdf:List>
                        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                        >3050</rdf:first>
                        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                      </rdf:List>
                    </rdf:rest>
                    <rdf:first rdf:resource="#salaryValue"/>
                  </rdf:List>
                </swrl:arguments>
              </swrl:BuiltinAtom>
            </rdf:first>
            <rdf:rest>
              <swrl:AtomList>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                <rdf:first>
                  <swrl:DatavaluedPropertyAtom>
                    <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
                    <swrl:argument1 rdf:resource="#courses"/>
                    <swrl:argument2 rdf:resource="#coursesAmount"/>
                  </swrl:DatavaluedPropertyAtom>
                </rdf:first>
```

```
            </swrl:AtomList>
           </rdf:rest>
          </swrl:AtomList>
         </rdf:rest>
        </swrl:AtomList>
       </rdf:rest>
      </swrl:AtomList>
     </swrl:body>
    </swrl:Imp>
    <swrl:Imp rdf:ID="RIS17">
     <swrl:body>
      <swrl:AtomList>
       <rdf:rest>
        <swrl:AtomList>
         <rdf:first>
          <swrl:BuiltinAtom>
           <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#listConcat"/>
           <swrl:arguments>
            <rdf:List>
             <rdf:rest>
              <rdf:List>
               <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
               >A</rdf:first>
               <rdf:rest>
                <rdf:List>
                 <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                 >B</rdf:first>
                 <rdf:rest>
                  <rdf:List>
                   <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                   >C</rdf:first>
                   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  </rdf:List>
                 </rdf:rest>
                </rdf:List>
               </rdf:rest>
              </rdf:List>
             </rdf:rest>
             <rdf:first rdf:resource="#positionValueSet"/>
            </rdf:List>
           </swrl:arguments>
          </swrl:BuiltinAtom>
         </rdf:first>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </swrl:AtomList>
       </rdf:rest>
       <rdf:first>
        <swrl:DatavaluedPropertyAtom>
         <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
         <swrl:argument1 rdf:resource="#position"/>
         <swrl:argument2>
          <swrl:Variable rdf:ID="positionValue"/>
         </swrl:argument2>
        </swrl:DatavaluedPropertyAtom>
       </rdf:first>
      </swrl:AtomList>
     </swrl:body>
     <swrl:head>
      <swrl:AtomList>
       <rdf:first>
        <swrl:BuiltinAtom>
         <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#member"/>
         <swrl:arguments>
```

```
          <rdf:List>
            <rdf:rest>
              <rdf:List>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                <rdf:first rdf:resource="#positionValueSet"/>
              </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#positionValue"/>
          </rdf:List>
        </swrl:arguments>
      </swrl:BuiltinAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS9">
 <swrl:body>
  <swrl:AtomList>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   <rdf:first>
    <swrl:ClassAtom>
     <swrl:classPredicate rdf:resource="#collectionNode"/>
     <swrl:argument1 rdf:resource="#researchLines"/>
    </swrl:ClassAtom>
   </rdf:first>
  </swrl:AtomList>
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
   <rdf:first>
    <swrl:ClassAtom>
     <swrl:argument1 rdf:resource="#researchLine.name"/>
     <swrl:classPredicate rdf:resource="#leafNode"/>
    </swrl:ClassAtom>
   </rdf:first>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS2">
 <swrl:body>
  <swrl:AtomList/>
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
   <rdf:first>
    <swrl:ClassAtom>
     <swrl:classPredicate rdf:resource="#leafNode"/>
     <swrl:argument1 rdf:resource="#age"/>
    </swrl:ClassAtom>
   </rdf:first>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS24">
 <swrl:body>
  <swrl:AtomList>
   <rdf:first>
    <swrl:DatavaluedPropertyAtom>
     <swrl:argument1 rdf:resource="#graduatedStudents"/>
     <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
```

```
        <swrl:argument2>
          <swrl:Variable rdf:ID="graduatedStudentsValue"/>
        </swrl:argument2>
      </swrl:DatavaluedPropertyAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
    <rdf:first>
      <swrl:BuiltinAtom>
        <swrl:arguments>
          <rdf:List>
            <rdf:rest>
              <rdf:List>
                <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >10</rdf:first>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#graduatedStudentsValue"/>
          </rdf:List>
        </swrl:arguments>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
      </swrl:BuiltinAtom>
    </rdf:first>
    <rdf:rest>
      <swrl:AtomList>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        <rdf:first>
          <swrl:BuiltinAtom>
            <swrl:arguments>
              <rdf:List>
                <rdf:rest>
                  <rdf:List>
                    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                    <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                    >0</rdf:first>
                  </rdf:List>
                </rdf:rest>
                <rdf:first rdf:resource="#graduatedStudentsValue"/>
              </rdf:List>
            </swrl:arguments>
            <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
          </swrl:BuiltinAtom>
        </rdf:first>
      </swrl:AtomList>
    </rdf:rest>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS20">
 <swrl:head>
  <swrl:AtomList>
    <rdf:first>
      <swrl:BuiltinAtom>
        <swrl:arguments>
          <rdf:List>
            <rdf:first rdf:resource="#ageValue"/>
            <rdf:rest>
              <rdf:List>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
```

```
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >60</rdf:first>
           </rdf:List>
          </rdf:rest>
         </rdf:List>
        </swrl:arguments>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:arguments>
           <rdf:List>
            <rdf:rest>
             <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >30</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#ageValue"/>
           </rdf:List>
          </swrl:arguments>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
         </swrl:BuiltinAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       </swrl:AtomList>
      </rdf:rest>
     </swrl:AtomList>
    </swrl:head>
    <swrl:body>
     <swrl:AtomList>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      <rdf:first>
       <swrl:DatavaluedPropertyAtom>
        <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
        <swrl:argument2 rdf:resource="#ageValue"/>
        <swrl:argument1 rdf:resource="#age"/>
       </swrl:DatavaluedPropertyAtom>
      </rdf:first>
     </swrl:AtomList>
    </swrl:body>
   </swrl:Imp>
   <swrl:Imp rdf:ID="RIS11">
    <swrl:body>
     <swrl:AtomList>
      <rdf:first>
       <swrl:ClassAtom>
        <swrl:argument1 rdf:resource="#masterStudents"/>
        <swrl:classPredicate rdf:resource="#collectionNode"/>
       </swrl:ClassAtom>
      </rdf:first>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     </swrl:AtomList>
    </swrl:body>
    <swrl:head>
     <swrl:AtomList>
      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      <rdf:first>
       <swrl:ClassAtom>
```

```
        <swrl:classPredicate rdf:resource="#leafNode"/>
        <swrl:argument1 rdf:resource="#masterStudent.name"/>
       </swrl:ClassAtom>
     </rdf:first>
    </swrl:AtomList>
  </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS30">
  <swrl:head>
    <swrl:AtomList>
      <rdf:first>
        <swrl:BuiltinAtom>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
          <swrl:arguments>
            <rdf:List>
              <rdf:first rdf:resource="#coursesAmount"/>
              <rdf:rest>
                <rdf:List>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                  <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                  >3</rdf:first>
                </rdf:List>
              </rdf:rest>
            </rdf:List>
          </swrl:arguments>
        </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
        <swrl:AtomList>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first>
            <swrl:BuiltinAtom>
              <swrl:arguments>
                <rdf:List>
                  <rdf:first rdf:resource="#coursesAmount"/>
                  <rdf:rest>
                    <rdf:List>
                      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                      >1</rdf:first>
                    </rdf:List>
                  </rdf:rest>
                </rdf:List>
              </swrl:arguments>
              <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
            </swrl:BuiltinAtom>
          </rdf:first>
        </swrl:AtomList>
      </rdf:rest>
    </swrl:AtomList>
  </swrl:head>
  <swrl:body>
    <swrl:AtomList>
      <rdf:first>
        <swrl:DatavaluedPropertyAtom>
          <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
          <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >doctor</swrl:argument2>
          <swrl:argument1 rdf:resource="#degree"/>
        </swrl:DatavaluedPropertyAtom>
      </rdf:first>
      <rdf:rest>
        <swrl:AtomList>
```

```
        <rdf:first>
          <swrl:DatavaluedPropertyAtom>
            <swrl:argument1 rdf:resource="#courses"/>
            <swrl:argument2 rdf:resource="#coursesAmount"/>
            <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
          </swrl:DatavaluedPropertyAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      </swrl:AtomList>
    </rdf:rest>
  </swrl:AtomList>
 </swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS5">
 <swrl:body>
  <swrl:AtomList/>
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   <rdf:first>
    <swrl:ClassAtom>
     <swrl:classPredicate rdf:resource="#leafNode"/>
     <swrl:argument1 rdf:resource="#completeName.firstName"/>
    </swrl:ClassAtom>
   </rdf:first>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS32">
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest>
    <swrl:AtomList>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
       <swrl:arguments>
        <rdf:List>
         <rdf:first rdf:resource="#masterStudentsAmount"/>
         <rdf:rest>
          <rdf:List>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
           >3</rdf:first>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          </rdf:List>
         </rdf:rest>
        </rdf:List>
       </swrl:arguments>
      </swrl:BuiltinAtom>
     </rdf:first>
    </swrl:AtomList>
   </rdf:rest>
   <rdf:first>
    <swrl:BuiltinAtom>
     <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     <swrl:arguments>
      <rdf:List>
       <rdf:rest>
        <rdf:List>
         <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
         >6</rdf:first>
```

```
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#masterStudentsAmount"/>
         </rdf:List>
        </swrl:arguments>
       </swrl:BuiltinAtom>
      </rdf:first>
     </swrl:AtomList>
    </swrl:head>
    <swrl:body>
     <swrl:AtomList>
      <rdf:first>
       <swrl:DatavaluedPropertyAtom>
        <swrl:argument1 rdf:resource="#degree"/>
        <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
        <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >doctor</swrl:argument2>
       </swrl:DatavaluedPropertyAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:first>
         <swrl:DatavaluedPropertyAtom>
          <swrl:argument2 rdf:resource="#masterStudentsAmount"/>
          <swrl:argument1 rdf:resource="#masterStudents"/>
          <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
         </swrl:DatavaluedPropertyAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       </swrl:AtomList>
      </rdf:rest>
     </swrl:AtomList>
    </swrl:body>
   </swrl:Imp>
   <swrl:Imp rdf:ID="RIS29">
    <swrl:head>
     <swrl:AtomList>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:arguments>
           <rdf:List>
            <rdf:rest>
             <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >3050</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#salaryValue"/>
           </rdf:List>
          </swrl:arguments>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
         </swrl:BuiltinAtom>
        </rdf:first>
       </swrl:AtomList>
      </rdf:rest>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
```

```
      <swrl:arguments>
       <rdf:List>
        <rdf:first rdf:resource="#salaryValue"/>
        <rdf:rest>
         <rdf:List>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >5000</rdf:first>
         </rdf:List>
        </rdf:rest>
       </rdf:List>
      </swrl:arguments>
     </swrl:BuiltinAtom>
    </rdf:first>
   </swrl:AtomList>
</swrl:head>
<swrl:body>
  <swrl:AtomList>
    <rdf:first>
     <swrl:DatavaluedPropertyAtom>
      <swrl:argument2 rdf:resource="#coursesAmount"/>
      <swrl:argument1 rdf:resource="#courses"/>
      <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
     </swrl:DatavaluedPropertyAtom>
    </rdf:first>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThan"/>
        <swrl:arguments>
         <rdf:List>
          <rdf:rest>
           <rdf:List>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >3</rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#coursesAmount"/>
         </rdf:List>
        </swrl:arguments>
       </swrl:BuiltinAtom>
      </rdf:first>
      <rdf:rest>
       <swrl:AtomList>
        <rdf:first>
         <swrl:BuiltinAtom>
          <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
          <swrl:arguments>
           <rdf:List>
            <rdf:rest>
             <rdf:List>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
              >1</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
             </rdf:List>
            </rdf:rest>
            <rdf:first rdf:resource="#coursesAmount"/>
           </rdf:List>
          </swrl:arguments>
         </swrl:BuiltinAtom>
        </rdf:first>
```

```
          <rdf:rest>
           <swrl:AtomList>
            <rdf:first>
             <swrl:DatavaluedPropertyAtom>
              <swrl:argument1 rdf:resource="#salary"/>
              <swrl:argument2 rdf:resource="#salaryValue"/>
              <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
             </swrl:DatavaluedPropertyAtom>
            </rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           </swrl:AtomList>
          </rdf:rest>
         </swrl:AtomList>
        </rdf:rest>
       </swrl:AtomList>
      </rdf:rest>
     </swrl:AtomList>
    </swrl:body>
  </swrl:Imp>
  <swrl:Imp rdf:ID="RIS16">
   <swrl:body>
    <swrl:AtomList>
     <rdf:rest>
      <swrl:AtomList>
       <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       <rdf:first>
        <swrl:BuiltinAtom>
         <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#listConcat"/>
         <swrl:arguments>
          <rdf:List>
           <rdf:first rdf:resource="#publicationConferenceQualisValueSet"/>
           <rdf:rest>
            <rdf:List>
             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
             >A</rdf:first>
             <rdf:rest>
              <rdf:List>
               <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
               >B</rdf:first>
               <rdf:rest>
                <rdf:List>
                 <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                 >C</rdf:first>
                 <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                </rdf:List>
               </rdf:rest>
              </rdf:List>
             </rdf:rest>
            </rdf:List>
           </rdf:rest>
          </rdf:List>
         </swrl:arguments>
        </swrl:BuiltinAtom>
       </rdf:first>
      </swrl:AtomList>
     </rdf:rest>
     <rdf:first>
      <swrl:DatavaluedPropertyAtom>
       <swrl:propertyPredicate rdf:resource="#hasQualifiedValue"/>
       <swrl:argument2 rdf:resource="#publicationConferenceQualisValue"/>
       <swrl:argument1 rdf:resource="#publication.conference.qualis"/>
      </swrl:DatavaluedPropertyAtom>
     </rdf:first>
```

```
  </swrl:AtomList>
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
   <rdf:first>
    <swrl:BuiltinAtom>
     <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#member"/>
     <swrl:arguments>
      <rdf:List>
       <rdf:rest>
        <rdf:List>
         <rdf:first rdf:resource="#publicationConferenceQualisValueSet"/>
         <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:List>
       </rdf:rest>
       <rdf:first rdf:resource="#publicationConferenceQualisValue"/>
      </rdf:List>
     </swrl:arguments>
    </swrl:BuiltinAtom>
   </rdf:first>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
 </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS25">
 <swrl:body>
  <swrl:AtomList>
   <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   <rdf:first>
    <swrl:DatavaluedPropertyAtom>
     <swrl:argument2>
      <swrl:Variable rdf:ID="publicationsValue"/>
     </swrl:argument2>
     <swrl:argument1 rdf:resource="#publications"/>
     <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
    </swrl:DatavaluedPropertyAtom>
   </rdf:first>
  </swrl:AtomList>
 </swrl:body>
 <swrl:head>
  <swrl:AtomList>
   <rdf:rest>
    <swrl:AtomList>
     <rdf:first>
      <swrl:BuiltinAtom>
       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
       <swrl:arguments>
        <rdf:List>
         <rdf:rest>
          <rdf:List>
           <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
           >0</rdf:first>
          </rdf:List>
         </rdf:rest>
         <rdf:first rdf:resource="#publicationsValue"/>
        </rdf:List>
       </swrl:arguments>
      </swrl:BuiltinAtom>
     </rdf:first>
     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
    </swrl:AtomList>
   </rdf:rest>
```

```
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:arguments>
       <rdf:List>
        <rdf:first rdf:resource="#publicationsValue"/>
        <rdf:rest>
         <rdf:List>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >20</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </rdf:List>
        </rdf:rest>
       </rdf:List>
      </swrl:arguments>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     </swrl:BuiltinAtom>
    </rdf:first>
   </swrl:AtomList>
  </swrl:head>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS1">
  <swrl:head>
   <swrl:AtomList>
    <rdf:first>
     <swrl:ClassAtom>
      <swrl:classPredicate rdf:resource="#leafNode"/>
      <swrl:argument1 rdf:resource="#degree"/>
     </swrl:ClassAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
   </swrl:AtomList>
  </swrl:head>
  <swrl:body>
   <swrl:AtomList/>
  </swrl:body>
</swrl:Imp>
<swrl:Imp rdf:ID="RIS35">
  <swrl:head>
   <swrl:AtomList>
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:arguments>
       <rdf:List>
        <rdf:rest>
         <rdf:List>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >6</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </rdf:List>
        </rdf:rest>
        <rdf:first rdf:resource="#masterStudentsAmount"/>
       </rdf:List>
      </swrl:arguments>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     </swrl:BuiltinAtom>
    </rdf:first>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
        <swrl:arguments>
         <rdf:List>
```

```
            <rdf:first rdf:resource="#masterStudentsAmount"/>
            <rdf:rest>
              <rdf:List>
                <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >3</rdf:first>
                <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              </rdf:List>
            </rdf:rest>
          </rdf:List>
        </swrl:arguments>
      </swrl:BuiltinAtom>
    </rdf:first>
    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  </swrl:AtomList>
 </rdf:rest>
</swrl:AtomList>
</swrl:head>
<swrl:body>
 <swrl:AtomList>
  <rdf:first>
   <swrl:DatavaluedPropertyAtom>
    <swrl:argument1 rdf:resource="#graduatedStudents"/>
    <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
    <swrl:argument2 rdf:resource="#graduatedStudentsAmount"/>
   </swrl:DatavaluedPropertyAtom>
  </rdf:first>
  <rdf:rest>
   <swrl:AtomList>
    <rdf:first>
     <swrl:BuiltinAtom>
      <swrl:arguments>
       <rdf:List>
        <rdf:rest>
         <rdf:List>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
          >10</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
         </rdf:List>
        </rdf:rest>
        <rdf:first rdf:resource="#graduatedStudentsAmount"/>
       </rdf:List>
      </swrl:arguments>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual"/>
     </swrl:BuiltinAtom>
    </rdf:first>
    <rdf:rest>
     <swrl:AtomList>
      <rdf:first>
       <swrl:BuiltinAtom>
        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
        <swrl:arguments>
         <rdf:List>
          <rdf:rest>
           <rdf:List>
            <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >5</rdf:first>
           </rdf:List>
          </rdf:rest>
          <rdf:first rdf:resource="#graduatedStudentsAmount"/>
         </rdf:List>
        </swrl:arguments>
       </swrl:BuiltinAtom>
```

```
            </rdf:first>
            <rdf:rest>
             <swrl:AtomList>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              <rdf:first>
               <swrl:DatavaluedPropertyAtom>
                <swrl:argument1 rdf:resource="#masterStudents"/>
                <swrl:argument2 rdf:resource="#masterStudentsAmount"/>
                <swrl:propertyPredicate rdf:resource="#hasQuantifiedValue"/>
               </swrl:DatavaluedPropertyAtom>
              </rdf:first>
             </swrl:AtomList>
            </rdf:rest>
           </swrl:AtomList>
          </rdf:rest>
         </swrl:AtomList>
        </rdf:rest>
       </swrl:AtomList>
      </swrl:body>
     </swrl:Imp>
     <swrl:Imp rdf:ID="RIS10">
      <swrl:body>
       <swrl:AtomList>
        <rdf:first>
         <swrl:ClassAtom>
          <swrl:argument1 rdf:resource="#courses"/>
          <swrl:classPredicate rdf:resource="#collectionNode"/>
         </swrl:ClassAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       </swrl:AtomList>
      </swrl:body>
      <swrl:head>
       <swrl:AtomList>
        <rdf:first>
         <swrl:ClassAtom>
          <swrl:argument1 rdf:resource="#course.subject"/>
          <swrl:classPredicate rdf:resource="#leafNode"/>
         </swrl:ClassAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
       </swrl:AtomList>
      </swrl:head>
     </swrl:Imp>
```