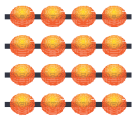
UNIVERSIDADE FEDERAL DO CEARÁ
DEPARTAMENTO DE COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO



Dissertação de Mestrado

XMLS+Matcher: Um Método para Identificação de Correspondências entre Esquemas XML Schemas Semânticos

por

Renata Maria de Figueirêdo Vilas Boas

Orientadora: Profª. Drª. Vânia Maria Ponte Vidal

Universidade Federal do Ceará
Departamento de Computação
Mestrado em Ciência da Computação

Renata Maria de Figueirêdo Vilas Boas

XMLS+Matcher: Um Método para Identificação de Correspondências entre Esquemas XML Schemas Semânticos

Este trabalho foi apresentado à Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará como requisito parcial para obtenção de grau de Mestre em Ciência da Computação.

Orientadora: Profª. Drª. Vânia Maria Ponte Vidal

AGRADECIMENTOS

Agradeço a Deus pela realização deste sonho e por colocar pessoas maravilhosas no meu caminho que me incentivaram, me deram oportunidades, me instruíram e me ajudaram a enfrentar os desafios desta jornada;

Agradeço a meus pais, João e Glaphyra, cujos ensinamentos me deram equilíbrio necessário para enfrentar, com garra, o período do mestrado;

Agradeço a meu irmão André que, apesar da distância, esteve ao meu lado em todos os momentos, assim como minhas tias Bete e Mônica;

Agradeço a meu companheiro Ruy pela compreensão, paciência, atenção e carinho para comigo em todas as horas, das mais fáceis às mais difíceis. Seu apoio diário foi imprescindível durante toda a jornada;

Agradeço a minha orientadora Profª. Vânia pelos ensinamentos, que foram de fundamental importância para esta conquista e pela sua dedicada orientação, que servirá de modelo para trabalhos futuros;

Agradeço a Profª. Ana Regina e a Profª. Káthia, que despertaram o meu interesse pela pesquisa e me ajudaram na escolha da universidade e do curso;

Agradeço a todos os meus amigos, em especial a: Fabiana, D. Zélia, Juliana, Bernadette, Roberta, Wamberg, Valdiana, Sabrina, Rosana, Orley, Andréia e Érika que, em diferentes momentos e de diferentes formas contribuíram para a realização deste trabalho;

Agradeço a meus professores pelos conhecimentos adquiridos durante o curso;

Agradeço a CAPES pelo indispensável apoio financeiro.

RESUMO

O objetivo dos sistemas de integração de dados é permitir o acesso integrado a várias fontes de informação heterogêneas e independentes, através de uma visão global (Virtual ou Materializada). Várias arquiteturas dos sistemas de integração na Web atuais são baseadas em mediadores e utilizam XML como modelo comum (*XML-Based Mediators*). Neste trabalho, usamos uma notação gráfica para representar os esquemas das fontes locais e o esquema do mediador definidos em XML Schema, denominada XML Schema Semântico (XMLS+).

Um dos aspectos mais importantes dos sistemas de integração de dados é a especificação precisa das correspondências entre os dados das várias fontes e estes na visão global (esquema do mediador). O processo que analisa dois esquemas para identificar os elementos correspondentes nos mesmos é denominado identificação de correspondências entre esquemas ou *matching* de esquemas. *Matching* de esquemas não é uma tarefa simples, principalmente, devido ao problema da heterogeneidade semântica existente entre os esquemas das fontes locais e o esquema do mediador.

Neste trabalho, propomos um método para *matching* de esquemas XMLS+, denominado XMLS+Matcher. É apresentada uma definição formal dos vários tipos de assertivas de correspondência que são utilizadas para especificar formalmente o relacionamento semântico entre elementos de esquemas XMLS+. A vantagem desse formalismo é que permite especificar de forma precisa as correspondências entre o esquema do mediador e os esquemas locais. São exatamente estas correspondências que determinarão como as consultas definidas na visão global (virtual) serão respondidas a partir de consultas nas fontes locais, e, no caso do enfoque materializado, estas correspondências serão usadas para manter as visões materializadas.

Cada passo do XMLS+Matcher é apoiado por heurísticas e regras de inferência que auxiliam na identificação das correspondências. Para isso, as heurísticas utilizam informações obtidas dos esquemas, assim como informações provenientes de consultas a dicionários semânticos.

ABSTRACT

The goal of data integration systems is to allow integrated access to multiple heterogeneous information sources, providing the user with a unified view (virtual or materialized) of the data. Several web data integration system architectures are based on mediators and use XML as the common data model (*XML-Based Mediators*). In this work, we use a graphical notation to represent the local sources' schemas and the mediated view's schema defined in XML Schema, called Semantic XML Schema (XMLS+).

One of the most important aspects in the design of data integration systems is the accurate specification of the correspondence between the data at the sources and those in the mediated schema. The process that analyzes two schemas to identify the correspondent elements between them is called identification of interschemas correspondence or schema matching. Schema matching is not an easy task, mainly, due to the problem of semantic heterogeneity that exist between local sources' schemas and the mediated's schema

In this work, we propose a method for matching XMLS+ schemas, called XMLS+Matcher. We present the formal definitions of several types of correspondence assertions, which are used to formally specify the semantic relationships among elements of XMLS+ schemas. The advantage of our formalism is that it allows us to define, in an accurate way, the correspondences between the sources and the mediated schema. It is exactly these correspondences that will determine how queries defined against the virtual mediated view will be answered based on queries formulated against the local sources, and, in case the mediated view is materialized, they are used to correctly maintain the materialized view.

Each step of the XMLS+Matcher is supported by heuristics and inference rules that aid the correspondence identification. In order to do that, the heuristics use schema information, as well as information derived from semantic dictionaries consultations.

ÍNDICE

LISTA DE TABELAS

LISTA DE FIGURAS

CAPÍTULO 1

INTRODUÇÃO

Nos últimos anos, o número de aplicações que requisitam acesso integrado a múltiplas fontes de informação heterogêneas e autônomas tem crescido bastante. Desta forma, o problema da integração de informações tem sido amplamente abordado pela literatura [CGMH94, Len02, Lós98, SP94, VL97, ZHK96]. Recentemente, com o aumento da popularidade da *World Wide Web* (Web) como um ambiente para disseminação de informações, muitos pesquisadores começaram a investigar o problema da integração de informações na Web [BGL+99, CA99, Cat99, DRRS+02, FLM98, KMAA+01, MCH02, MK99]. Assim, vários sistemas têm sido propostos e desenvolvidos para integrar múltiplas fontes Web.

Integrar informações de múltiplas fontes é uma tarefa complexa. Uma das razões para tal complexidade é a heterogeneidade na estrutura das fontes. Uma forma de minimizar a complexidade da integração, adotada pela maioria dos sistemas de integração, é definir um modelo de dados comum para representar a estrutura e conteúdo das fontes. Recentemente, XML (*eXtensible Markup Language*) [BPMM00] foi proposta pelo W3C [W3C] como um padrão para representação de informações na Web. Devido à flexibilidade da XML de representar fontes com estruturas heterogêneas e a facilidade de converter qualquer dado em XML, esta linguagem está sendo proposta, também, como um padrão para integração de informações [CCS00, GMW99]. Desta forma, pode-se fornecer uma representação uniforme e flexível dos dados de fontes arbitrárias.

Vários sistemas de integração na Web atuais são baseados em mediadores e utilizam XML como modelo comum, os quais são denominados de *XML-Based Mediators* [ABS00, BGL+99,

GSN99, OV99, VBO01, VLS01]. Estes sistemas são baseados em uma arquitetura de três níveis de esquema mostrada na Figura 1.1. O mediador suporta uma visão integrada XML e as fontes locais exportam visões XML. A visão integrada do mediador pode ser tanto virtual quanto materializada:

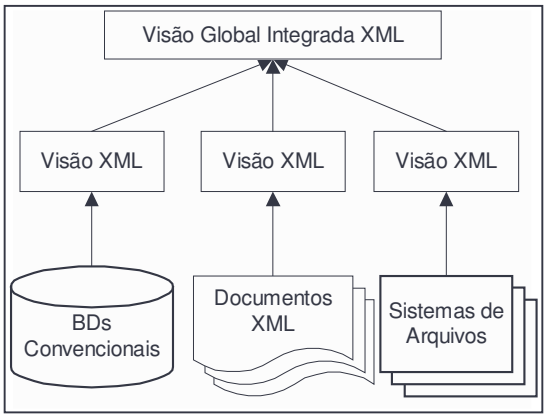


Figura 1.1: ***Arquitetura de Três Níveis de Esquema***

- No enfoque virtual, o sistema de integração utiliza um conjunto de correspondências entre o esquema da visão do mediador e os esquemas das fontes de informação, chamados *esquemas locais*, para determinar como as consultas na visão do mediador serão respondidas a partir de consultas nas fontes locais. Os resultados dessas consultas são traduzidos, filtrados e integrados, e depois, o resultado final é retornado ao usuário.

- No enfoque materializado, as informações relevantes são previamente extraídas das fontes de informação, e, posteriormente, traduzidas, filtradas, integradas e armazenadas em um repositório (também chamado de *Data Warehouse*), de maneira que as consultas possam ser avaliadas diretamente neste repositório centralizado, sem a necessidade de acessar as fontes de informação locais. Um ponto crítico do enfoque materializado é manter as visões materializadas consistentes com relação às atualizações nas fontes (Manutenção da Visão Materializada). Os algoritmos de manutenção de visões materializadas [Vid02a, Vid02b] utilizam as correspondências entre o esquema do mediador e os esquemas locais para atualizar corretamente a visão, de forma a refletir as atualizações ocorridas nas fontes.

Como vimos, as correspondências do esquema do mediador com os esquemas locais desempenham um papel central tanto nos sistemas de integração de informações que usam visão virtual, quanto naqueles que usam visão materializada. O processo de identificar correspondências entre esquemas ou *Matching* de esquemas [RB01], não é simples, principalmente, devido ao problema da heterogeneidade semântica existente entre os esquemas. Heterogeneidade semântica expressa a multiplicidade de possíveis representações de um mesmo conceito do mundo real. Este tipo heterogeneidade pode ocorrer, principalmente, porque projetistas têm diferentes percepções da realidade. Assim, um mesmo conceito do mundo real pode ser representado, por exemplo, por

diferentes nomes (heterogeneidade terminológica) e diferentes construtores de modelagem (heterogeneidade estrutural).

Neste trabalho, para facilitar o processo de análise, comparação e identificação de correspondências entre esquemas, usamos uma notação gráfica, denominada XML Schema Semântico (XMLS+), para representar os esquemas das fontes locais e o esquema do mediador definidos em XML Schema [Fal01]. XMLS+ permite representar graficamente a estrutura dos tipos XML e incorpora mais semântica aos esquemas, uma vez que representa relacionamentos semânticos implícitos entre os tipos XML.

Propomos, ainda, um método, denominado XMLS+Matcher, para *matching* de esquemas XMLS+. No enfoque proposto, antes de iniciar o processo de *matching*, cada esquema deve ser mapeado em XMLS+. Cada passo do XMLS+Matcher é apoiado por um conjunto de heurísticas e regras de inferência, as quais auxiliam na identificação das correspondências entre os esquemas. As heurísticas utilizam as informações dos esquemas, tais como, nome e tipo dos elementos, e informações provenientes de consultas a dicionários semânticos para identificar possíveis correspondências. As regras de inferência utilizam as correspondências identificadas anteriormente para inferir novas correspondências, otimizando, assim, o processo de *matching*.

Uma outra contribuição deste trabalho é a definição formal dos vários tipos de Assertiva de Correspondência (AC) que são utilizados para especificar formalmente o relacionamento semântico entre elementos de esquemas XMLS+. O formalismo proposto permite especificar várias formas de correspondências, inclusive casos onde elementos semelhantes são representados de formas diferentes [VL99]. A vantagem desse formalismo é que permite especificar de forma precisa as correspondências do esquema do mediador com os esquemas locais, chamadas ACs do Mediador. Os sistemas de integração utilizam as ACs do Mediador para selecionar um conjunto de fontes locais que podem ser usadas para responder a uma consulta submetida ao sistema [AKMP01, BBM01, IFF+99].

Recentemente, vários trabalhos para *matching* de esquemas XML têm sido propostos [BBVC98, DRRS+02, MBR01, MCH02, RB01]. Os trabalhos em [BBVC98, MCH02] propõem o uso de modelos semânticos para representar esquemas XML que não usam a estrutura em árvore. [MCH02] propõe o uso de um modelo conceitual baseado no modelo orientado a objetos para representar DTDs. A desvantagem do uso destes modelos é que as correspondências geradas não

podem ser utilizadas diretamente na reformulação de consultas globais ou na manutenção de visões materializadas dos sistemas de integração. Outros trabalhos tratam a geração das correspondências entre esquemas representados em estrutura de árvore [DRRS+02, MBR01], entretanto estas abordagens identificam apenas a existência de uma correspondência, não sendo precisas o suficiente para informar a natureza da correspondência identificada, além de não permitirem especificar correspondências entre esquemas com estruturas diferentes. Neste trabalho, mostraremos como o uso de XMLS+ juntamente com as assertivas de correspondência resolvem essas limitações. Outros trabalhos relacionados serão comentados nos demais capítulos.

A Dissertação é dividida em sete Capítulos como se segue. No Capítulo 2, discutimos as principais questões envolvidas no problema da integração de informações em sistemas convencionais e em sistemas Web. No Capítulo 3, apresentamos o padrão XML. No Capítulo 4, discutimos a notação gráfica usada neste trabalho. No Capítulo 5, abordamos o método para *matching* de XMLS+ proposto. No Capítulo 6, apresentamos as heurísticas e as regras de inferência para suportar cada passo do XMLS+Matcher. E, no Capítulo 7, apresentamos nossas conclusões e direções para trabalhos futuros.

CAPÍTULO 2

INTEGRAÇÃO DE INFORMAÇÕES

Integrar informações distribuídas em múltiplas fontes de informação, autônomas e heterogêneas, tem surgido como um novo e estratégico requisito nas modernas empresas. Neste capítulo, discutimos a integração de informações. Na Seção 2.1, caracterizamos o problema da integração de informações. Na Seção 2.2, descrevemos as principais arquiteturas propostas na literatura para integrar informações. Na Seção 2.3, discutimos a integração de informações na Web. Na Seção 2.4, discutimos os sistemas de integração baseados em um esquema global. E, finalmente, na Seção 2.5, apresentamos o ambiente para geração e manutenção de mediadores do projeto ADaWeb.

2.1 Introdução

Várias organizações utilizam bancos de dados em diferentes plataformas, incluindo computadores de grande porte (*mainframes*), estações de trabalho (*workstations*) e servidores configurados para uma rede corporativa. Isto estimula a descentralização das informações dentro de uma organização e induz a uma proliferação de bancos de dados. Geralmente, estes bancos de dados obedecem a diferentes conjuntos de requisitos para modelar objetos idênticos ou similares. Isto possibilita o surgimento de informações relacionadas, bem como, a presença de informações redundantes nos diferentes bancos de dados.

Avanços nas áreas de banco de dados e computação distribuída, juntamente com a necessidade de prover acesso integrado às informações persistentes em diferentes plataformas, estimularam o surgimento de sistemas de integração de informações [EN00, OV99]. Neste contexto, podemos

identificar alguns cenários onde usuários requisitam as funcionalidades de um sistema de integração. Em um cenário comum, a necessidade da integração surge quando as informações dos consumidores de uma empresa estão descentralizadas em muitos bancos de dados e a empresa gostaria de conhecer o perfil dos seus consumidores, integrando todas as informações em uma única visão. Em um outro cenário, a necessidade surge quando se deseja disponibilizar informações integradas de múltiplos bancos de dados de diferentes organizações. Para ilustrarmos o segundo cenário, imaginemos um usuário que deseja comprar um imóvel em uma determinada região e deseja utilizar um sistema de integração de informações disponível na Web, que integra os bancos de dados sobre imóveis, escolas e crimes da sua cidade. Uma possível consulta desse usuário poderia ser:

> *"Encontre todas os imóveis com três quartos, dois banheiros, área interior de até 500 m^2, preço até R$ 70.000,00, em uma região que tenha escola a uma distância de até 200m e uma média de crimes que não exceda 15 incidentes por ano. Agrupe as respostas por região e em ordem decrescente de preço. Para cada imóvel, mostre também todas as escolas próximas".*

Existem diversas arquiteturas propostas para permitir o acesso integrado a informações armazenadas em diferentes bancos de dados. Na Seção 2.2, apresentamos as principais arquiteturas.

2.2 Arquiteturas para Integração de Informações

As arquiteturas para integração de informações diferem quanto ao grau de distribuição dos dados, heterogeneidade e autonomia:

- Em muitos sistemas, o dado está distribuído em diversos bancos de dados. Estes bancos de dados podem estar em um ou mais computadores e podem estar distribuídos geograficamente, porém interconectados através de uma rede.
- A heterogeneidade pode ocorrer em diferentes níveis dos bancos de dados. Por exemplo, diferentes bancos de dados podem utilizar diferentes linguagens para desenvolver aplicações, diferentes linguagens de consulta, diferentes modelos de dados, diferentes Sistemas Gerenciadores de Bancos de Dados (SGBDs), diferentes vocabulários, etc. No Capítulo 5, discutimos a heterogeneidade semântica [MRJ99, ST98]. Tal heterogeneidade ocorre quando

um conceito do mundo real é modelado de diferentes formas pelos projetistas dos bancos de dados, e envolve outras heterogeneidades, tais como de vocabulário, de modelo e estrutural.

- Autonomia refere-se à distribuição do controle, não dos dados, e indica o nível no qual os SGBDs individuais podem operar independentemente. A *autonomia de projeto* refere-se à liberdade que um SGBD tem para utilizar um modelo de dados mais adequado às suas necessidades. A *autonomia de comunicação* refere-se à liberdade que um SGBD tem de decidir quando e como responder às requisições de outros SGBDs. A *autonomia de execução* refere-se à liberdade que um SGBD tem de decidir a ordem de execução das operações locais e externas. A *autonomia de associação* refere-se à liberdade que um SGBD tem de decidir quando e como ele compartilha suas funcionalidades e seus recursos com outros SGBDs.

Nas Seções seguintes, apresentamos as principais arquiteturas propostas na literatura para solucionar o problema da integração de informações [ERS99, Wie92].

2.2.1 Esquema Global

A arquitetura do Esquema Global foi a primeira a prover o compartilhamento de dados entre sistemas de bancos de dados heterogêneos e distribuídos. Esta arquitetura é baseada na integração total dos múltiplos esquemas dos bancos de dados para oferecer uma única visão integrada destes bancos.

A Figura 2.1 apresenta esta arquitetura que consiste dos seguintes componentes: (i) bancos de dados locais (BD Li); (ii) esquemas dos bancos de dados locais; (iii) esquema global, resultante da integração de todos os esquemas locais; e (iv) usuários/aplicações, que consultam os bancos de dados locais através do esquema global.

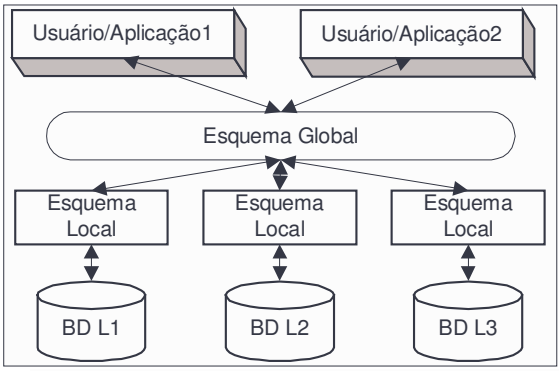


Figura 2.1: ***Esquema Global***

A principal vantagem desta arquitetura é que ela provê transparência no acesso a dados heterogêneos e distribuídos. Assim, múltiplos bancos de dados podem ser apresentados, logicamente, aos usuários como um grande banco de dados. Algumas das desvantagens desta arquitetura são: a complexidade na geração automática do esquema global, devido à dificuldade de

identificar as correspondências entre os esquemas locais; a visão única e extensa oferecida pelo esquema global, representando a integração unificada de todos os bancos de dados e restringindo os usuários a visualizarem apenas uma representação dos dados; e a dificuldade na manutenção do esquema global, visto que, para qualquer evolução dos esquemas locais, o esquema global tem que ser atualizado.

2.2.2 Sistema de Bancos de Dados Federados

Sistema de Bancos de Dados Federados é uma coleção de sistemas de bancos de dados cooperantes e autônomos que participam da federação para permitir o compartilhamento parcial e controlado de seus dados. A característica principal de uma federação é a cooperação entre sistemas independentes.

Esta arquitetura está apresenta na Figura 2.2 e consiste dos seguintes componentes: (i) bancos de dados locais (BD Li); (ii) esquemas dos bancos de dados locais; (iii) esquemas componentes, que correspondem à representação dos bancos de dados no modelo comum; (iv) esquemas de exportação, que definem a parte dos esquemas componentes acessível para a camada da federação; (v) esquema federado, que provê a visão integrada de todos os esquemas de exportação disponíveis; (vi) esquema externo, que representa um nível de abstração usado quando o esquema federado é muito complexo; e (vii) usuários/aplicações, que podem acessar os dados através do esquema federado ou do esquema externo.

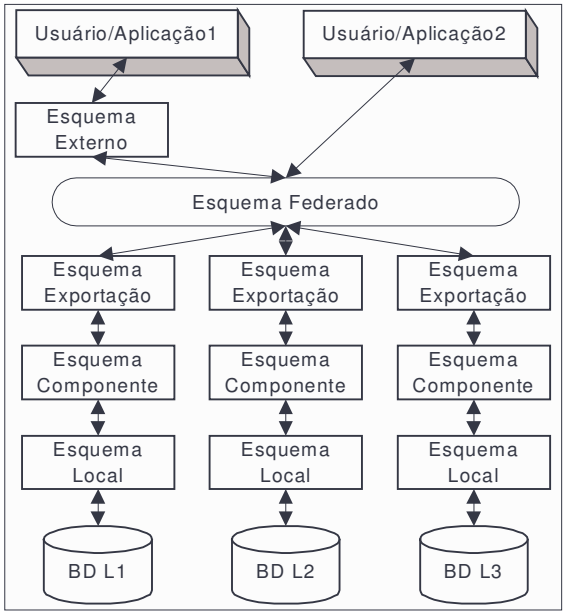


Figura 2.2: ***Bancos de Dados Federados***

A arquitetura do Sistema de Bancos de Dados Federados apresenta algumas vantagens similares àquelas apresentadas pela arquitetura do Esquema Global, tais como: a transparência de distribuição e heterogeneidade dos dados. Outra vantagem, é que o usuário pode definir um esquema de acordo com seus próprios requisitos (esquema externo), a partir do esquema federado.

Algumas desvantagens dessa arquitetura também estão presentes na arquitetura de Esquema Global, como por exemplo, a dificuldade para geração e manutenção do esquema federado.

2.2.3 Sistema de Bancos de Dados Múltiplos

Sistema de Bancos de Dados Múltiplos é uma coleção de múltiplos sistemas de bancos de dados locais. Nesta arquitetura não existe o conceito de esquema integrado que represente a unificação dos esquemas locais. Para permitir o acesso integrado aos bancos de dados é disponibilizada uma linguagem que oferece construtores para executar consultas envolvendo vários bancos de dados ao mesmo tempo.

A Figura 2.3 apresenta esta arquitetura que consiste dos seguintes componentes: (i) bancos de dados locais; (ii) esquemas locais; (iii) esquemas conceituais, representados em um mesmo modelo de dados, os quais definem a parte do esquema local acessível para o sistema de integração; (iv) esquema de dependências, que define as dependências e o conjunto de restrições de integridade globais envolvendo todos os bancos de dados; (v) esquema externo, que representa uma visão externa envolvendo dados de um ou mais esquemas conceituais; e (vi) usuários/aplicações, que consultam os dados distribuídos através do esquema externo.

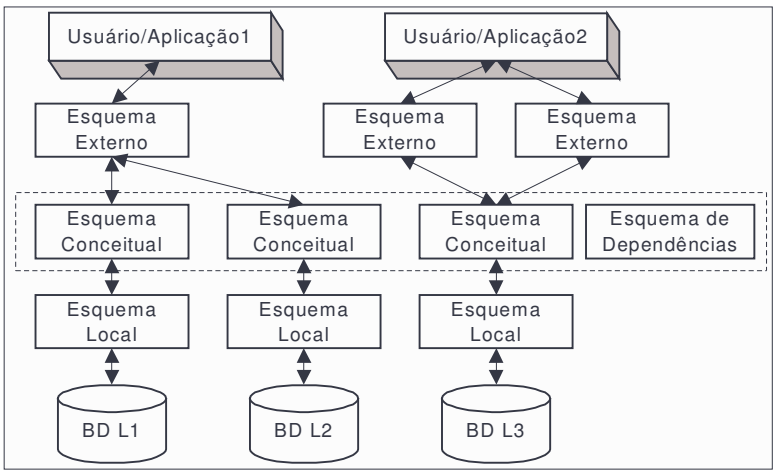


Figura 2.3: ***Sistemas de Bancos de Dados Múltiplos***

Uma vantagem apresentada por esta arquitetura é que ela permite que os bancos de dados participantes do sistema tenham autonomia total. Por outro lado, como não existe a noção de um esquema integrado, a desvantagem desta arquitetura é que, o usuário deve conhecer a localização exata dos bancos de dados, os conceitos definidos nos esquemas locais e o conteúdo de cada banco

para realizar uma consulta. Como conseqüência, o usuário é responsável por detectar e resolver os conflitos semânticos entre os esquemas conceituais.

2.2.4 Arquitetura de Mediadores

A arquitetura de mediadores provê acesso integrado e transparente a múltiplas fontes de informação através de mediadores. Mediadores são módulos de *software* que exploram o conhecimento representado em um conjunto ou subconjunto de dados para gerar informações para aplicações residentes em uma camada superior [Wie92].

Na arquitetura de mediadores, o acesso aos dados distribuídos em múltiplas fontes de informação é efetuado através de consultas que são submetidas ao sistema através do mediador. O mediador oferece uma visão integrada dessas fontes. Basicamente, existem dois enfoques para suportar visões integradas: virtual e materializado. No enfoque virtual, o sistema de integração utiliza um conjunto de correspondências entre o esquema da visão do mediador e os esquemas locais, para determinar como as consultas na visão do mediador serão respondidas a partir de consultas nas fontes locais. Os resultados dessas consultas são traduzidos, filtrados e integrados, e depois, o resultado final é retornado ao usuário. No enfoque materializado, as informações relevantes são previamente extraídas das fontes de informação, e, posteriormente, traduzidas, filtradas, integradas e armazenadas em um repositório (também chamado de *Data Warehouse*), de maneira que as consultas possam ser avaliadas diretamente neste repositório centralizado, sem a necessidade de acessar as fontes de informação locais. Um ponto crítico do enfoque materializado é manter as visões materializadas consistentes com relação às atualizações nas fontes (Manutenção da Visão Materializada). Os algoritmos de manutenção de visões materializadas [Vid02a, Vid02b] utilizam as correspondências entre o esquema do mediador e os esquemas locais para atualizar corretamente a visão, de forma a refletir as atualizações ocorridas nas fontes.

A arquitetura de mediadores para o enfoque virtual é apresentada na Figura 2.4 e consiste de quatro componentes distribuídos em três níveis:

1. *Nível dos dados*. Nível que representa as informações locais.
   - *Fontes de Informação* (FIs): podem ser autônomas e heterogêneas, porque, na maioria das vezes, elas não foram projetadas com a intenção de compartilhamento. As FIs podem ser

bancos de dados, depósitos de objetos, bibliotecas digitais, páginas HTML e XML, entre outros tipos de fonte.

- *Tradutores*: convertem os dados das FIs para um modelo comum e convertem consultas de aplicações em consultas específicas da fonte correspondente.

2. *Nível da integração*. Nível que representa a integração das FIs que fazem parte do sistema de integração de informações.

- *Mediadores*: "interfaces" através das quais os usuários consultam e atualizam múltiplas fontes de informação.

3. *Nível dos usuários*. Nível que representa o acesso às FIs.

- *Usuário/Aplicação*: podem acessar as informações das fontes heterogêneas de maneira transparente e uniforme através de mediadores e tradutores.

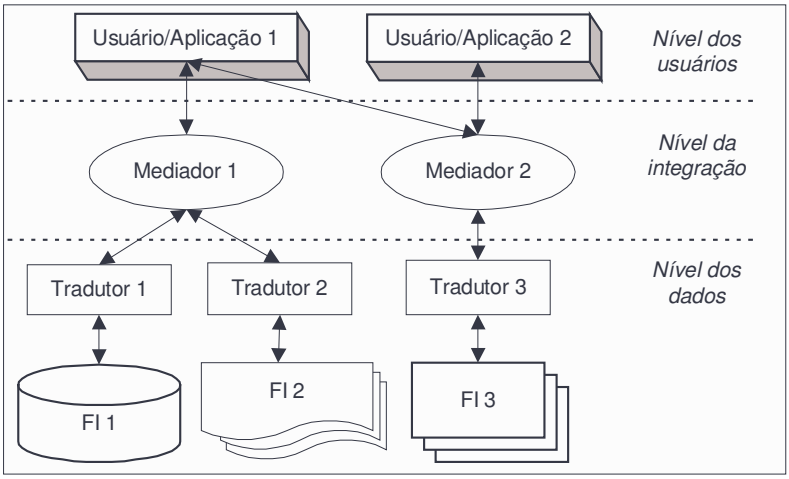


Figura 2.4: ***Arquitetura de Mediadores - Enfoque Virtual***

2.3 Integração de Informações na Web

A última década presenciou o crescimento de um importante veículo de disseminação de informações, a *World Wide Web* (Web). A Web provê uma padronização simples e universal para publicação e troca de informações. Devido a estas características, a Web vem crescendo em número de usuários (individuais, empresas, governos) e aplicações (serviços de busca, comércio eletrônico, etc.), possibilitando que cada vez mais informações sejam disponibilizadas neste ambiente. Atualmente, um dos grandes desafios dos pesquisadores em bancos de dados e Web é prover acesso integrado e transparente a essas informações [BGL+99, CA99, Cat99, DRRS+02, FLM98, GSN99, KMAA+01, MCH02, MK99].

Muitos dos problemas encontrados na construção dos sistemas de integração de informações na Web são similares aos problemas encontrados em sistemas de integração de bancos de dados convencionais. Porém, para integrar dados na Web, alguns problemas adicionais precisam ser solucionados:

- o número de fontes de informação pode ser muito grande, o que dificulta a construção de uma visão integrada destas fontes;
- as fontes de informação são muito dinâmicas, assim a adição ou remoção destas fontes deve ser feita de maneira a minimizar o impacto na visão integrada;
- a maioria dos dados disponíveis na Web é considerado dado semi-estruturado. Este tipo de dado, apesar de ter uma estrutura, esta pode ser desconhecida, irregular, ou mesmo estar embutida no dado [Abi97, Bun97]. A Tabela 2.1 apresenta, de forma resumida, uma comparação entre um banco de dados convencional e uma fonte de informação semi-estruturada.

Bancos de Dados Convencionais	***Fontes de Informação Semi-estruturadas***
Esquema predefinido	Nem sempre há um esquema predefinido
Estrutura regular	Estrutura irregular
Estrutura independente dos dados	Estrutura embutida no dado
Estrutura reduzida	Estrutura extensa
Estrutura fracamente evolutiva	Estrutura fortemente evolutiva

Tabela 2.1: ***Comparação entre Bancos de Dados Convencionais e Fontes de Informação Semi-estruturadas***

Os dados semi-estruturados vêm surgindo como um importante tópico de pesquisa por várias razões [Bun97]: (i) existem fontes de informação semi-estruturadas, as quais poderiam ser tratadas como bancos de dados. Para isso, técnicas, ferramentas e metodologias disponíveis para a manipulação de bancos de dados convencionais devem ser estendidas para tratar estas fontes; (ii) é importante ter um formato extremamente flexível para troca de informações entre fontes heterogêneas, no qual todos os modelos de dados pudessem ser convertidos facilmente; e (iii) mesmo tratando com dados estruturados, pode ser interessante visualizar esses dados como sendo semi-estruturados. Por exemplo, consultar as fontes de informação sem necessariamente conhecer o esquema associado a elas.

A integração de informações na Web requer a definição de um modelo de dados comum, flexível o bastante para capturar as diferentes representações dos dados nas diversas fontes heterogêneas. Vários trabalhos estão propondo modelos baseados em grafos ou árvores [Abi97,

BPMM00, CDSS98, CGMH94]. As principais razões para a popularidade deste tipo de modelo são a sua simplicidade e a facilidade de representar qualquer outro modelo em um modelo baseado em grafo ou árvore.

Um dos primeiros modelos de dados baseado em grafos utilizado para representar dados semi-estruturados foi o OEM (*Object Exchange Model*) [CGMH94]. O OEM foi projetado para ser usado no TSIMMIS (*The Stanford IBM Manager of Multiple Information Sources*). Um outro modelo de dados é XML (*eXtensible Markup Language*) [BPMM00], o qual é o novo padrão proposto pelo W3C (*World Wide Web Consortium*) [W3C] para representar informações na Web. Devido à flexibilidade da XML para representar tanto dados estruturados como semi-estruturados, e a facilidade de converter qualquer dado em XML, existe um grande interesse em utilizar este padrão como modelo de dados comum para troca e integração de informações [CCS00, GMW99]. No Capítulo 3, apresentamos uma visão mais aprofundada do XML.

A arquitetura que vem sendo bastante adotada nos sistemas de integração de informações baseados em XML é a arquitetura de mediadores (*XML-Based Mediators*) [ABS00, BGL+99, GSN99, OV99, VBO01, VLS01]. Estes sistemas são baseados em uma arquitetura de três níveis de esquema mostrada na Figura 1.1 do Capítulo 1. O mediador suporta uma visão integrada XML (virtual ou materializada) e as fontes locais exportam visões XML.

2.4 Sistemas de Integração de Informações baseados em um Esquema Global

Os sistemas de integração de informações que abordamos neste trabalho são caracterizados por uma arquitetura baseada em um esquema global (ou esquema do mediador) e em um conjunto de fontes locais. Desta forma, objetivam combinar informações armazenadas em diferentes fontes e disponibilizar ao usuário uma visão unificada destas informações. Esta visão é representada pelo esquema do mediador e pode ser consultada pelo usuário.

Para construir um sistema de integração, deve ser especificado um esquema do mediador e fornecidas as descrições das fontes locais. Cada descrição de uma fonte local contém o esquema da fonte, que descreve o conteúdo da mesma, e as correspondências entre esse e o esquema do mediador. Estas correspondências determinam como as consultas submetidas ao esquema do mediador serão respondidas a partir de consultas nas fontes locais.

Um serviço básico fornecido pelos sistemas de integração é responder às consultas submetidas ao esquema do mediador [Len02]. O processamento de consulta nos sistemas de integração requer um passo para reformulação de consulta. A consulta submetida ao esquema do mediador tem que ser reformulada em um conjunto de consultas a serem submetidas às fontes locais. Para reformular a consulta, o sistema de integração utiliza as correspondências entre os esquemas das fontes e o esquema do mediador fornecidas previamente.

Um exemplo de sistema de integração baseado no esquema do mediador é o Tukwila [IFF+99]. A arquitetura deste sistema é apresentada na Figura 2.5. Neste sistema, o componente *Reformulator* é responsável por reformular uma consulta do usuário em consultas a serem submetidas às fontes locais. Para isso, o *Reformulator* utiliza as correspondências entre o esquema do mediador e as fontes locais (*source mappings*), armazenadas no *Catalog*. O componente *Optimizer* transforma a consulta recebida pelo *Reformulator* em um plano de execução de consulta a ser executada pelo *Execution Engine*.

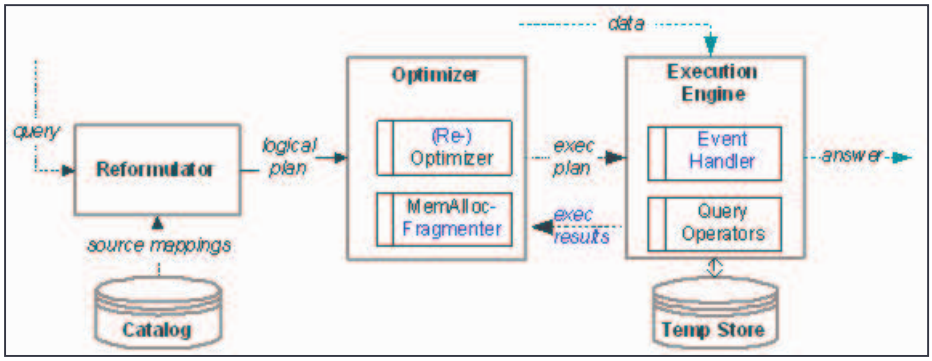


Figura 2.5: ***Arquitetura do Sistema Tukwila*** *[IFF+99]*

Para que os sistemas de integração, como o Tukwila, possam responder às consultas de forma atualizada, os mesmos devem manter os catálogos, como o *Catalog*, sempre consistentes com relação às informações armazenadas nas fontes locais. Objetivando contribuir com esta linha de pesquisa, foi proposto no projeto ADaWeb [ADaWeb01], um ambiente para geração e manutenção de mediadores, o qual contém uma ferramenta que identifica e mantém atualizadas as correspondências entre o esquema do mediador e os esquemas das fontes locais. A vantagem deste ambiente é que o mesmo pode ser utilizado por qualquer sistema de integração baseado em um esquema do mediador. A arquitetura deste ambiente é descrita na Seção 2.5.

2.5 Ambiente para Geração e Manutenção de Mediadores

Um dos objetivos do projeto ADaWeb é auxiliar o projetista nas tarefas de geração e manutenção de mediadores. Para isso, é proposto neste projeto um Ambiente para Geração e Manutenção de Mediadores. Gerar e manter mediadores significa gerar e manter catálogos que contém as descrições das fontes locais, o esquema do mediador e as correspondências entre os esquemas das fontes e o esquema do mediador. Gerar e manter catálogos são tarefas complexas. Algumas das razões para esta complexidade são a identificação das correspondências entre o esquema do mediador e os esquemas locais e a manutenção das mesmas em decorrência da evolução dos esquemas.

Para auxiliar na geração dos mediadores, o ambiente proposto identifica, previamente, as correspondências entre os esquemas locais e disponibilizando os dados necessários para guiar o projetista durante o processo de geração dos mediadores. Para auxiliar na manutenção, o ambiente informa sempre que uma fonte de informação for inserida ou excluída do sistema de integração ou tiver o seu esquema evoluído, e realiza as atualizações necessárias para manter os mediadores consistentes com as fontes locais. Os mediadores gerados pelo ambiente são baseados em XML. A arquitetura deste ambiente está apresentada na Figura 2.6 e seus componentes são listados a seguir.

- **Fontes de Informação**. As Fontes de Informação podem ser heterogêneas, com modelos de dados e linguagens de consulta próprios, e autônomas. Além disso, elas podem ser dinâmicas, visto que, podem suportar aplicações locais e atualizar suas informações e seus esquemas independentemente, sem se preocupar em como estas atualizações afetarão o sistema de integração de informações que as mesmas participam.

- **Ferramenta para Mapear Esquemas Locais em Esquemas XML Schemas Semânticos**. Mapeia os esquemas locais definidos no modelo de dados próprio da fonte em esquemas definidos em XML Schema Semântico, representação comum deste ambiente. O processo de geração de esquemas XML Schemas Semânticos é descrito no Capítulo 4.

- **Ferramenta para *Matching* de Esquemas XML Schemas Semânticos**. Identifica correspondências entre esquemas XML Schemas Semânticos. Desta forma, esta ferramenta pode ser utilizada tanto para identificar as correspondências entre o esquema do mediador e os

esquemas locais como para identificar correspondências entre os esquemas locais.

A principal contribuição desta dissertação é a definição do método para identificação de correspondências entre esquemas XML Schemas Semânticos, utilizado pela ferramenta. Este método é apresentado nos Capítulos 5 e 6. Vale ressaltar que o mesmo pode ser utilizado por qualquer ambiente para integração de informações.

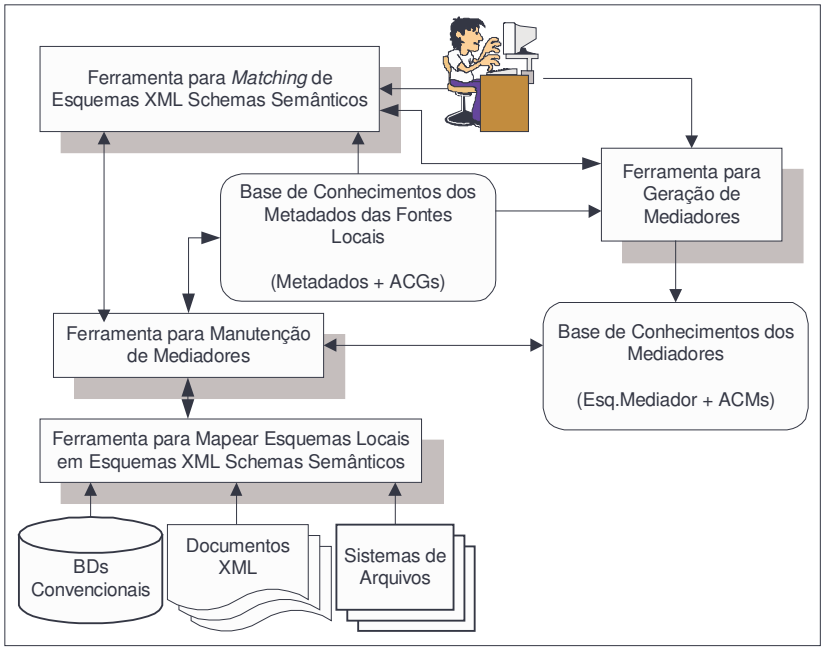


Figura 2.6: ***Arquitetura do Ambiente de Geração e Manutenção de Mediadores***

- **Ferramenta para Geração de Mediadores**. Gera os mediadores de acordo com a visão que os usuários desejam obter das informações armazenadas nas fontes locais.

- **Base de Conhecimentos dos Metadados das Fontes Locais**. Armazena as descrições das fontes de informação que participam do sistema de integração. Esta descrição inclui o esquema da fonte de informação (metadados) e as correspondências entre os esquemas locais, chamadas de Assertivas de Correspondência Globais (ACGs).

- **Base de Conhecimentos dos Mediadores**. Armazena o esquema da visão do mediador e as correspondências entre este esquema e os esquemas locais, chamadas de Assertivas de Correspondência do Mediador (ACMs).

- **Ferramenta para Manutenção de Mediadores**. Mantém as bases de conhecimentos e os mediadores atualizados em decorrência de atualizações no sistema de integração.

É importante notar que a base de conhecimentos dos metadados das fontes locais e a base de conhecimentos do mediador correspondem ao *Catalog* do sistema Tukwila.

CAPÍTULO 3

XML: MODELO DE DADOS, LINGUAGEM DE CONSULTA E LINGUAGENS DE ESQUEMA

XML está sendo usado como padrão para troca de informações na Web, principalmente devido à sua flexibilidade de representar tanto dados estruturados como semi-estruturados. Neste Capítulo, apresentamos XML[1]: na Seção 3.1, discutimos as motivações ao uso do XML, incluindo uma visão geral e sua sintaxe básica; na Seção 3.2, apresentamos a XPath, uma linguagem para manipulação de documentos XML; e na Seção 3.3, descrevemos a DTD e a XML Schema, as duas principais linguagens para definição de esquemas XML.

3.1 XML

Atualmente, devido à sua simplicidade, HTML (*HyperText Markup Language*) é o formato mais utilizado para publicação de dados na Web. Porém, HTML possui algumas limitações, tais como:

- O conjunto de marcas fornecido pela HTML não é extensível, ou seja, não existe suporte para criação de novas marcas;
- As marcas da HTML se restringem à apresentação do documento. Desta forma, um documento HTML corresponde a uma visão particular e pré-definida do conteúdo disponibilizado pelo documento;

[1] Não é intenção deste Capítulo conter uma total descrição do XML.

- HTML não provê suporte para aplicações que precisam validar os dados que estão sendo importados;
- HTML associa conteúdo do documento à sua apresentação. Desta forma, pouca semântica pode ser extraída de um documento HTML.

Para suprir essas limitações, o W3C (*World Wide Web Consortium*) [W3C], órgão independente que regulamenta padrões e métodos adotados pela Internet, desenvolveu uma linguagem de marcação extensível, chamada XML (*eXtensible Markup Language*) [BPMM00]. XML, assim como HTML, tem origem na SGML (*Standard Generalized Markup Language*), que é um padrão internacional para definição de formatos de representação de texto em meio eletrônico. XML oferece:

- Independência de conteúdo e apresentação. XML aborda apenas o conteúdo de um documento. A apresentação pode ser tratada por linguagens específicas para apresentação, tais como: *Cascading Style Sheets* (CSS) e *eXtensible Stylesheet Language* (XSL);
- Linguagem Extensível. XML permite que os usuários definam novas marcas de acordo com o domínio que está sendo modelado. Estas marcas servem para descrever o conteúdo de um documento;
- Validação. Um documento XML pode ser associado a uma definição de esquema XML, a qual define a estrutura do documento. Assim, aplicações podem validar seus dados de acordo com um esquema.

Os benefícios da XML incluem:

- Suporte para múltiplas visões de um mesmo conteúdo para diferentes usuários e tecnologias;
- Semântica agregada aos documentos através das marcas;
- Consultas baseadas na semântica fornecida pelos documentos;
- Uma plataforma para compartilhamento de informações através de uma sintaxe comum.

A seguir, descrevemos e exemplificamos os componentes básicos de um documento XML:

- ***Elemento***

O elemento é descrito por uma marca inicial (i.é, <nome_do_elemento>) e uma marca final (i.é, </nome_do_elemento>). As marcas inicial e final são também chamadas de marcação. O conteúdo de um elemento é delimitado pela marcação. Cada elemento pode conter texto, conter outros elementos aninhados (subelementos), ter conteúdo misto ou ser vazio.

Exemplo 3.1: Considere o documento XML abaixo:

```
<pessoa>
   <nome> Renata </nome>
   <e-mail> renata@lia.ufc.br </e-mail>
</pessoa>
```

As marcas <pessoa> e </pessoa> descrevem a estrutura do elemento pessoa. O elemento pessoa possui subelementos, como: nome e e-mail, todos contendo suas marcações. Como podemos observar, as marcas de um documento são balanceadas, ou seja, são fechadas na ordem inversa da que foram abertas.

- ***Atributo***

XML permite associar atributos aos elementos. XML usa o termo "atributo" para especificar propriedades dos elementos [ABS00]. O valor de um atributo é sempre um texto e deve aparecer entre aspas.

Exemplo 3.2: Considere o documento XML apresentado no Exemplo 3.1 acrescido do CPF como atributo do elemento pessoa. O novo documento é apresentado a seguir:

```
<pessoa cpf="58726596802">
   <nome> Renata </nome>
   <e-mail> renata@lia.ufc.br </e-mail>
</pessoa>
```

Um documento XML pode ser considerado bem-formado e válido. Para ser bem-formado, o documento deve obedecer algumas regras, tais como: (1) conter pelo menos um elemento; (2) conter uma única marca inicial e marca final descrevendo um elemento que contenha todo o documento (chamado elemento raiz); e (3) todas as outras marcas devem estar aninhadas apropriadamente. Para ser válido, além do documento ser bem-formado, este deve obedecer a uma

estrutura especificada por um esquema XML. A Seção 3.3 mostra como definir esquemas para documentos XML.

A Figura 3.1 apresenta um exemplo de documento XML que publica os artigos, os livros, seus respectivos autores e instituições que estão disponíveis na fonte de informação Bib_1. Este documento XML será utilizado ao longo deste Capítulo para exemplificar a linguagem de consulta e as linguagens de esquema. Além disso, o mesmo será enriquecido sempre que necessário, de modo a facilitar a compreensão dessas linguagens.

```
<bib1>
 <livro1 autoresref1="A1">
    <ano1> 1999 </ano1>
    <isbn1> 3826561422 </isbn1>
    <título1> Transaction Management in Multidatabase Systems </título1>
    <editora1> Shaker-Verlag </editora1>
 </livro1>
 <artigo1 autoresref1="A1 A2">
    <ano1> 2000 </ano1>
    <cdu1> 681.31:061.68 </cdu1>
    <título1> Temporal Serialization Graph Testing </título1>
    <local_ publicação1> XV SBBD </local_ publicação1>
 </artigo1>
 <autor1 id_autor1="A1" instituiçãoref1="I1">
    <nome1> Ângelo Brayner </nome1>
    <e-mail1> brayner@unifor.br </e-mail1>
 </autor1>
 <autor1 id_autor 1="A2" instituiçãoref1="I1 I2">
    <nome1> José Maria Monteiro </nome1>
    <e-mail1> zemaria@lia.ufc.br </e-mail1>
 </autor1>
```

```
 <instituição1 id_instituição1="I1">
    <nome1> Unifor </nome1>
    <endereço1>
       <cidade1> Fortaleza </cidade1>
       <estado1> Ceará </estado1>
       <país1> Brasil </país1>
    </endereço1>
 </instituição1>
 <instituição1 id_instituição1="I2">
    <nome1> UFC </nome1>
    <telefone1> 288-9845  </telefone1>
    <endereço1>
       <cidade1> Fortaleza </cidade1>
       <estado1> Ceará </estado1>
       <país1> Brasil </país1>
    </endereço1>
 </instituição1>
</bib1>
```

Figura 3.1: ***Documento XML da Fonte Bib_1***

3.2 Navegando em Documentos XML

Muitas linguagens de consulta XML estão sendo propostas [CS00, DFF+99]. Uma análise comparativa das linguagens de consulta é apresentada nos trabalhos [BL01, FSW00]. Nesta Seção, apresentamos a linguagem XPath [BBC+01] utilizada para navegar através de "caminhos" em um documento XML. A XPath está sendo desenvolvida pelos grupos *W3C XML Query Working Group* e *XSL Working Group* e projetada para ser inserida em outras linguagens (chamadas de *host language*), como por exemplo, XQuery [BCFF+02].

XPath utiliza o modelo de dados de consulta XML do W3C [FR01], o qual representa um documento XML como uma árvore rotulada nos nós. A Figura 3.2 apresenta a árvore de dados do

documento Bib_1. As folhas da árvore contêm valores atômicos (e.g. *String*), enquanto que os nós internos (e.g. $nome_1$) correspondem a elementos ou atributos.

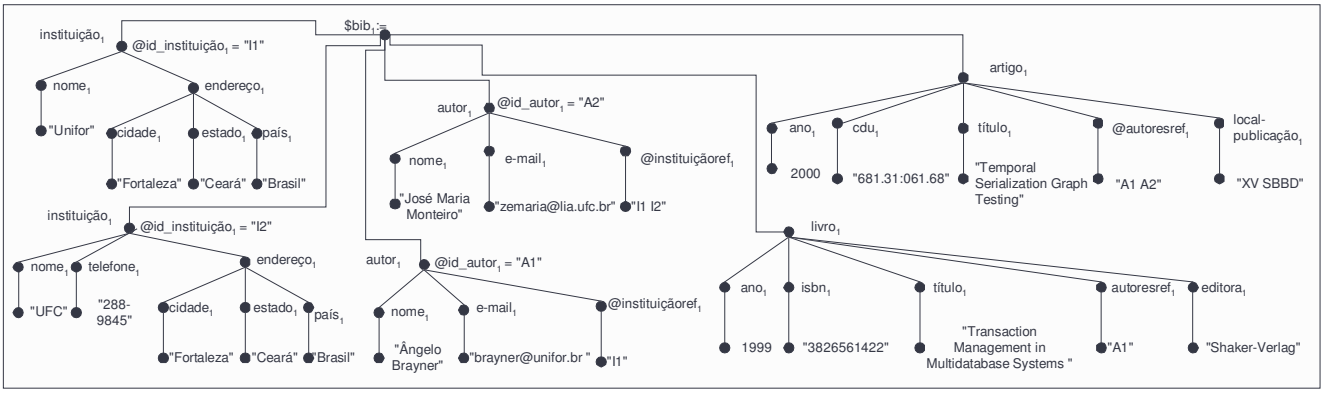


Figura 3.2: ***Árvore de Dados do Documento Bib1***

O bloco de construção básico da XPath é a expressão. A linguagem provê diversos tipos de expressão que podem ser construídos a partir de palavras-chave, símbolos e operandos. Em geral, os operandos de uma expressão são outras expressões. XPath é uma linguagem funcional que permite que vários tipos de expressão sejam aninhados.

A expressão que localiza um nó em uma árvore e retorna uma seqüência de nós distintos na ordem do documento é chamada de expressão de caminho. Tal expressão consiste de um ou mais passos. Cada passo representa um movimento ao longo do documento em uma determinada direção e retorna uma lista de nós que servem como um ponto de partida para o próximo passo.

A seguir, exemplificamos algumas expressões de caminho (o valor da variável bib_1 corresponde ao documento Bib_1):

Exemplo 3.3: A expressão de caminho **bib_1/artigo$_1$/título$_1$** consiste de três passos. O primeiro passo seleciona o nó raiz do documento Bib_1. O segundo passo seleciona os elementos $artigo_1$, filhos do elemento raiz. O terceiro passo seleciona os elementos $título_1$ dos elementos $artigo_1$ recuperados no passo anterior. Assim, essa expressão retorna todos os elementos $título_1$ contidos nos elementos $artigo_1$ contidos em bib_1.

Exemplo 3.4: A expressão de caminho **bib_1/autor$_1$/nome$_1$/text()** retorna o $nome_1$ de todos os elementos $autor_1$ contidos em bib_1.

Exemplo 3.5: A expressão de caminho **bib_1$/livro$_1$ [ano$_1$ = "1999"]** retorna todos os elementos livro$_1$ em bib_1$ que possuem o elemento ano$_1$ igual a "1999".

3.3 Definindo Esquemas para Documentos XML

Um esquema descreve a estrutura lógica de uma fonte de informação, incluindo os tipos dos dados, os relacionamentos entre os dados e as restrições envolvendo os dados. Para XML, um esquema é usado para descrever os elementos, o conteúdo dos elementos, a lista de atributos de um determinado elemento e as restrições sobre os elementos e os atributos.

A primeira linguagem proposta para definição de esquema XML foi a DTD (*Document Type Definition*) [BPMM00]. Entretanto, a DTD possui algumas limitações como pode ser observado na Seção 3.3.1. Para sobrepor as limitações da DTD, outras linguagens estão sendo propostas. Entre elas, destacamos: XML Schema [BM01, Fal01, TBMM01] e RDF [BG00]. Estas linguagens são mais ricas em semântica e oferecem recursos adicionais para definição de esquemas. Uma análise comparativa dessas e de outras linguagens de esquema XML é apresentada nos trabalhos [BL01, LC00].

Nas Seções seguintes, apresentamos as linguagens DTD e XML Schema.

3.3.1 DTD

O principal objetivo de uma DTD é especificar quais elementos são permitidos e em que ordem estes devem aparecer no documento.

A seguir, descrevemos os principais componentes de uma DTD. Para exemplificar o uso destes componentes, utilizamos a DTD que representa a estrutura do documento Bib$_1$, apresentada na Figura 3.3.

- ***Declaração do Tipo do Documento***

A declaração do tipo do documento permite definir restrições na estrutura lógica do documento. Esta declaração deve ser especificada como se segue:
<DOCTYPE nome_elemento_raiz [declarações de elementos e/ou declarações de atributos]>

```
1.1 <!DOCTYPE bib1 [
1.2 <!ELEMENT bib1 (artigo1*, livro1*, autor1*, instituição1*)>
1.3 <!ELEMENT artigo1 (ano1, (isbn1 | cdu1), título1, local_publicação1)>
1.4 <!ELEMENT livro1 (ano1, isbn1, título1, editora1)>
1.5 <!ATTLIST artigo1 autoresref1 IDREFS #required>
1.6 <!ATTLIST livro1 autoresref1 IDREFS #required>
1.7 <!ELEMENT autor1 (nome1, e-mail1)>
1.8 <!ATTLIST autor1 id_autor1 ID #required>
1.9 <!ATTLIST autor1 instituiçãoref1 IDREF #implied>
1.10 <!ELEMENT instituição1 (nome1, telefone1?,endereço1)>
1.11 <!ATTLIST instituição1 id_instituição1 ID #required>
1.12 <!ELEMENT endereço1 (cidade1, estado1, país1)>
1.13 <!ELEMENT ano1 (#PCDATA)>
1.14 <!ELEMENT isbn1 (#PCDATA)>
1.15 <!ELEMENT cdu1 (#PCDATA)>
1.16 <!ELEMENT título1 (#PCDATA)>
1.17 <!ELEMENT editora1 (#PCDATA)>
1.18 <!ELEMENT local_publicação1 (#PCDATA)>
1.19 <!ELEMENT nome1 (#PCDATA)>
1.20 <!ELEMENT e-mail1 (#PCDATA)>
1.21 <!ELEMENT telefone1 (#PCDATA)>
1.22 <!ELEMENT cidade1 (#PCDATA)>
1.23 <!ELEMENT estado1 (#PCDATA)>
1.24 <!ELEMENT país1 (#PCDATA)>
1.25 ]>
```

Figura 3.3: ***DTD de Bib1***

Considere, por exemplo, a DTD de Bib_1. A declaração <DOCTYPE bib_1 [<!ELEMENT bib_1...>...]> (linhas 1.1 a 1.25) especifica o tipo do documento Bib_1.

No nosso exemplo, a DTD está inserida no documento (DTD interna). Entretanto, uma DTD também pode estar em um arquivo separado ou residir em uma URL diferente da URL do documento correspondente (DTD externa). Declarar uma DTD externa permite que diversos documentos na Web compartilhem uma DTD comum, facilitando a troca de informações.

- ***Declaração de Elemento***

Um elemento é declarado na DTD como se segue:
<!ELEMENT nome_elemento (conteúdo)>

A DTD permite que usuários especifiquem as restrições de ocorrência dos elementos, através das expressões regulares. A Tabela 3.1 apresenta as expressões regulares disponibilizadas pela DTD.

Símbolo	*Significado*
, (vírgula)	Significa *e* em ordem especificada
|	Significa *ou*
?	Significa opcional, porém, não é permitido mais de um
*	Significa zero ou mais
+	Significa um ou mais

Tabela 3.1: ***Expressões Regulares***

Considerando o elemento $instituição_1$ de Bib_1, a declaração <!ELEMENT $instituição_1$ ($nome_1$, $telefone_1$?, $endereço_1$)> (linha 1.10) especifica que este elemento contém uma seqüência de um elemento $nome_1$, zero ou um elemento $telefone_1$ e um elemento $endereço_1$.

- ***Declaração de Atributo***

Um atributo é declarado na DTD como se segue:
<!ATTLIST nome_elemento nome_atributo tipo_atributo valor_atributo>

O **nome_elemento** indica o nome do elemento para o qual o atributo está sendo declarado. O **tipo_atributo** indica a restrição aplicada aos valores deste atributo. Por exemplo: o tipo CDATA indica que o valor do atributo é texto; o tipo ID permite associar identificadores únicos aos elementos; o tipo IDREF indica que o valor do atributo é o identificador de um outro elemento. O tipo IDREFS indica que o valor do atributo é uma lista de identificadores separados por espaços em branco. O **valor_atributo** indica que o valor do atributo é obrigatório (*#Required*), opcional (*#Implied*), ou fixado em um determinado valor (*#Fixed*).

Considere o atributo instituiçãoref$_1$ do elemento autor$_1$ de Bib$_1$. A declaração <!ATTLIST autor$_1$ instituiçãoref$_1$ IDREF #implied> (linha 1.9) especifica que o valor deste atributo é uma referência a um identificador e seu valor é opcional.

Como podemos verificar, a DTD provê uma forma, relativamente fácil, de descrever a sintaxe e a semântica de documentos XML. Entretanto, a DTD possui várias limitações, como por exemplo:

- As restrições de integridade definidas pelos atributos dos tipos ID, IDREF e IDREFS. Os valores de todos os atributos ID têm que ser distintos em todo o documento, e todos IDREF/IDREFS devem conter valores de ID existentes, porém, nenhuma restrição ao tipo do elemento referenciado pelo IDREF e IDREF é imposta;
- Imposição de ordem para apresentar os dados do documento;
- Não há a noção de tipos básicos. O único tipo básico para um elemento é #PCDATA e para um atributo é CDATA;
- Apresenta uma sintaxe diferente da sintaxe proposta pela linguagem XML.

Para sobrepor as limitações da DTD, o *W3C XML Schema Working Group* está propondo a XML Schema, apresentada a seguir.

3.3.2 XML Schema

A linguagem XML Schema introduz novos construtores que fazem com que esta linguagem seja mais expressiva do que a DTD. Com isso, a XML Schema pode ser utilizada em uma maior variedade de aplicações. Algumas das principais características da XML Schema são:

- Um esquema XML Schema é escrito em XML. Desta forma, o usuário não precisa aprender uma nova sintaxe proprietária. Além disso, o esquema pode ser armazenado em um sistema de armazenamento XML, junto com os documentos XML;
- Permite que o usuário defina novos tipos de dados;
- Permite que o usuário defina o relacionamento de tipo/subtipo entre tipos.

A seguir, descrevemos os principais componentes da XML Schema. Para exemplificar o uso destes componentes, utilizamos o esquema XML Schema que representa a estrutura do documento Bib_1, apresentado na Figura 3.4.

- ***Declaração de Elemento***

Elementos são declarados utilizando o construtor element. Os principais atributos deste construtor são: name, que associa o elemento declarado a um nome; type, que atribui um tipo ao elemento; e minOccurs e maxOccurs, que especificam a restrição mínima e máxima de ocorrência do elemento, respectivamente: se a restrição mínima for igual a zero, o elemento é opcional, caso contrário é obrigatório; e se a restrição máxima for igual a um, o elemento é mono-ocorrência, caso contrário é multi-ocorrência.

Considere, por exemplo, o elemento $autor_1$. A declaração <element name="$autor_1$" type="$Tautor_1$" minOccurs="0" maxOccurs="unbounded"/> (linha 2.7) especifica que este elemento é do tipo $Tautor_1$ e a restrição de ocorrência é opcional e multi-ocorrência.

- ***Declaração de Atributo***

Atributos são declarados utilizando o construtor attribute. Os principais atributos deste construtor são: name, que associa o atributo declarado a um nome; type, que atribui um tipo ao atributo. O tipo do atributo indica a sua cardinalidade: se o tipo do atributo for IDREFS, este é multivalorado, caso contrário é monovalorado; e use, que especifica a restrição de ocorrência do atributo: se o use não

```
2.1 <schema>
2.2  <element name="bib1">
2.3   <complexType>
2.4    <sequence>
2.5     <element name="artigo1" type="Tartigo1" minOccurs="0"
          maxOccurs="unbounded"/>
2.6     <element name="livro1" type="Tlivro1" minOccurs="0"
          maxOccurs="unbounded"/>
2.7     <element name="autor1" type="Tautor1" minOccurs="0"
          maxOccurs="unbounded"/>
2.8     <element name="instituição1" type="Tinstituição1"
          minOccurs="0" maxOccurs="unbounded"/>
2.9    </sequence>
2.10   </complexType>
2.11 </element>

2.12  <complexType name=" Tpublicação1">
2.13  <sequence>
2.14   <element name="ano1" type="String"/>
2.15   <choice>
2.16    <element name="isbn1" type="Integer"/>
2.17    <element name="cdu1" type="Integer"/>
2.18   </choice>
2.19   <element name="título1" type="String"/>
2.20  </sequence>
2.21  <attribute name="autoresref1" type="IDREFS" use="required"/>
2.22 </complexType>

2.23 <complexType name="Tartigo1">
2.24  <complexContent>
2.25   <extension base="Tpublicação1" >
2.26    <sequence>
2.27     <element name="local-publicação1" type="String"/>
2.28    </sequence>
2.29   </extension>
2.30  </complexContent>
2.31 </complexType>

2.32 <complexType name="Tlivro1">
2.33  <complexContent>
2.34   <extension base="Tpublicação1">
2.35    <sequence>
2.36     <element name="editora1" type="String"/>
2.37    </sequence>
2.38   </extension>
2.39  </complexContent>
2.40 </complexType>

2.41 <complexType name=" Tautor1">
2.42  <sequence>
2.43   <element name="nome1" type="String"/>
2.44   <element name="e-mail1" type="String"/>
2.45  </sequence>
2.46  <attribute name="id_autor1" type="ID" use="required"/>
2.47  <attribute name="instituiçãoref1" type="IDREF" use="optional"/>
2.48 </complexType>

2.49 <complexType name=" Tinstituição1">
2.50  <sequence>
2.51   <element name="nome1" type="String"/>
2.52   <element name="telefone1" minOccurs="0" maxOccurs="1"/>
2.53   <element name="endereço1" type="Tendereço1"/>
2.54  </sequence>
2.55  <attribute name="id_instituição1"  type="ID"  use="required"/>
2.56 </complexType>

2.57 <complexType name="Tendereço1">
2.58  <sequence>
2.59   <element name="cidade1" type="String"/>
2.60   <element name="estado1" type="String"/>
2.61   <element name="país1" type="String"/>
2.62  </sequence>
2.63 </complextype>

2.64 </schema>
```

Figura 3.4: ***XML Schema de Bib1***

estiver explícito na declaração do atributo ou se seu valor for *optional*, o valor do atributo é opcional, caso contrário é obrigatório.

Considere o atributo id_instituição1. A declaração <attribute name="id_instituição1" type="ID" use="required"/> (linha 2.55) especifica que este atributo é do tipo ID é monovalorado e obrigatório.

- ***Definição de Tipo Complexo***

Tipos complexos podem ser definidos utilizando o construtor complexType. Tais definições geralmente contêm um nome, declarações de elementos e declarações de atributos.

Além das restrições de ocorrência declaradas para elementos individuais, a XML Schema provê restrições a serem aplicadas em grupos de elementos. Estas restrições também podem ser chamadas de delimitadores de grupos. Existem três tipos de delimitador de grupo:

- *sequence*, indica que os elementos do grupo devem aparecer no documento na mesma ordem em que foram declarados no esquema,
- *all*, estabelece que todos os elementos do grupo podem aparecer uma ou nenhuma vez, e que eles podem aparecer em qualquer ordem, e
- *choice*, indica que somente um dos elementos declarados no grupo pode aparecer no documento.

Considerando o tipo $T_{instituição_1}$, a sua definição (linhas 2.49 a 2.56) especifica que este tipo complexo contém os elementos $nome_1$, $telefone_1$ e $endereço_1$, que devem aparecer nesta ordem no documento, e o atributo $id\_instituição_1$.

Esquemas podem ser construídos a partir de definições de tipos complexos nomeados, tais como T_{autor_1} e $T_{endereço_1}$, e declarações de elementos, tais como $autor_1$ e $endereço_1$, que referenciam os tipos através do atributo type. Porém, um tipo complexo também pode ser definido de forma anônima, inserido na declaração de um elemento.

No esquema Bib_1, a declaração do elemento bib_1 (linhas 2.2 a 2.11) contém uma definição de tipo anônima, especificando que este elemento contém os elementos $artigo_1$, $livro_1$, $autor_1$ e $instituição_1$.

- ***Tipos Simples***

O esquema Bib_1 atribui tipos simples a vários atributos e elementos. Estes tipos simples[2], tais como, *Integer* e *String*, são tipos atômicos definidos na XML Schema. O valor de um tipo atômico é indivisível de acordo com a perspectiva da XML Schema.

Para manter a compatibilidade entre XML Schema e XML 1.0 DTDs, os tipos simples ID, IDREF e IDREFS, devem ser atribuídos apenas a atributos.

- ***Derivação de Tipos por Extensão***

Um tipo complexo também pode ser definido a partir da extensão de um outro tipo complexo ou de um tipo simples.

[2] Uma relação completa dos tipos simples definidos na XML Schema pode ser encontrada em [Fal01].

No esquema Bib_1, a definição do tipo $Tartigo_1$ (linhas 2.23 a 2.31) especifica que o tipo complexo $Tartigo_1$ é definido a partir do tipo complexo $Tpublicação_1$ (atributo base do elemento extension). Assim, um elemento do tipo complexo $Tartigo_1$ contém os elementos do tipo complexo $Tpublicação_1$, além do elemento $editora_1$. Neste caso, podemos dizer que $Tartigo_1$ é um subtipo (especialização) de $Tpublicação_1$.

- ***Restrições de Integridade***

XML Schema oferece, além do ID/IDREF/IDREFS, outros mecanismos de restrição de integridade, tais como: key e keyref. Estas restrições de integridade oferecem vantagens em relação ao ID/IDREF/IDREFS, tais como: (i) podem ser aplicadas tanto a elementos como a atributos; (ii) podem ser aplicadas a mais de um elemento ou atributo; (iii) permitem limitar o escopo, no qual o valor do elemento ou do atributo deve ser único, ou deve referenciar; e (iv) a restrição keyref permite capturar o tipo dos elementos ou dos atributos que estão sendo referenciados através do escopo da chave referenciada.

Para exemplificar a especificação de restrições de integridade na XML Schema, redefinimos o esquema XML Schema Bib_1. Esta nova versão é apresentada na Figura 3.5.

- ***Restrição de Integridade de Chave***

  XML Schema permite especificar, através do construtor key, que o valor de um conjunto de elementos ou atributos deve ser único dentro de um determinado escopo e, além de único, pode ser referenciado.

  Considere a restrição de chave key#3 (linhas 3.12 a 3.16). Esta restrição especifica que o valor dos elementos $nome_1$ e $e\text{-}mail_1$ é único dentro do escopo do conjunto de elementos $autor_1$ selecionados pela expressão de caminho $\$bib_1/autor_1$.

  A Figura 3.6 mostra como uma key é declarada. Como podemos observar, o construtor key contém:

  - um atributo name (N), que identifica unicamente a restrição.
  - um elemento selector (δ_1), que especifica o escopo, através de uma expressão de caminho, para o qual a restrição é declarada.

```
3.1 <schema>
3.2  <element name="bib1">
3.3   <key name="key#1">
3.4    <selector xpath="artigo1"/>
3.5    <field xpath="isbn1"/>
3.6    <field xpath="título1"/>
3.7   </key>
3.8   <key name="key#2">
3.9    <selector xpath="livro1"/>
3.10    <field xpath="isbn1"/>
3.11   </key>
3.12   <key name="key#3">
3.13    <selector xpath="autor1"/>
3.14    <field xpath="nome1"/>
3.15    <field xpath="e-mail1"/>
3.16   </key>
3.17   <key name="key#4">
3.18    <selector xpath="instituição1"/>
3.19    <field xpath="@id_instituição1"/>
3.20   </key>
3.21   <keyref name="keyref#1" refer="key#3">
3.22    <selector xpath="artigo1"/>
3.23    <field xpath="nome-autor1"/>
3.24    <field xpath="e-mail-autor1"/>
3.25   </keyref>
3.26   <keyref name="keyref#2" refer="key#3">
3.27    <selector xpath="livro1"/>
3.28    <field xpath="nome-autor1"/>
3.29    <field xpath="e-mail-autor1"/>
3.30   </keyref>
3.31   <keyref name="keyref#3" refer="key#4">
3.32    <selector xpath="autor1"/>
3.33    <field xpath="@instituiçãoref1"/>
3.34   </keyref>

3.35   <complexType>
3.36    <sequence>
3.37      <element name="artigo1" type="Tartigo1"
           minOccurs="0" maxOccurs="unbounded"/>
3.38      <element name="livro1" type="Tlivro1"
          minOccurs="0" maxOccurs="unbounded"/>
3.39      <element name="autor1" type="Tautor1"
          minOccurs="0"  maxOccurs="unbounded"/>
3.40      <element name="instituição1" type="Tinstituição1"
          minOccurs="0"  maxOccurs="unbounded"/>
3.41    </sequence>
3.42   </complexType>
3.43  </element>

3.44  <complexType name=" Tpublicação1">
3.45   <sequence>
3.46    <element name="ano1" type="Integer"/>
3.47    <element name="isbn1" type="String"/>
3.48    <element name="título1" type="String"/>
3.49    <element name="nome-autor1" type="String"
          minOccurs="1" maxOccurs="unbounded"/>
3.50    <element name="e-mail-autor1" type="String"
          minOccurs="1" maxOccurs="unbounded"/>
3.51   </sequence>
3.52  </complexType>

3.53  <complexType name="Tartigo1">
3.54   <complexContent>
3.55    <extension base="Tpublicação1" >
3.56     <sequence>
3.57      <element name="local-publicação1" type="String"/>
3.58     </sequence>
3.59    </extension>
3.60   </complexContent>
3.61  </complexType>

3.62  <complexType name="Tlivro1">
3.63   <complexContent>
3.64    <extension base="Tpublicação1">
3.65     <sequence>
3.66      <element name="editora1" type="String"/>
3.67     </sequence>
3.68    </extension>
3.69   </complexContent>
3.70  </complexType>

3.71  <complexType name="Tautor1">
3.72   <sequence>
3.73    <element name="nome1" type="String"/>
3.74    <element name="e-mail1" type="String"/>
3.75   </sequence>
3.76   <atribute name="instituiçãoref1" type="IDREF"/>
3.77  </complexType>

3.78  <complexType name="Tinstituição1">
3.79   <sequence>
3.80    <element name="nome1" type="String"/>
3.81    <element name="endereço1" type="String"/>
3.82   </sequence>
3.83   <atribute name="id_instituição1" type="ID"/>
3.84  </complexType>

3.85 </schema>
```

Figura 3.5: ***XML Schema de Bib1 com Restrições de Integridade***

- um ou mais elementos field (δ_{11}, ..., δ_{1n}), que determina(m), através de uma ou mais expressões de caminho, o conjunto de elementos ou atributos que deve ser único dentro do escopo especificado pelo selector.

```
<key name="N">
  <selector xpath="δ1">
  <field xpath="δ11">
        ...
  <field xpath="δ1n">
</key>
```

Figura 3.6: ***Declaração de Restrição de Chave***

- ***Restrição de Integridade Referencial***

  XML Schema permite declarar, através do construtor keyref, que o valor de um conjunto de elementos ou atributos é uma referência ao valor de um outro conjunto de elementos ou atributos.

  Considere a restrição referencial keyref#1 (linhas 3.21 a 3.25). Esta restrição especifica que, dentro do escopo do conjunto de elementos em $bib_1/artigo_1$, os elementos $nome\text{-}autor_1$ e $e\text{-}mail\text{-}autor_1$ destes elementos referenciam a chave key#3. Ou seja, para qualquer $p em $bib_1/artigo_1$, existe um $a em $bib_1/autor_1$ tal que $p/nome\text{-}autor_1 = a/nome_1$ e $p/e\text{-}mail\text{-}autor_1 = a/e\text{-}mail_1$.

  A Figura 3.7 mostra como uma keyref é declarada. O construtor keyref contém:

  - um atributo name (ref), que identifica unicamente a restrição.
  - um atributo refer (N), que indica a restrição de chave associada à referência.
  - um elemento selector (δ_2), que especifica o escopo, através de uma expressão de caminho, para o qual a restrição é declarada.
  - um ou mais elementos field (δ_{21}, ..., δ_{2n}), que determina(m), através de uma ou mais expressões de caminho, o conjunto de elementos ou atributos que deve corresponder ao conjunto de elementos ou atributos especificado pela restrição de chave correspondente.

```
<keyref name="ref" refer="N">
  <selector xpath="δ2">
  <field xpath="δ21">
        ...
  <field xpath="δ2n">
</keyref>
```

Figura 3.7: ***Declaração de Restrição de Integridade Referencial***

CAPÍTULO 4

XML SCHEMA SEMÂNTICO

Neste trabalho, usamos uma notação gráfica, denominada XML Schema Semântico (XMLS+), para representar os esquemas das fontes locais e o esquema do mediador definidos em XML Schema. XMLS+ permite representar graficamente a estrutura dos tipos dos esquemas XML Schemas e algumas formas de relacionamentos semânticos implícitos entre os tipos, tais como, especialização e associação. Desta forma, XMLS+ incorpora mais semântica aos esquemas XML Schemas o que facilita o processo de análise, comparação e identificação de correspondências entre esquemas XML Schemas. Neste Capítulo, descrevemos como gerar esquemas XMLS+ para esquemas XML Schemas, para esquemas relacionais e para esquemas orientados a objetos.

4.1 Gerando Esquemas XMLS+ para Esquemas XML Schemas

XML Schema Semântico (XMLS+) é uma notação gráfica baseada em árvore usada para representar graficamente esquemas XML Schemas. A representação gráfica facilita a visualização dos esquemas e favorece um melhor entendimento da semântica fornecida pelos mesmos. Uma vantagem da XMLS+ é a representação de algumas formas de relacionamentos implícitos entre tipos de esquemas XML Schema (tipos XML), tais como especialização e associação. O relacionamento de especialização entre tipos ocorre quando um tipo é definido a partir da extensão de outro tipo (tipo/subtipo). O relacionamento de associação entre tipos ocorre quando o conteúdo de um elemento ou atributo contido em um tipo complexo é uma referência a um elemento de um outro tipo complexo.

O processo de geração de esquemas XMLS+ é baseado em um conjunto de passos que consideram os tipos XML, juntamente com os seus elementos e atributos, a restrição de ocorrência dos elementos, a restrição de cardinalidade dos atributos, os delimitadores de grupo, os tipos derivados por extensão e as restrições de integridade.

Para exemplificar o processo de geração do esquema XMLS+ para um dado esquema XML Schema, utilizamos o esquema XML Schema Bib_1 apresentado na Figura 3.5. Este processo consiste de quatro passos, descritos a seguir:

Passo 1: *Geração das Árvores dos Tipos Complexos*
Para cada definição de tipo complexo é gerada uma árvore cujo nó raiz contém um nó filho para cada um dos elementos e atributos do tipo, o qual contém um único nó (terminal) que indica o tipo do elemento/atributo. No caso de definições de tipos complexos anônimas, também é gerada uma árvore de tipo que deve então ser nomeado.

As arestas que conectam o nó raiz aos nós correspondentes a elementos são rotuladas de acordo com a restrição de ocorrência dos elementos:

- Se o elemento for mono-ocorrência e obrigatório, a aresta não é rotulada
- Se o elemento for mono-ocorrência e opcional, a aresta é rotulada com o símbolo "?"
- Se o elemento for multi-ocorrência e opcional, a aresta é rotulada com o símbolo "*"
- Se o elemento for multi-ocorrência e obrigatório, a aresta é rotulada com o símbolo "+"

As arestas que conectam o nó raiz aos nós correspondentes a atributos são rotuladas de acordo com a restrição de cardinalidade dos atributos:

- Se o atributo for monovalorado e opcional, a aresta é rotulada com o símbolo "?"
- Se o atributo for monovalorado e obrigatório, a aresta não é rotulada

Em XMLS+, os atributos do tipo IDREFS (multivalorados), são tratados como atributos multi-ocorrência do tipo IDREF. Desta forma, as arestas que conectam o nó raiz aos nós correspondentes a estes atributos devem ser rotuladas de forma semelhante a dos elementos multi-ocorrência:

- Se o atributo for multivalorado e opcional, a aresta é rotulada com o símbolo "*"
- Se o atributo for multivalorado e obrigatório, a aresta é rotulada com o símbolo "+"

As arestas podem ser conectadas por um semicírculo. Isso indica que os elementos associados a estas arestas estão agrupados pelo delimitador de grupo *choice*. Um exemplo dessa situação é mostrado na Figura 5.4 do Capítulo 5.

A Figura 4.1 mostra o esquema XMLS+ preliminar de Bib_1. Este esquema será enriquecido com os relacionamentos semânticos entre os tipos nos próximos passos. O esquema XMLS+ da Figura 4.1 contém seis árvores de tipos, um para cada tipo do esquema XML Schema. Note que tipo T_{bib1} corresponde ao tipo anônimo do elemento bib_1.

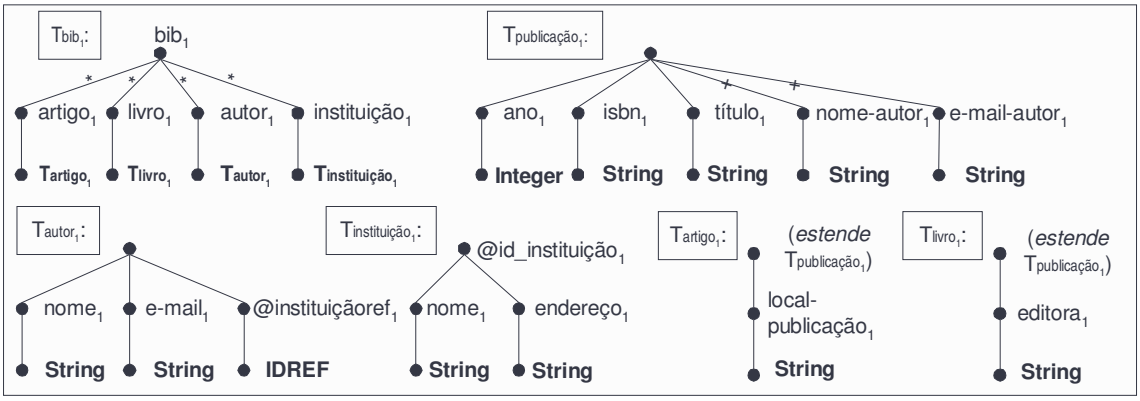


Figura 4.1: ***Esquema XMLS+ Preliminar de Bib1***

A árvore do tipo T_{bib1}, que é o tipo do elemento raiz bib_1, é a única árvore que tem o nó raiz rotulado (com o nome do elemento raiz). Isto porque, diferentes elementos podem ter o mesmo tipo complexo como conteúdo e apenas um elemento pode ter o tipo complexo do elemento raiz. A fonte em negrito indica o nome do tipo dos elementos e atributos. O prefixo "@" indica um nome de atributo.

Os relacionamentos de <u>especialização</u> entre tipos de um esquema XML Schema (tipo/subtipo) devem ser representados explicitamente. Por exemplo, os tipos T_{livro1} e $T_{artigo1}$ são especializações do tipo $T_{publicação1}$. T_{livro1} e $T_{artigo1}$ são representados graficamente como mostrado na Figura 4.1. Relacionamentos de associação entre tipos serão definidos nos passos seguintes.

Passo 2: *Definição de Relacionamentos de Associação entre Tipos Capturados por Atributos de Referência (IDREF)*

Os relacionamentos de associação entre tipos de um esquema XML Schema devem ser identificados e representados explicitamente no esquema XMLS+. Um atributo do tipo IDREF (atributo de referência) captura uma associação entre tipos, entretanto o tipo referenciado por este atributo não é definido pela declaração. No esquema XML Schema, uma keyref é usada para definir o tipo referenciado por um atributo IDREF (escopo de um atributo IDREF).

Em Bib_1, a restrição referencial keyref#3 especifica o escopo do atributo $@instituiçãoref_1$. Dado que o escopo da restrição keyref#3 é um conjunto de elementos do tipo $Tautor_1$, e o escopo da chave referenciada (key#4) é um conjunto de elementos do tipo $Tinstituição_1$, então, podemos dizer que o escopo do atributo $@instituiçãoref_1$ é um conjunto de elementos do tipo $Tinstituição_1$. Assim, temos que $Tautor_1$ *referencia* $Tinstituição_1$ através de $@instituiçãoref_1\rightarrow$ (O símbolo → é utilizado para indicar atributo de referência). Dado que $Tautor_1$ *referencia* $Tinstituição_1$ *através de* $@instituiçãoref_1\rightarrow$, então podemos dizer que $Tinstituição_1$ *referencia* $Tautor_1$ *através da inversa de* $@instituiçãoref_1\rightarrow$ $(@instituiçãoref_1\rightarrow)^{-1}$.

Desta forma, um atributo de referência e a sua inversa permitem capturar a existência de uma associação entre tipos. A representação da associação entre os tipos $Tautor_1$ e $Tinstituição_1$ é apresentada nas Figuras 4.2 e 4.3. O símbolo "&" é utilizado para indicar referência a um tipo. Como pode ser observado, é acrescentado ao tipo $Tinstituição_1$ o elemento $(@instituiçãoref_1\rightarrow)^{-1}$. Este nó é preenchido de branco para diferenciar dos nós correspondentes aos elementos contidos na definição do tipo do esquema XML Schema. No caso de um tipo conter mais de uma inversa com o mesmo nome, o nome do tipo referenciado deve ser acrescido ao nome da inversa, como por exemplo, $(Tautor_1.@instituiçãoref_1\rightarrow)^{-1}$. A restrição de ocorrência da inversa de $@instituiçãoref_1\rightarrow$ é capturada a partir da análise do relacionamento dos elementos do tipo $Tinstituição_1$ com os elementos do tipo $Tautor_1$. Como um elemento do tipo $Tinstituição_1$ pode conter um conjunto elementos $Tautor_1$, temos que a inversa de $@instituiçãoref_1\rightarrow$ é multi-ocorrência e opcional.

A vantagem de representarmos explicitamente as inversas dos atributos de referência é que permite definirmos caminhos nas duas direções do relacionamento. Desta forma, temos que um elemento do tipo $Tautor_1$ tem associado um elemento do tipo $Tinstituição_1$, e um elemento do tipo $Tinstituição_1$ tem associado um conjunto de elementos do tipo $Tautor_1$. Nós estendemos as expressões de

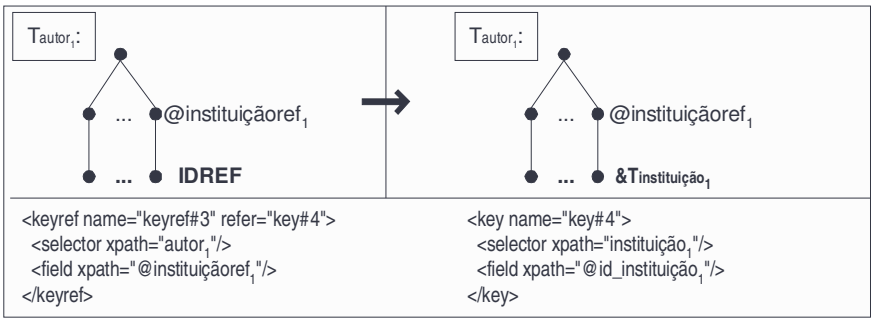


Figura 4.2: ***Tautor1 referencia Tinstituição1 através de @instituiçãoref1 →***

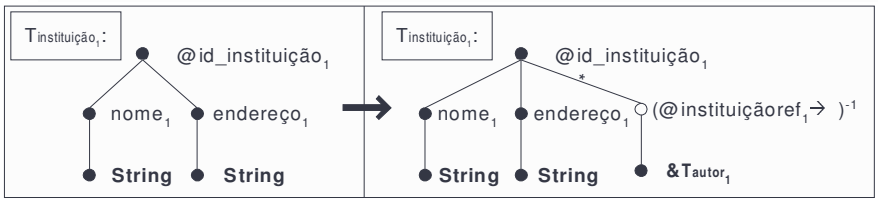


Figura 4.3: ***Tinstituição1 referencia Tautor1 através da inversa de @instituiçãoref1→***

caminho de XPath para permitir a navegação nas duas direções do relacionamento, como mostrado nos exemplos a seguir:

Exemplo 4.1: suponha $a um nó em bib_1$/autor$_1$. A expressão de caminho $a/@instituiçãoref$_1$→ retorna o elemento instituição$_1$ associado ao autor $a, i.é, o elemento $i em bib_1$/instituição$_1$ tal que $i/@id_instituição$_1$ = $a/@instituiçãoref$_1$→.

Exemplo 4.2: suponha $i em bib_1$/instituição$_1$. A expressão de caminho $i/(@instituiçãoref$_1$ →)$^{-1}$ retorna os elementos autor$_1$ associados à instituição $i, i.é, todos os elementos $a em bib_1$/autor$_1$ tais que $i/(@instituiçãoref$_1$ →)$^{-1}$ = $a/@instituiçãoref$_1$.

Passo 3: *Representação de outras Associações entre Tipos*

No esquema XML Schema, podem existir keyrefs que não estejam associadas a atributos do tipo IDREF. Estas keyrefs também capturam a associação entre tipos. Em Bib$_1$, a restrição referencial keyref#1 especifica que dentro do escopo do conjunto de elementos em bib_1$/artigo$_1$, os elementos nome-autor$_1$ e e-mail-autor$_1$ destes elementos referenciam a chave key#3. Mais formalmente, para qualquer $p em bib_1$/artigo$_1$, existe um $a em bib_1$/autor$_1$ tal que $p/nome-autor$_1$ = $a/nome$_1$ e $p/e-mail-autor$_1$ = $a/e-mail$_1$. Como o escopo da restrição referencial keyref#1 é um conjunto de elementos do tipo Tartigo$_1$, e o escopo da chave referenciada (key#3) é um conjunto de elementos do tipo Tautor$_1$, nós dizemos que Tartigo$_1$ *referencia* Tautor$_1$ *através de* keyref#1. Dado que Tartigo$_1$ *referencia* Tautor$_1$ *através de* keyref#1, então podemos dizer que Tautor$_1$ *referencia* Tartigo$_1$ *através da inversa de* keyref#1 (keyref#1)$^{-1}$. Desta forma, uma keyref e a sua inversa também permitem capturar a existência de uma

associação entre tipos. A representação da associação entre os tipos T_{artigo_1} e T_{autor_1} é apresentada na Figura 4.4 e 4.5. Neste caso, T_{artigo_1} contém um elemento keyref#1 cujo tipo é uma referência para T_{autor_1} (&T_{autor_1}) e o T_{autor_1} contém um elemento $(keyref\#1)^{-1}$ cujo tipo é uma referência a T_{artigo_1} (&T_{artigo_1}).

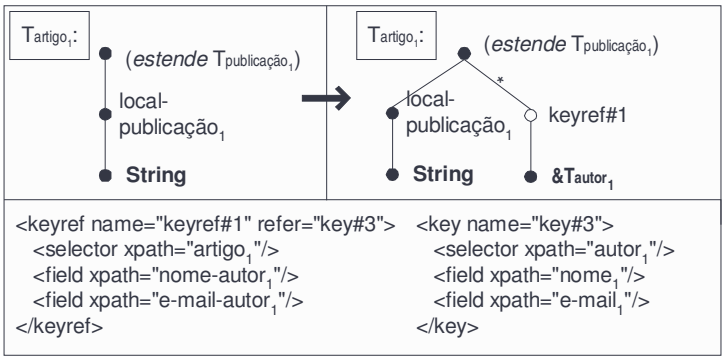


Figura 4.4: ***T_{artigo_1} referencia T_{autor_1} através de keyref#1***

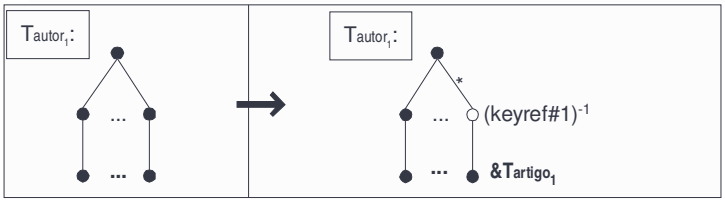


Figura 4.5: ***T_{autor_1} referencia T_{artigo_1} através da inversa de keyref#1***

Nós estendemos expressões de caminho de XPath para navegarmos através de uma keyref ou inversa de keyref, como mostrado nos exemplos a seguir:

Exemplo 4.3: suponha $p um nó em $bib_1/artigo_1$. A expressão de caminho $p/keyref#1 retorna os elementos $autor_1$ associados ao artigo $p, i.é, todos $a em $bib_1/autor_1$, tais que $a/nome_1 = $p/nome-autor_1$ e $a/e-mail_1 = $p/e-mail-autor_1$.

Exemplo 4.4: suponha $a um nó em $bib_1/autor_1$. A expressão de caminho $a/(keyref#1)^{-1}$ retorna os elementos $artigo_1$ associados ao autor $a, i.é, todos $p em $bib_1/artigo_1$, tais que $a/nome_1 = $p/nome-autor_1$ e $a/e-mail_1 = $p/e-mail-autor_1$.

No caso em que uma keyref é composta de apenas um elemento field, a representação é semelhante à representação de atributo de referência, com uma pequena adaptação. Para exemplificar este caso, suponha que o tipo T_{autor_1} referencia o tipo T_{livro_1} através da keyref#4 composta de apenas um elemento field $livro_1$, como mostrado na Figura 4.6. Note que o nó do

elemento $livro_1$ é representado com dois círculos indicando que este nó está associado a uma keyref, neste caso, a keyref#4.

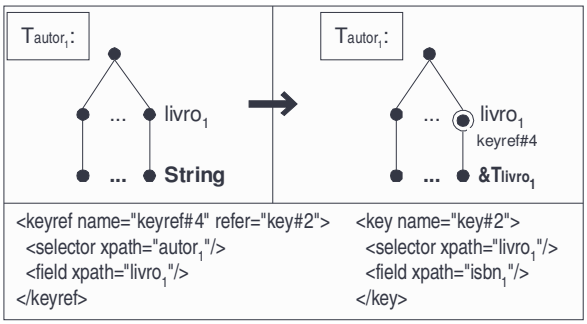


Figura 4.6: ***Tautor1 referencia Tlivro1 através da keyref#4***

Como veremos no Capítulo 6, a principal vantagem de representarmos as keyrefs e suas inversas no esquema XMLS+ é podermos identificar e especificar correspondências entre caminhos dos tipos com representações diferentes.

A Figura 4.7 mostra o esquema XMLS+ final correspondente a Bib_1, o qual será utilizado no estudo de caso do Capítulo 5. É importante observar que no esquema XMLS+ são representadas apenas as informações do esquema XML Schema que são relevantes para o processo de *matching*.

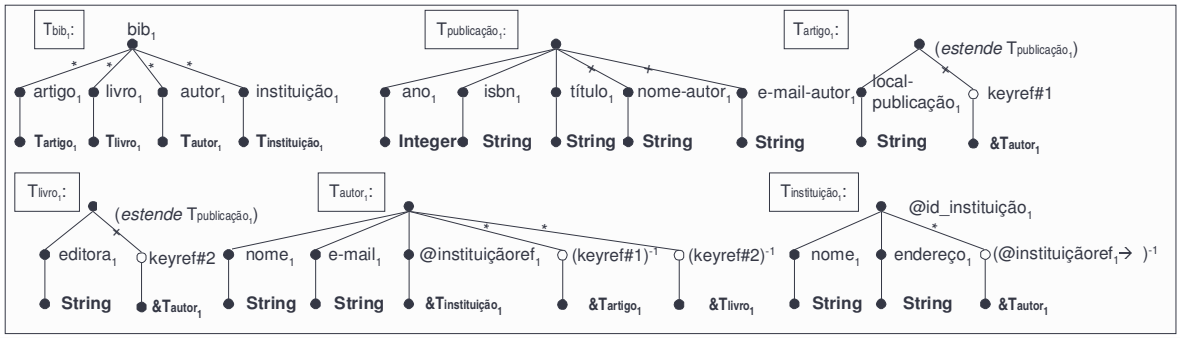


Figura 4.7: ***Esquema XMLS+ Final de Bib1***

Para ilustrar a expressividade da XMLS+, nas Seções seguintes descrevemos a geração de esquemas XMLS+ para esquemas relacionais e esquemas orientados a objetos.

4.2 Gerando Esquemas XMLS+ para Esquemas Relacionais

O modelo relacional representa um banco de dados como uma coleção de relações onde uma relação contém um conjunto de tuplas [EN00]. Cada tupla em uma relação representa uma entidade do mundo real e obedece a uma especificação estrutural, a qual consiste em uma lista de atributos pertencentes a esta relação. Existem diversas restrições que podem ser especificadas nas relações, tais como as chaves primárias e as chaves estrangeiras. A chave primária é usada para identificar unicamente cada tupla de uma relação. A chave estrangeira especifica restrições referenciais entre relações.

O processo de geração de esquemas XMLS+ para esquemas relacionais é baseado em um conjunto de passos que consideram as relações, juntamente com os seus atributos, o domínio dos atributos, as chaves primárias e as chaves estrangeiras.

A seguir, mostramos o processo de geração de esquemas XMLS+ para esquemas relacionais. Para exemplificar este processo, considere o esquema relacional Bib_2, cujas relações e seus respectivos atributos são mostrados na Figura 4.8.

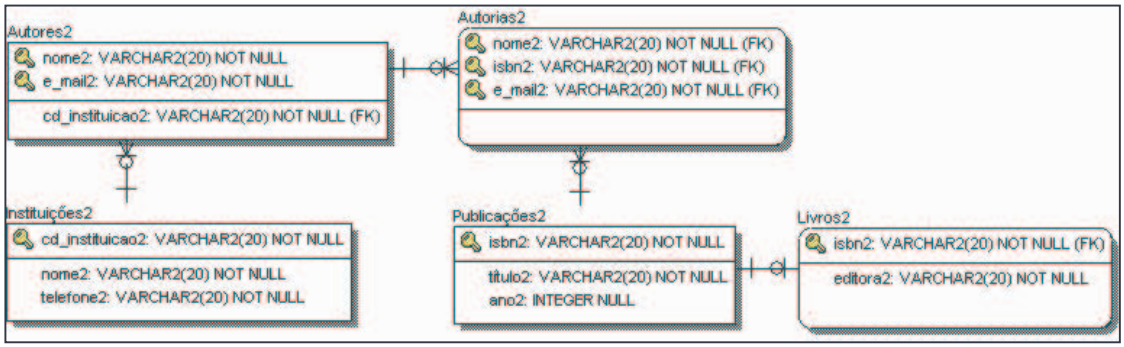


Figura 4. 8: ***Esquema Relacional da Fonte Bib2***

No restante desta Seção, considere o esquema relacional $S_1 = \{R_1, R_2, ..., R_m\}$, tal que R_1, R_2,..., R_m são relações de S_1. O processo de geração do esquema XMLS+ para o esquema relacional S_1 consiste de três passos, descritos a seguir.

Passo 1: *Geração do Tipo Raiz*

O tipo raiz $Traiz_{S1}$, como mostrado na Figura 4.9, conterá um elemento R_i do tipo TR_i, mono-ocorrência e obrigatório para cada relação R_i de S_1. O tipo TR_i conterá apenas um elemento $tupla\_R_i$ do tipo $Ttupla\_R_i$, multi-ocorrência e opcional. A Figura 4.10 mostra o tipo raiz do esquema Bib_2.

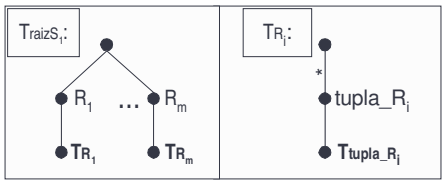


Figura 4.9: ***Gerando Esquemas XMLS+ para Esquemas Relacionais - Passo 1***

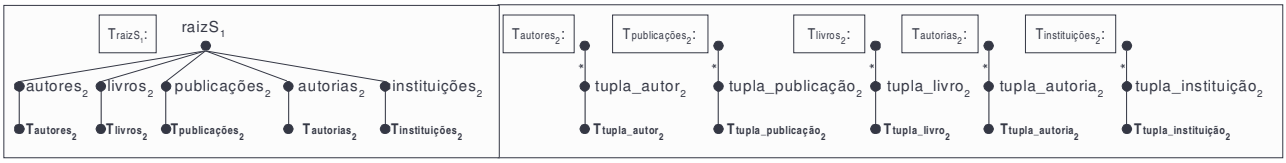


Figura 4.10: ***Gerando Esquema XMLS+ para Bib2 - Passo 1***

É importante notar que as relações $Autores_2$, $Livros_2$, $Publicações_2$, $Autorias_2$ e $Instituições_2$, correspondem às coleções de elementos definidas pelas expressões de caminho $\$bib_2/autores_2/tupla\_autor_2$, $\$bib_2/livros_2/tupla\_livro_2$, $\$bib_2/publicações_2/tupla\_publicação_2$, $\$bib_2/autorias_2/tupla\_autoria_2$ e $\$bib_2/instituições_2/tupla\_instituição_2$, respectivamente. Estas coleções são chamadas de *coleções globais*, e receberão tratamento especial no processo de *matching*, como discutiremos no Capítulo 5.

Passo 2: *Geração dos Tipos $T_{tupla\_R_1}$, ..., $T_{tupla\_R_m}$*

O tipo $T_{tupla\_R_i}$ conterá um elemento para cada atributo de R_i. O tipo deste elemento deve ser um tipo compatível com um tipo simples da XML Schema. A restrição de ocorrência do elemento é definida de acordo com a cardinalidade do atributo: se o atributo estiver associado a uma restrição de não-nulo (*not null*), o elemento é mono-ocorrência e obrigatório, caso contrário o elemento é mono-ocorrência e opcional.

A Figura 4.11 mostra os tipos gerados neste passo para o esquema Bib_2.

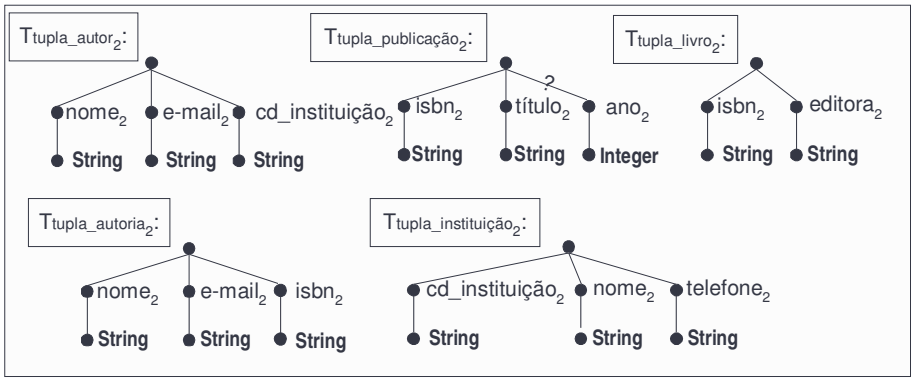


Figura 4.11: ***Gerando Esquema XMLS+ para Bib_2 – Passo 2***

Passo 3: *Representação dos Relacionamentos Capturados pelas Chaves Estrangeiras*

As chaves estrangeiras devem ser analisadas para identificar o tipo do relacionamento semântico que estas representam. Uma chave estrangeira pode capturar um relacionamento de especialização ou associação, conforme os casos abaixo.

Suponha que a relação R_j referencia a relação R_k através de uma chave estrangeira FK.

Caso 1 - FK é chave de R_j: Se FK for chave de R_j, então existe um relacionamento de especialização entre $T_{tupla\_R_j}$ e $T_{tupla\_R_k}$. Desta forma, este relacionamento é representado graficamente no esquema

XMLS+ e o(s) elemento(s) do tipo $T_{tupla\_R_j}$ correspondente(s) a chave estrangeira FK é(são) eliminado(s).

Considere, por exemplo, a chave estrangeira FK#1:($Livros_2$ [$isbn_2$] ⊆ $Publicações_2$ [$isbn_2$]) definida na relação $Livros_2$ do esquema Bib_2. Neste caso, dado que $isbn_2$ também é chave de $Livros_2$, então existe um relacionamento de especialização entre os tipos $T_{tupla\_publicação_2}$ e $T_{tupla\_livro_2}$, como mostrado na Figura 4.12. Além disso, a coleção global $bib_2/livros_2/tupla\_livro_2$ é um subconjunto da coleção $bib_2/publicações_2/tupla\_publicação_2$. Esta restrição é especificada através de uma assertiva de correspondência de subconjunto definida no capítulo 5.

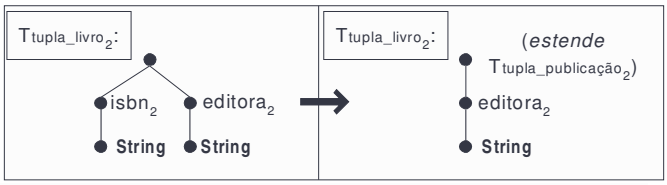


Figura 4.12: ***Ttupla_autor2 estende Ttupla_publicação2***

Caso 2 - FK não é chave de R_j: Se FK não for chave de R_j, então existe um relacionamento de associação entre $T_{tupla\_R_j}$ e $T_{tupla\_R_k}$.

Considere, por exemplo, a chave estrangeira FK#2:($Autores_2$ [$cd\_instituição_2$] ⊆ $Instituições_2$ [$cd\_instituição_2$]) definida na relação $Autores_2$ do esquema Bib_2. Neste caso, dado que $cd\_instituição_2$ não é chave de $Autores_2$, então existe um relacionamento de associação entre os tipos $T_{tupla\_autor_2}$ e $T_{tupla\_instituição_2}$. A representação da restrição referencial e inversa é similar à representação de keyref e inversa, discutida na Seção 4.1. A Figura 4.13 mostra a representação da chave estrangeira FK#2 e de sua inversa $(FK\#2)^{-1}$. A restrição de ocorrência de $(FK\#2)^{-1}$ deve ser capturada a partir da análise do relacionamento dos elementos do tipo $T_{tupla\_instituição_2}$ com os elementos do tipo $T_{tupla\_autor_2}$. Assim, $(FK\#2)^{-1}$ é multi-ocorrência e opcional.

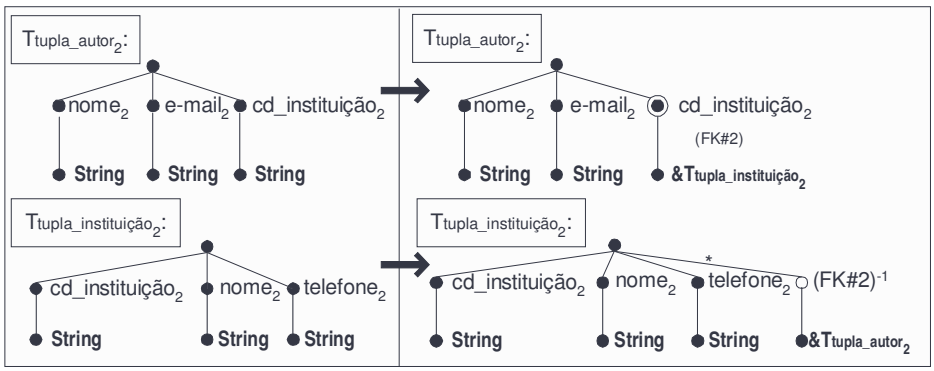


Figura 4.13: ***Ttupla_autor2 referencia Ttupla_instituição2 através de FK#2***

A Figura 4.14 mostra o esquema XMLS+ final de Bib_2, o qual será utilizado no estudo de caso do Capítulo 5.

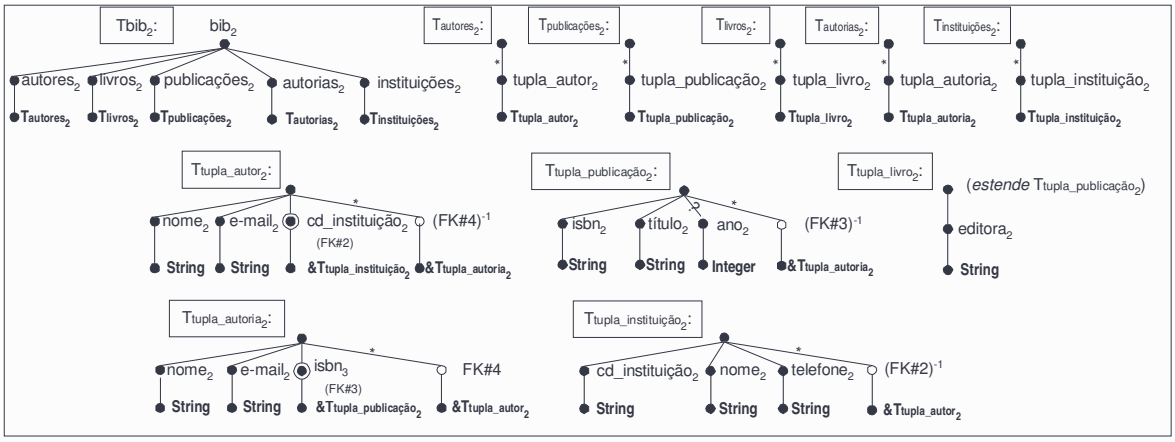


Figura 4.14: ***Esquema XMLS+ Final para Bib2***

4.3 Gerando Esquemas XMLS+ para Esquemas Orientados a Objetos

Os elementos básicos de um modelo Orientado a Objetos (OO) de acordo com a ODMG [CBB+97] são os objetos, os literais e as classes:

- Os objetos representam entidades do mundo real e possuem um identificador único, chamado *Object Identifier* (OID). Um objeto tem estado e comportamento. O estado de um objeto é definido pelos valores que o mesmo possui para um conjunto de propriedades (atributos e relacionamentos). As propriedades podem ser monovaloradas ou multivaloradas. O comportamento de um objeto é definido por um conjunto de operações (métodos) que podem ser executadas nos objetos;
- Um literal é um tipo especial de objeto que não possui um identificador;
- Uma classe agrupa objetos ou literais (chamada *classe de objetos* ou *classe literal*) que possuem propriedades e operações comuns. Algumas classes podem ter uma extensão, que consiste em um conjunto de objetos que são instâncias da classe. Além disso, uma classe pode herdar o estado e o comportamento de uma outra classe (relacionamento de especialização entre classes).

O processo de geração de esquemas XMLS+ para esquemas orientados a objetos é baseado em um conjunto de passos que consideram as classes, juntamente com seus atributos e relacionamentos, o domínio e restrição de cardinalidade dos atributos e dos relacionamentos, as extensões de classes e os relacionamentos de especialização entre classes.

A seguir, mostramos o processo de geração de esquemas XMLS+ para um esquema orientado a objetos. Para exemplificar este processo, considere o esquema orientado a objetos Bib_3, cujas classes e seus respectivos atributos e relacionamentos são mostrados na Figura 4.15.

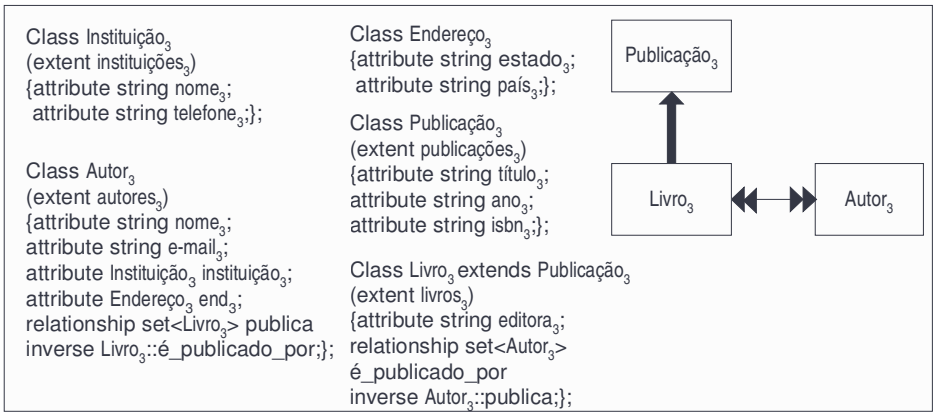


Figura 4.15: ***Esquema Orientado a Objetos da Fonte Bib3***

No restante desta Seção, considere $S_1 = \{C_1, C_2, ..., C_n\}$, tal que $C_1, C_2, ..., C_n$ são classes de S_1. Sejam $Ext_1, Ext_2, ..., Ext_m$ nomes de extensões de classes em S_1.

Passo 1: *Geração dos Tipos Complexos das Classes*

Para cada classe C_i de S_1, será definido um tipo TC_i, o qual conterá um elemento para cada atributo de C_i. A restrição de ocorrência do elemento é definida de acordo com a cardinalidade do atributo: se o atributo for monovalorado, o elemento é mono-ocorrência, caso contrário o elemento é multi-ocorrência. O tipo do elemento é definido de acordo com o domínio do atributo:

- Se o domínio do atributo for uma classe literal, então o tipo do elemento será o tipo XML correspondente, o qual pode ser predefinido ou um tipo complexo correspondente a classe literal;
- Se o domínio do atributo for uma classe de objetos, então o tipo do elemento será uma referência ao tipo XML correspondente. Neste caso, a inversa do atributo de referência deve ser especificada no esquema. A restrição de ocorrência desta inversa deve ser capturada a partir da análise do relacionamento entre os elementos dos tipos.

Além de conter os atributos, o tipo TC_i conterá um elemento para cada relacionamento de C_i, cujo o tipo será uma referência ao tipo correspondente à classe de objetos que participa do

relacionamento. A restrição de ocorrência do elemento é definida de acordo com a cardinalidade do relacionamento: se o relacionamento for monovalorado, o elemento é mono-ocorrência, caso contrário o elemento é multi-ocorrência.

Os relacionamentos de <u>especialização</u> entre classes de S_1 devem ser representados explicitamente. Se uma classe C_i for uma subclasse da C_j, então existe um relacionamento de especialização entre os tipos TC_i e TC_j. Este relacionamento é representado conforme descrito na Seção 4.1.

A Figura 4.16 mostra os tipos gerados neste passo para o esquema Bib_3.

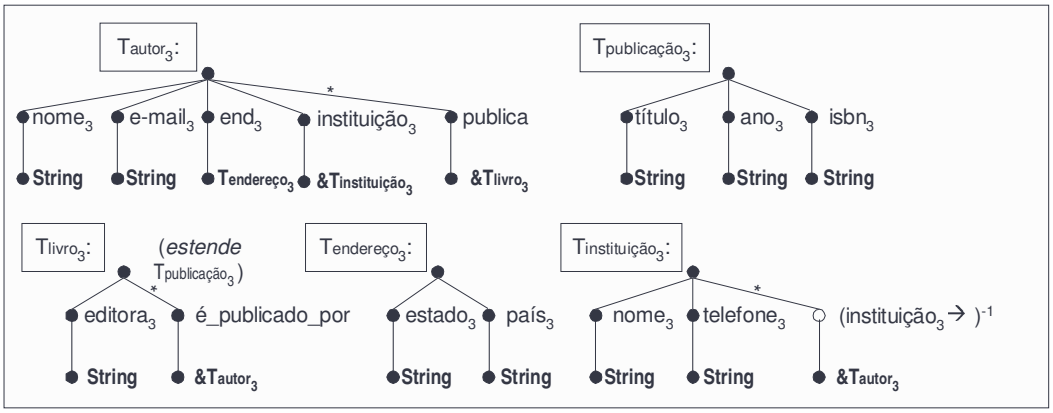


Figura 4.16: ***Gerando Esquema XMLS+ para Bib3 – Passo 1***

<u>Passo 2</u>: *Geração do Tipo Raiz*

O tipo raiz $Traiz_{S1}$, como mostrado na Figura 4.17, conterá um elemento Ext_i do tipo $TExt_i$, mono-ocorrência e obrigatório para cada extensão de classe Ext_i de S_1. O tipo $TExt_i$ conterá apenas um elemento C_i do tipo TC_i, multi-ocorrência e opcional. A Figura 4.18 mostra o tipo raiz do esquema Bib_3.

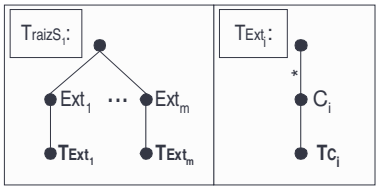


Figura 4.17: ***Gerando Esquemas XMLS+ para Esquemas Orientados a Objetos – Passo 2***

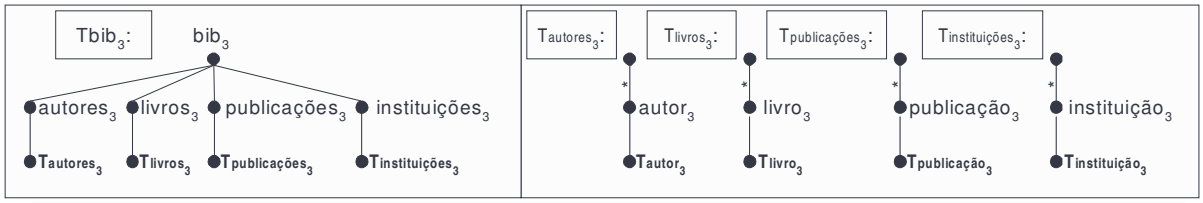


Figura 4.18: ***Gerando Esquema XMLS+ para Bib3 – Passo 2***

Note que as extensões de classes $Autores_3$, $Livros_3$, $Publicações_3$ e $Instituições_3$, correspondem às coleções de elementos definidas pelas expressões de caminho $\$bib_3/autores_3/autor_3$, $\$bib_3/livros_3/livro_3$, $\$bib_3/publicações_3/publicação_3$ e $\$bib_3/instituições_3/instituição_3$, respectivamente. Estas coleções são chamadas de *coleções globais*, e receberão tratamento especial no processo de *matching*, como discutiremos no Capítulo 5.

É importante notar que como a classe $Livro_3$ é uma subclasse da $Publicação_3$, então temos que a extensão da classe $Livro_3$ ($Livros_3$) é um subconjunto da extensão da classe $Publicação_3$ ($Publicações_3$). Dessa forma, no esquema XMLS+, a coleção global $\$bib_3/livros_3/livro_3$, correspondente a $Livros_3$, é um subconjunto da coleção $\$bib_3/publicações_3/publicação_3$, correspondente a $Publicação_3$. Esta restrição é especificada através de uma assertiva de correspondência de subconjunto definida no Capítulo 5.

CAPÍTULO 5

MATCHING DE ESQUEMAS XMLS+

Neste Capítulo, descrevemos o método para *matching* de esquemas XMLS+, denominado XMLS+Matcher. Na Seção 5.1, definimos o problema do *matching* de esquemas e a sua complexidade. Na Seção 5.2, definimos os conceitos preliminares que são necessários para a definição formal dos vários tipos de correspondência entre esquemas XMLS+. Na Seção 5.3, apresentamos um formalismo para especificar correspondências entre esquemas XMLS+. Na Seção 5.4, descrevemos o método proposto. Na Seção 5.5, aplicamos o método a um estudo de caso. E, na Seção 5.6, mostramos, através de um exemplo, como as correspondências geradas podem ser utilizadas para a reformulação de consultas.

5.1 Introdução

Nos últimos anos, o número de sistemas que requisitam acesso integrado a múltiplas fontes de informação heterogêneas, autônomas e distribuídas tem aumentado consideravelmente. Integrar informações de múltiplas fontes é uma tarefa muito complexa. Uma das razões para tal complexidade é a heterogeneidade semântica (ou *relativismo semântico*) existente entre as fontes a serem integradas. A heterogeneidade semântica expressa a multiplicidade de possíveis representações de um mesmo conceito do mundo real, dificultando a identificação das várias correspondências que existem entre conceitos similares ou relacionados nos diferentes esquemas [MRJ99, ST98]. As diferentes representações de um mesmo conceito podem ocorrer, principalmente, porque projetistas têm diferentes percepções da realidade. Os tipos mais comuns de heterogeneidade semântica são a heterogeneidade terminológica, extensional, de modelo e estrutural:

- Heterogeneidade terminológica expressa a diferença no significado ou na interpretação dos mesmos conceitos, conceitos relacionados ou conceitos diferentes representados nos esquemas a serem integrados. As causas mais comuns para tal heterogeneidade são quando termos diferentes são usados para representar o mesmo conceito do mundo real (termos sinônimos) ou quando os mesmos termos são usados para representar conceitos diferentes do mundo real (termos homônimos). Por exemplo, um esquema utiliza o termo *estudante*, um outro esquema, o termo *aluno* e um terceiro esquema, o termo *universitário* para armazenar informações sobre pessoas que estudam em universidades. Neste caso, termos sinônimos são usados para expressar o mesmo conceito do mundo real. Em um outro exemplo, o esquema do setor de produção de uma empresa utiliza o termo *equipamento* para manter informações sobre os equipamentos utilizados pela empresa, e o esquema do setor de vendas desta mesma empresa utiliza o termo *equipamento* para disponibilizar informações sobre os equipamentos vendidos pela empresa. Neste caso, termos iguais são usados para representar diferentes conceitos.

- Heterogeneidade extensional expressa diferença entre as coleções de objetos similares nas diferentes fontes de informação. Por exemplo, se existirem duas coleções *estudante* em duas fontes distintas, então estas coleções podem conter objetos correspondendo ao mesmo objeto do mundo real ou podem referenciar conjuntos disjuntos de objetos do mundo real. Identificar o relacionamento extensional entre as coleções de objetos de diferentes fontes é fundamental para a integração de informações.

- Heterogeneidade de modelo ocorre quando diferentes fontes de informação são modeladas através de diferentes modelos de dados. Como cada modelo de dados provê um conjunto próprio de construtores de modeladgem, um determinado conceito pode ser modelado através de diferentes construtores. Por exemplo, um mesmo conceito pode ser modelado como relação em um banco de dados relacional e como objeto em um banco de dados orientado a objetos.

- Mesmo quando um único modelo de dados é utilizado em dois esquemas distintos, diferentes construtores podem ser aplicados para representar um mesmo conceito. Neste caso, é identificada uma heterogeneidade estrutural entre os conceitos. Por exemplo, o atributo *posição* do tipo *empregado* em um esquema é modelado como as entidades subtipo *secretária* e *gerente* do supertipo *empregado* em outro esquema.

Como vimos no Capítulo 2, a maioria dos sistemas de integração disponibiliza uma visão integrada e transparente das informações das fontes locais que fazem parte do sistema, independente das heterogeneidades identificadas. Neste caso, o sistema de integração utiliza as correspondências entre o esquema da visão integrada (esquema do mediador) e os esquemas locais para: (i) determinar como as consultas definidas no esquema do mediador serão respondidas a partir de consultas nas fontes locais (enfoque virtual); ou (ii) atualizar corretamente o esquema do mediador, de forma a refletir as atualizações ocorridas nas fontes (enfoque materializado).

Como podemos observar, as correspondências entre o esquema do mediador e os esquemas das fontes locais desempenham um papel central nos sistemas de integração de informações. O processo de identificar correspondências entre os esquemas é denominado de *matching de esquemas* [RB01]. *Matching* de esquemas é um processo que analisa e compara dois esquemas para identificar elementos correspondentes entre os mesmos. Assim, este processo é responsável por identificar se as informações disponibilizadas em um esquema estão (e como estão) relacionadas com as informações disponibilizadas em um outro esquema.

Matching de esquemas é um problema complexo e que consome tempo, principalmente, devido à heterogeneidade semântica existente entre os esquemas das fontes de informação que não foram, originalmente, projetadas para trabalharem juntas. Além disso, os construtores de modelagem utilizados para representar os conceitos de uma fonte nem sempre capturam a real semântica da mesma. Conseqüentemente, o processo de *matching* requer uma forte interação com projetistas para entender a semântica das fontes de informação. Isto também significa que este processo não pode ser completamente automatizado [ERS99]. Entretanto, abordagens envolvendo metodologias e ferramentas vêm sendo propostas com o intuito de reduzir a interação humana [BBVC98, BLN86, CGMH94, ERS99, Mir97].

Com a explosão da Web como um ambiente para desenvolver sistemas de integração de informações, o problema de *matching* de esquemas ganhou um novo desafio: o nível de irregularidade da estrutura dos dados na Web. Assim, abordagens já existentes para *matching* de esquemas devem ser adaptadas e novas abordagens devem ser propostas [CA97, DDH01, DRRS+02, MBR01, MCH02, RB01].

Neste Capítulo, apresentamos o método para *matching* de esquemas XMLS+, denominado XMLS+Matcher. Como discutido anteriormente, antes de iniciar o processo de *matching*, os esquemas

são mapeados em XMLS+, proposto neste trabalho. XMLS+Matcher está inserido no *Ambiente para Geração e Manutenção de Mediadores*, descrito no Capítulo 2. No entanto, este método pode ser utilizado em qualquer sistema de integração de informações, independente dos enfoques adotados (virtual ou materializado).

5.2 Terminologia

Nesta Seção, apresentamos algumas definições necessárias para a definição formal das ACs formalizadas na Seção 5.3.

Definição 5.1: XMLS+ usa o conceito de *ligação* para representar relacionamentos entre tipos. Considere T_1 e T_2 tipos de um esquema XMLS+, onde T_1 é um tipo complexo. Existe uma ligação de T_1 para T_2 nas seguintes situações (vide Figura 5.1):
(i) (Ligação de Atributo) T_1 contém um atributo *a* cujo tipo é T_2. Assim, *a*: $T_1 \rightarrow T_2$ é uma ligação de T_1 para T_2 (vide Figura 5.1(a)).
(ii) (Ligação de Elemento) T_1 contém um elemento *e* cujo tipo é T_2. Assim, *e*: $T_1 \rightarrow T_2$ é uma ligação de T_1 para T_2 (vide Figura 5.1(b)).
(iii) (Ligação de Atributo de Referência) T_1 contém um atributo *a* do tipo IDREF que referencia o tipo T_2 ($\&T_2$). Assim, $a \rightarrow: T_1 \rightarrow T_2$ é uma ligação de T_1 para T_2 (vide Figura 5.1(c)).
(iv) (Ligação de Restrição Referencial (keyref)) T_1 contém uma keyref k tal que T_1 referencia T_2 através de k. Assim, k: $T_1 \rightarrow T_2$ é uma ligação de T_1 para T_2 (Quando a keyref é composta de apenas um elemento, esta é representada como mostra a Figura 5.1(d), caso contrário, é representada como mostra a Figura 5.1(e)).
(v) (Inversa de Ligação) T_2 tem uma ligação l: $T_2 \rightarrow T_1$ tal que *l* é uma ligação de Elemento, de Atributo de Referência ou de Restrição Referencial. Então, a inversa da ligação l, dada por l^{-1}: $T_1 \rightarrow T_2$, é uma ligação de T_1 para T_2 (vide Figura 5.1(f)).

Definição 5.2: (*Caminho de tipo*) Considere as ligações: l_1: $T_1 \rightarrow T_2$, l_2: $T_2 \rightarrow T_3$, …, l_{n-1}: $T_{n-1} \rightarrow T_n$, onde T_1, …, T_n são tipos de um esquema XML Schema. Assim sendo, $l_1/\ l_2\ /\ldots/\ l_{n-1}$ é um caminho de T_1.

Definição 5.3: (*Tipo e Restrição de Ocorrência do Caminho*) Considere $\delta = l_1/\ l_2\ /\ldots/\ l_{n-1}$ um caminho de T_1. Dada uma instância $e de T_1, a expressão de caminho $e/δ seleciona um conjunto

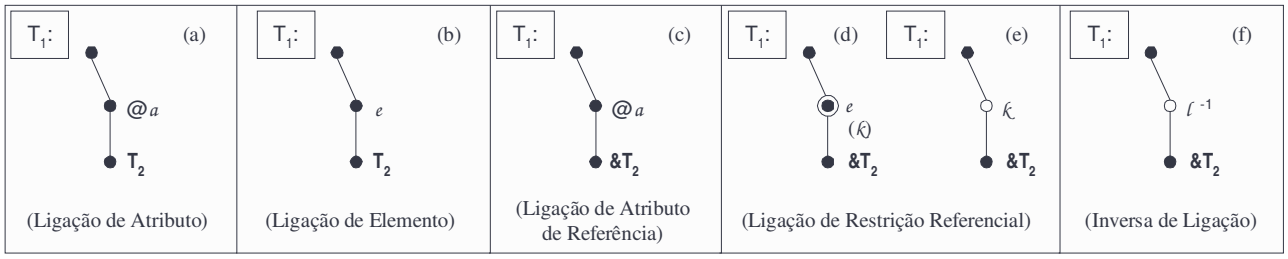


Figura 5.1: ***Tipos de Ligação de T_1 para T_2***

de elementos do tipo T_n. Assim, o tipo do caminho δ ($T_δ$) é T_n. Se as ligações l_1, l_2,... e l_n forem mono-ocorrência, então δ é mono-ocorrência, caso contrário δ é multi-ocorrência.

Definição 5.4: (*Coleção Global e Coleção Aninhada*) Suponha $t uma instância do tipo T e δ um caminho multi-ocorrência de T. A coleção de elementos retornados pela expressão de caminho $t/δ é uma coleção de $t. Se $t é o elemento raiz então $t/δ é uma coleção global, caso contrário é uma coleção aninhada.

Como discutido no Capítulo 4, as extensões das classes no modelo orientado a objetos, assim como as relações no modelo relacional correspondem às coleções globais no XMLS+, como veremos na Seção 5.5, estas coleções terão tratamento especial no XMLS+Matcher.

No restante deste trabalho, assumimos que dada uma fonte denominada S então existe uma variável global denominada $S que referencia o elemento raiz de S.

5.3 Assertivas de Correspondência em XMLS+

Para estabelecer as correspondências entre esquemas utilizamos as Assertivas de Correspondência (AC), que são tipos especiais de restrição de integridade usados para especificar que a semântica de algumas partes de um esquema está de alguma forma relacionada com a semântica de algumas partes de outro(s) esquema(s).

As ACs já foram definidas em diversos modelos: em [Lós98] foram definidas no modelo Entidades e Relacionamentos [Che76] e em [Peq00], no modelo Orientado a Objetos [CBB+97]. Neste trabalho, definimos as ACs em XMLS+, as quais são classificadas em: (i) AC de Coleções Globais; (ii) AC de Caminhos; e (iii) AC de Elementos.

A Tabela 5.1 relaciona as ACs especificadas em XMLS+ com estas no modelo Orientado a Objetos, modelo Relacional e modelo Entidades e Relacionamentos.

XMLS+	Orientado a Objetos	Relacional	ER
AC de Coleções Globais	AC de Extensão	AC de Relações	AC de Tipos de Entidades e AC de Tipos de Relacionamentos
AC de Caminhos	AC de Propriedades e AC de Caminhos	AC de Atributos	AC de Atributos e AC de Caminhos
AC de Elementos	---	---	---

Tabela 5.1: ***Relação entre AC Especificadas em XMLS+ e em Outros Modelos***

5.3.1 Assertivas de Correspondência de Coleções Globais

As ACs de Coleções Globais (ACCGs) especificam relacionamentos existentes entre coleções globais de um ou mais esquemas. Como vimos na Tabela 5.1, estas assertivas correspondem às assertivas de extensão no modelo Orientado a Objetos, de relações no modelo Relacional, de tipos de entidades e de tipos de relacionamentos no modelo Entidades e Relacionamentos. No restante desta Seção, considere T_1 e T_2 tipos raízes dos esquemas das fontes S_1 e S_2, respectivamente. Considere também δ_1 e δ_2 caminhos multi-ocorrência de T_1 e T_2, respectivamente.

A Tabela 5.2 contém a definição das ACs de Coleções Globais. Considere $C_1 = \$S_1/\delta_1$ uma coleção global de S_1 e $C_2 = \$S_2/\delta_2$ uma coleção global de S_2, tais que C_1 e C_2 estão relacionadas através de uma AC. Se uma instância e_1 de $\$S_1/\delta_1$ é mapeada na instância e_2 de $\$S_2/\delta_2$ ($f(e_1) = e_2$), então e_1 e e_2 são "semanticamente equivalentes" ($e_1 \equiv e_2$), i.é, correspondem ao mesmo objeto no mundo real. As ACs de Elementos (ver Seção 5.3.3) são utilizadas para especificar as funções de mapeamento.

5.3.2 Assertivas de Correspondência de Caminhos

As ACs de Caminhos (ACCs) especificam relacionamentos entre caminhos de tipos semanticamente relacionados. Os tipos T_1 e T_2, dos esquemas das fontes S_1 e S_2, são semanticamente relacionados quando suas instâncias podem representar objetos semanticamente equivalentes. Mais formalmente, T_1 e T_2 são semanticamente relacionados sss (i) Existe uma ACCG

relacionando as coleções globais $\$S_1/\delta_1$ e $\$S_2/\delta_2$ tais que $T\delta_1 = T_1$ e $T\delta_2 = T_2$; ou (ii) Existe uma ACC relacionando os caminhos δ_1 e δ_2 tais que $T\delta_1 = T_1$ e $T\delta_2 = T_2$.

Relação	Notação	Condição de Validação
Subconjunto	$[\$S_1/\delta_1] \subset [\$S_2/\delta_2]$	Existe uma função injetiva *f*: $[\$S_1/\delta_1] \rightarrow [\$S_2/\delta_2]$
Equivalência	$[\$S_1/\delta_1] \equiv [\$S_2/\delta_2]$	Existe uma função bijetiva *f*: $[\$S_1/\delta_1] \rightarrow [\$S_2/\delta_2]$
Disjunção	$[\$S_1/\delta_1] \mid [\$S_2/\delta_2]$	$\neg\exists$ (e_1 em $\$S_1/\delta_1 \wedge e_2$ em $\$S_2/\delta_2 \wedge e_1 \equiv e_2$)
Diferença	$[\$S/\delta] \equiv [\$S_1/\delta_1] - [\$S_2/\delta_2]$	Existe uma função bijetiva *f*: $[\$S/\delta] \rightarrow [\$S_1/\delta_1] - [\$S_2/\delta_2]$
União	$[\$S/\delta] \equiv \bigcup_{i=1}^{n} [\$S_i/\delta_i]$	Existe uma função bijetiva *f*: $[\$S/\delta] \rightarrow \bigcup_{i=1}^{n} [\$S_i/\delta_i]$
Insterseção	$[\$S/\delta] \equiv \bigcap_{i=1}^{n} [\$S_i/\delta_i]$	Existe uma função bijetiva *f*: $[\$S/\delta] \rightarrow \bigcap_{i=1}^{n} [\$S_i/\delta_i]$
Sobreposição	$[\$S_1/\delta_1] \varnothing [\$S_2/\delta_2]$	Existe uma função injetiva parcial *f*: $[\$S_1/\delta_1] \rightarrow [\$S_2/\delta_2]$

Tabela 5.2: ***Assertivas de Correspondência de Coleções Globais***

Neste trabalho, tratamos dois tipos de AC de Caminhos: ACC de Equivalência e ACC de Subconjunto. Estes tipos são definidos a seguir.

- ACC de Equivalência

Sejam δ_1 e δ_2 caminhos (mono-ocorrência ou multi-ocorrência) dos tipos T_1 e T_2, respectivamente, onde T_1 e T_2 são tipos semanticamente relacionados. A ACC $[T_1/\delta_1] \equiv [T_2/\delta_2]$, especifica que para quaisquer instâncias $\$t_1$ e $\$t_2$ de T_1 e T_2, respectivamente, se $\$t_1 \equiv \$t_2$ então $\$t_1/\delta_1 \equiv \$t_2/\delta_2$.

- ACC de Subconjunto

Sejam δ_1 e δ_2 caminhos multi-ocorrência dos tipos T_1 e T_2, respectivamente, onde T_1 e T_2 são tipos semanticamente relacionados. A ACC de Subconjunto $[T_1/\delta_1] \subset [T_2/\delta_2]$, especifica que para quaisquer instâncias $\$t_1$ e $\$t_2$ de T_1 e T_2, respectivamente, se $\$t_1 \equiv \$t_2$ então $\$t_1/\delta_1 \subset \$t_2/\delta_2$.

É possível a existência de outras formas de correspondências entre caminhos multi-ocorrência, entretanto, estas não serão tratadas neste trabalho.

5.3.3 Assertivas de Correspondência de Elementos

As ACs de Elementos (ACEs) especificam sob que condições dois elementos, os quais são instâncias de tipos semanticamente relacionados, representam o mesmo objeto do mundo real, ou seja, são semanticamente equivalentes.

Sejam T_1 e T_2, tipos semanticamente relacionados. Considere δ_{11}, ..., δ_{1n} caminhos de T_1 e δ_{21}, ..., δ_{2n} caminhos de T_2. A ACE $[T_1, \{\delta_{11}, ..., \delta_{1n}\}] \equiv [T_2, \{\delta_{21}, ..., \delta_{2n}\}]$, especifica que para quaisquer instâncias $\$t_1$ e $\$t_2$ de T_1 e T_2, respectivamente, se $\$t_1/\delta_{1i} \equiv \$t_2/\delta_{2i}$, $n \leq i \leq 1$, então $\$t_1 \equiv \$t_2$.

No estudo de caso apresentado na Seção 5.5, mostramos exemplos do uso dessas assertivas.

5.4 XMLS+Matcher

Nesta Seção, apresentamos o método XMLS+Matcher. As características gerais do método proposto são descritas abaixo:

- Mecanismo Binário de Comparação de Esquemas
  Uma das primeiras considerações a serem feitas antes de iniciar o processo de *matching* é definir a estratégia utilizada para comparar os esquemas [BLN86, OV99]. As principais estratégias discutidas na literatura são a binária, na qual dois esquemas são comparados a cada momento, e a n-ária, na qual os esquemas são comparados em apenas uma iteração do processo. Nosso enfoque utiliza o mecanismo binário de comparação de esquemas.

- Formalização das Correspondências Identificadas
  Uma outra consideração importante é formalizar as correspondências identificadas durante o processo de *matching*. No enfoque proposto, as correspondências identificadas entre os componentes dos esquemas são formalmente especificadas através das ACs.

- Top-Down e Recursivo
  De acordo com as características de um esquema XMLS+, tais como, a sua estrutura em árvore e o alto grau de aninhamento dos elementos, o método proposto é *top-down* e recursivo: *top-down*

visto que inicia comparando os elementos contidos nas raízes (i.é, ligados diretamente às raízes) dos esquemas e, em seguida, compara seus outros elementos; e recursivo porque a comparação de dois elementos pode envolver a comparação de outros dois elementos e assim por diante.

- Interativo e Semi-automatizado

  Identificar correspondências é uma atividade desafiante por várias razões, tais como: (i) mesmo esquemas com conceitos idênticos podem apresentar diferenças na terminologia, na estrutura e na extensão; e (ii) além disso, esta atividade é bastante subjetiva - esquemas podem não capturar, completamente, a semântica dos dados que eles descrevem e podem existir várias correspondências plausíveis entre os esquemas. Assim, o nosso enfoque requer uma interação com o Engenheiro do Conhecimento (EgC)[3] para validar as correspondências identificadas durante o processo de *matching*. Por outro lado, mesmo quando o processo não identifica automaticamente uma AC, esta pode ser especificada pelo EgC ao interagir com o processo.

A seguir, descrevemos cada passo do XMLS+Matcher para gerar as correspondências entre dois esquemas XMLS+ S_1 e S_2. Cada passo do XMLS+Matcher é apoiado por um conjunto de heurísticas, apresentadas no Capítulo 6. Estas heurísticas auxiliam o EgC na identificação das assertivas, quando estas não forem de fácil identificação. O processo de geração de correspondência entre S_1 e S_2 consiste dos seguintes passos:

Passo1: Identificação das ACCGs

Neste passo, as coleções globais de S_1 e S_2 são comparadas para serem geradas as ACCGs. Este passo é detalhado na Seção 6.3.1, onde apresentamos uma heurística que apóia esta identificação.

Passo2: Identificação das ACEs e ACCs

Para cada ACCG ψ identificada no Passo1, onde ψ relaciona a coleção em $\$S_1/\delta_1$ com a coleção em $\$S_2/\delta_2$ faça:

Passo2.1: Identificação da ACE

Neste passo, as chaves associadas às coleções globais são comparadas para serem identificadas as ACs dos elementos em $\$S_1/\delta_1$ com os elementos em $\$S_2/\delta_2$. Este passo é detalhado na Seção 6.3.3.

[3] Neste trabalho, utilizamos o termo Engenheiro do Conhecimento (EgC) para referenciar a pessoa responsável pelo *matching* de esquemas.

Passo2.2: Identificação das ACCs

Neste passo, os tipos dos elementos das coleções são comparados, usando o algoritmo da Figura 5.2, de forma a gerar as ACCs. Este passo é detalhado na Seção 6.3.2, onde apresentamos uma heurística que apóia esta identificação.

Compara_tipos (T_1 , T_2)

1. Identifica ACCs relacionando caminhos de T_1 com caminhos de T_2 .
2. Para cada ACC ψ identificada no passo 1, onde ψ relaciona o caminho δ_1 de T_1 com caminho δ_2 de T_2 , e $T\delta_1$ e $T\delta_2$ são tipos complexos ou referência para tipos complexos, faça :
   2.1 Se δ_1 e δ_2 forem caminhos multi-ocorrência
      {Identifica ACs das instâncias $T\delta_1$ com as instâncias $T\delta_2$}
   2.2 Compara_tipos ($T\delta_1$, $T\delta_2$)

Figura 5.2: ***Algoritmo de Comparação de Tipos***

Como mostramos no Capítulo 6, as heurísticas utilizam informações dos esquemas, tais como nome e tipo do elemento, e informações de dicionários semânticos, para identificar possíveis correspondências entre coleções globais e entre caminhos. Além disso, o XMLS+Matcher também é apoiado por regras de inferência. As regras de inferência identificam outras correspondências a partir das correspondências identificadas anteriormente.

5.5 Estudo de Caso

Suponha que queiramos construir um mediador Med que integra informações das fontes Bib_1, apresentada na Figura 4.7, Bib_2, apresentada na Figura 4.14, e Bib_4, apresentada na Figura 5.3. A Figura 5.4 mostra o esquema de Med. A seguir, usaremos o método descrito na Seção 5.4 para gerarmos as correspondências de Med com Bib_1, Bib_2 e Bib_4.

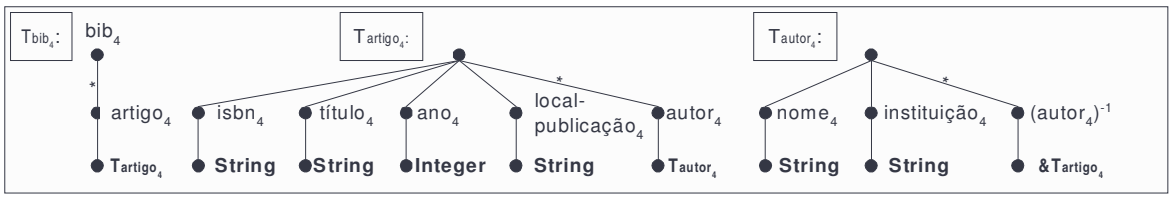


Figura 5.3: ***Esquema XMLS+ Bib4***

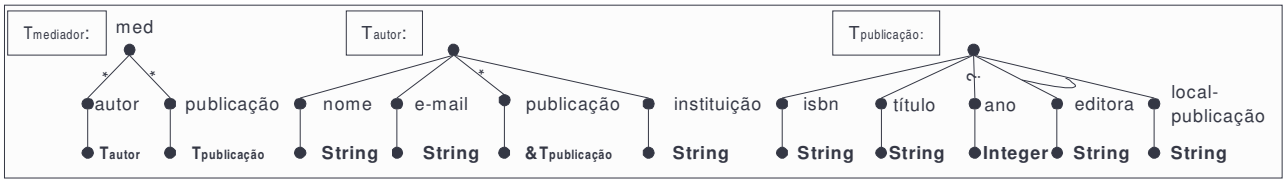


Figura 5.4: ***Esquema XMLS+ de Med***

As ACs de Med com cada fonte local são armazenadas em uma estrutura de árvore. A árvore de assertivas da Figura 5.5 contém as ACs geradas do *matching* de Med com Bib_1. O esquema XMLS+ da árvore do mediador e os documentos XML gerados são apresentados no Apêndice A.

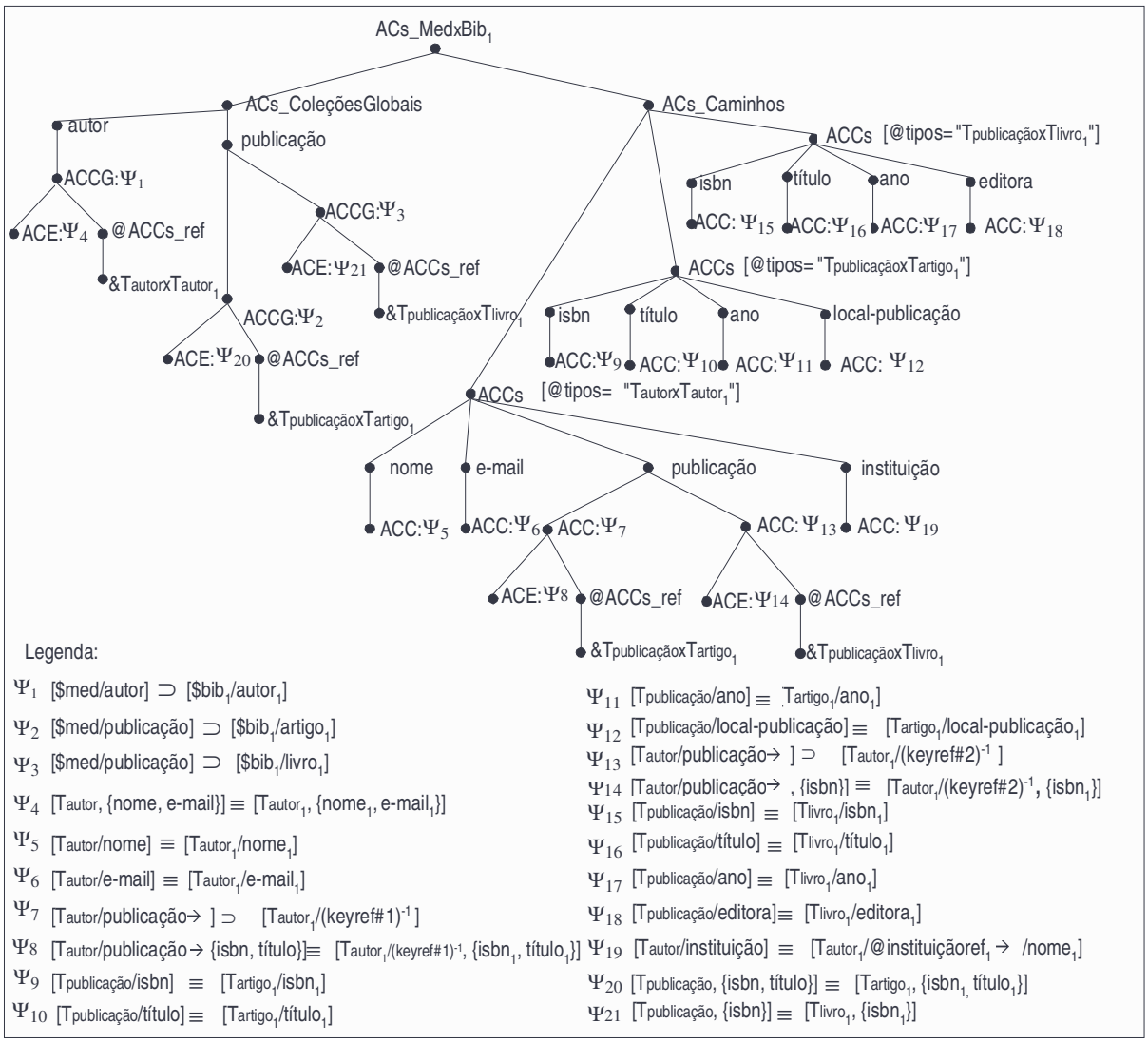


Figura 5.5: ***ACs de Med e Bib1***

O elemento raiz ($ACs\_MedxBib_1$) contém um elemento ACs_ColeçõesGlobais, o qual contém as ACCGs, e um elemento ACs_Caminhos, o qual contém as ACCs. A seguir, discutimos como a árvore de assertivas é gerada durante a comparação de Med e Bib_1 em cada passo do método proposto.

Passo1: Comparando as coleções globais de Med e Bib_1 identificamos 3 ACCGs: Ψ_1: [$med/autor] ⊃ [$bib$_1$/autor$_1$], Ψ_2: [$med/publicação] ⊃ [$bib$_1$/artigo$_1$] e Ψ_3: [$med/publicação] ⊃ [$bib$_1$/livro$_1$]. Dessa forma, na árvore da Figura 5.5, o elemento ACs_ColeçõesGlobais contém 2 subelementos, um para cada coleção global de Med. Como a coleção

global $med/publicação está relacionada com duas coleções globais de Bib_1 ($bib_1/artigo_1$ e $bib_1/livro_1$), então o elemento publicação possui dois subelementos ACCG. A ACCG de subconjunto Ψ_1 especifica que para qualquer elemento e_1 em $med/autor, existe um elemento e_2 em $bib_1/autor_1$, tal que $e_1 \equiv e_2$ (e_1 e e_2 são semanticamente equivalente, i.é, representam o mesmo objeto do mundo real). Ψ_2 e Ψ_3 são similares à Ψ_1.

Passo2: De acordo com o nosso método, para cada ACCG deve-se especificar:

Passo2.1: A ACE que indica as condições de equivalência dos elementos das coleções. Na árvore da Figura 5.5, a ACCG Ψ_1 tem associado um elemento ACE que contém a assertiva Ψ_4:[Tautor, {nome, e-mail}] ≡ [$Tautor_1$, {$nome_1$, $e\text{-}mail_1$}], a qual especifica que para quaisquer instâncias t_1 e t_2 de Tautor e $Tautor_1$, respectivamente, se $t_1/nome \equiv t_2/nome_1$ e $t_1/e\text{-}mail \equiv t_2/e\text{-}mail_1$, então $t_1 \equiv t_2$.

Passo2.2: Para cada ACCG deve-se especificar também, todas ACCs que indicam as correspondências entre os caminhos dos tipos dos elementos das coleções. Na árvore da Figura 5.5, a ACCG Ψ_1 tem associado um atributo ACCs_ref o qual contém uma referência para o elemento que contém as ACs de Caminhos de Tautor com $Tautor_1$ (ACCs[@tipos="TautorxTautor$_1$"). Estas Assertivas são obtidas através da comparação dos tipos Tautor e $Tautor_1$ de acordo com o algoritmo da Figura 5.2.

Comparando os tipos TautorxTautor$_1$ são identificadas 5 ACC:

Ψ_5:[Tautor/nome] ≡ [$Tautor_1/nome_1$],

Ψ_6:[Tautor/e-mail]≡[$Tautor_1/e\text{-}mail_1$],

Ψ_7:[Tautor/publicação →]⊃[$Tautor_1/(keyref\#1)^{-1}$],

Ψ_{13}:[Tautor/publicação →] ⊃ [$Tautor_1/(keyref\#2)^{-1}$] ,

Ψ_{19}:[Tautor/instituição] ≡ [$Tautor_1$/@instituiçãoref$_1$ → /$nome_1$] .

Estas ACCs são representadas na árvore como mostradas na Figura 5.5.

A assertiva Ψ_5 especifica que para quaisquer t_1, t_2 instâncias de Tautor e $Tautor_1$, respectivamente, se $t_1 \equiv t_2$ então $t_1/nome \equiv t_1/nome_1$. Ψ_6 é similar a Ψ_5. A assertiva Ψ_7, especifica que para quaisquer instâncias t_1 e t_2 de Tautor e $Tautor_1$, respectivamente, se $t_1 \equiv t_2$, então

t_1/publicação → ⊃ t_2/(keyref#1)$^{-1}$ (i.é, para qualquer elemento e_1 em t_1/publicação →, existe um elemento e_2 em t_2/(keyref#1)$^{-1}$, tal que $e_1 \equiv e_2$). Ψ_{13} é similar à Ψ_7. A assertiva Ψ_{19} especifica que para quaisquer instâncias t_1 e t_2 de T_{autor} e T_{autor_1}, respectivamente, se $t_1 \equiv t_2$, então t_1/instituição ≡ t_2/@instituiçãoref$_1$→ /nome$_1$.

No caso da ACC Ψ_7, como $T_{publicação}$ e T_{artigo_1} são tipos complexos, deve-se identificar as ACCs de $T_{publicação}$ e T_{artigo_1}. Além disso, como os caminhos são multi-ocorrência, deve-se especificar a ACE correspondente. Dessa forma, como mostrado na árvore da Figura 5.5, a ACC Ψ_7 tem associado a ACE Ψ_8 e um atributo @ACCs_ref o qual contém uma referência para o elemento que contém as ACCs de $T_{publicação}$ com T_{artigo_1} (ACCs[@tipos="$T_{publicação}$xT_{artigo_1}"). Estas assertivas são obtidas através da comparação dos tipos $T_{publicação}$ e T_{artigo_1} de acordo com o algoritmo da Figura 5.2.

Devemos proceder de forma similar com relação à Ψ_{13} que tem associado a ACE Ψ_{14} e um atributo @ACCs_ref, o qual contém uma referência para o elemento que contém as ACCs de $T_{publicação}$ com T_{livro_1} (ACCs[@tipos ="$T_{publicação}$xT_{livro_1}"). Deve-se repetir os passos 2.1 e 2.2 para as ACsCGs Ψ_2 e Ψ_3.

A Figura 5.6 mostra a árvore de ACs resultante da comparação de Med e Bib_2 e a Figura 5.7 mostra a árvore de ACs resultante da comparação de Med e Bib_4. É importante observar que apesar dos esquemas de Med, Bib_1, Bib_2 e Bib_4 terem estruturas diferentes, as correspondências entre eles podem ser precisamente especificadas através das ACs. A seguir, mostramos o uso das ACs do mediador para determinar como as consultas definidas na visão global (esquema do mediador) serão respondidas a partir de consultas nas fontes locais.

5.6 Usando as ACs do Mediador na Integração de Informações

Como discutido anteriormente, as correspondências do esquema do mediador com os esquemas das fontes locais desempenham um papel central tanto nos sistemas de integração de informações que usam visão virtual, quanto naqueles que usam visão materializada. No caso dos sistemas que usam visão virtual, o componente de reformulação de consultas [AKMP01, BBM01, IFF+99] utiliza as ACs do mediador para selecionar um conjunto de fontes que podem ser usadas para responder uma consulta global. Além do reformulador de consultas, os sistemas de integração têm um componente

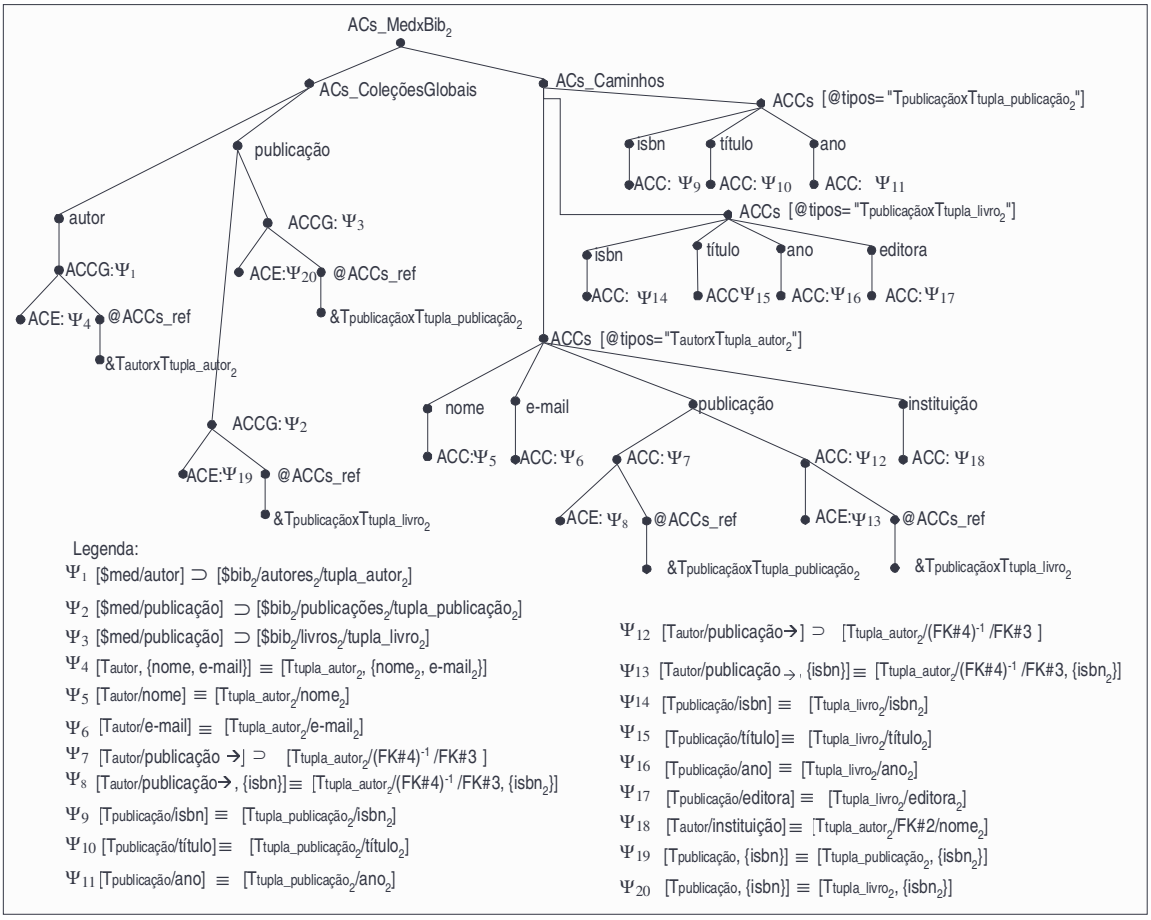


Figura 5.6: ***ACs de Med e Bib₂***

de otimização de consulta o qual gera um plano eficiente de execução da consulta. Neste trabalho só discutimos brevemente a questão do uso das ACs na seleção das fontes locais e geração das subconsultas para estas.

Quando um usuário submete uma consulta ao esquema do mediador, o sistema de integração deve traduzir a consulta em uma série de subconsultas que serão enviadas para as fontes locais. Considere, por exemplo, a consulta submetida ao mediador Med: "Obtenha todos os títulos das publicações de autoria de "Ângelo Brayner" a partir do ano de 1990", correspondente a seguinte consulta em XPath:

Q = $med/autor[nome = "Ângelo Brayner"]/publicação[ano > 1990]/título.

As ACs de Med nos permite identificar precisamente os caminhos nas fontes de informação locais que contém informações que podem ser usadas para responder a consulta **Q**. A partir da ACs de Med com Bib_1 (Figura 5.5) é gerada a expressão de caminho $\mathbf{Q}_1$ na Figura 5.8. Para cada passo desta expressão é indicada a AC que foi utilizada para geração deste passo. A partir da ACs de Med

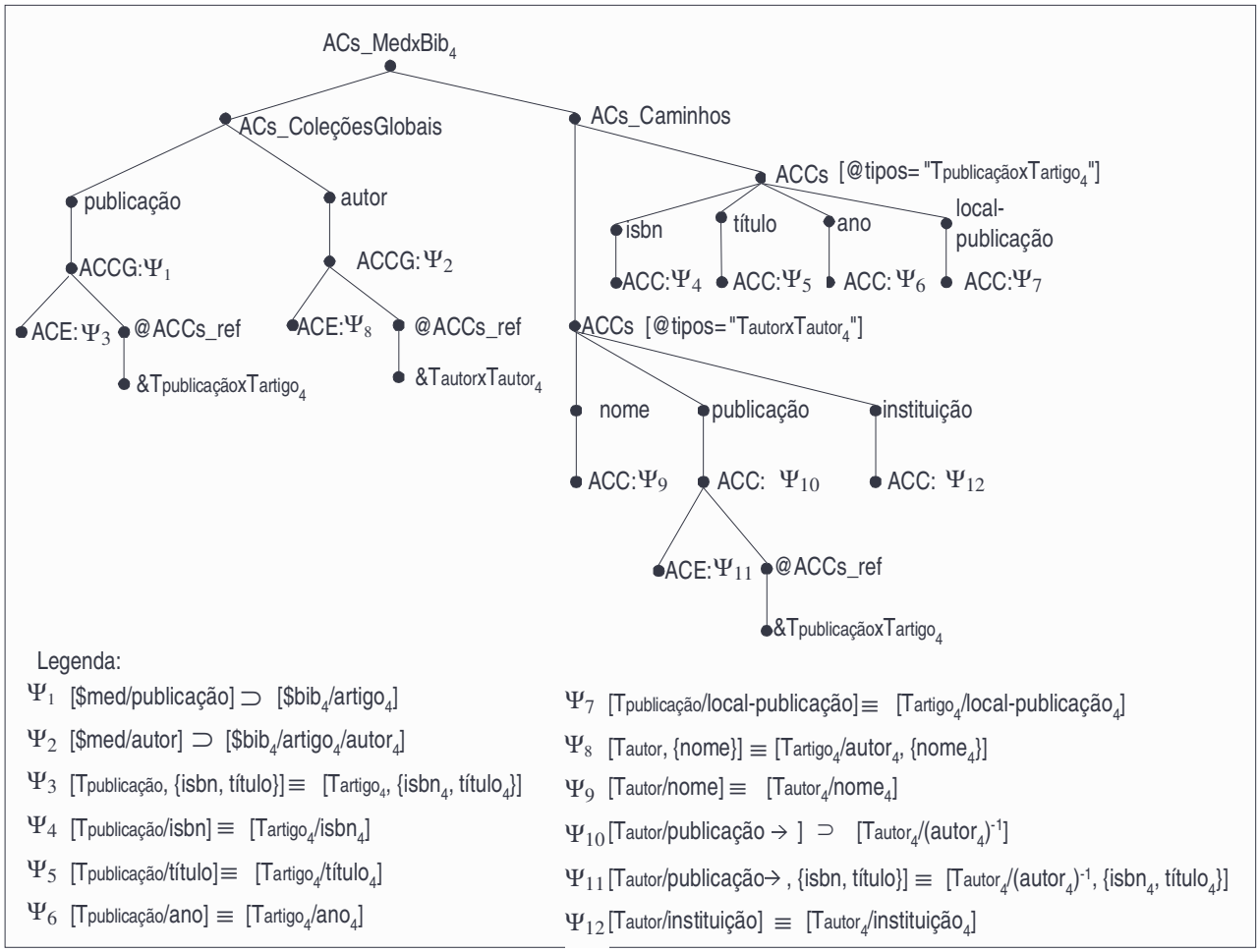


Figura 5.7: ***ACs de Med e Bib4***

com Bib2 (Figura 5.6) é gerada a expressão de caminho $\mathbf{Q}_2$ e a partir da ACs de Med com Bib4 (Figura 5.7) é gerada a expressão de caminho $\mathbf{Q}_4$ na Figura 5.8.

É importante notar que as expressões de caminho $\mathbf{Q}_1$, $\mathbf{Q}_2$ e $\mathbf{Q}_4$ não são expressões de caminho válidas de XPath. Usamos a extensão de XPath proposta no Capítulo 4 que permite navegar através de um atributo de referência, de uma inversa de um atributo de referência, de uma keyref e de uma inversa de uma keyref. Portanto, $\mathbf{Q}_1$, $\mathbf{Q}_2$ e $\mathbf{Q}_4$ devem ser traduzidas em consultas XQuery [BCFF+02] correspondentes. Em [Abr02] está sendo desenvolvido um algoritmo que trata desse problema.

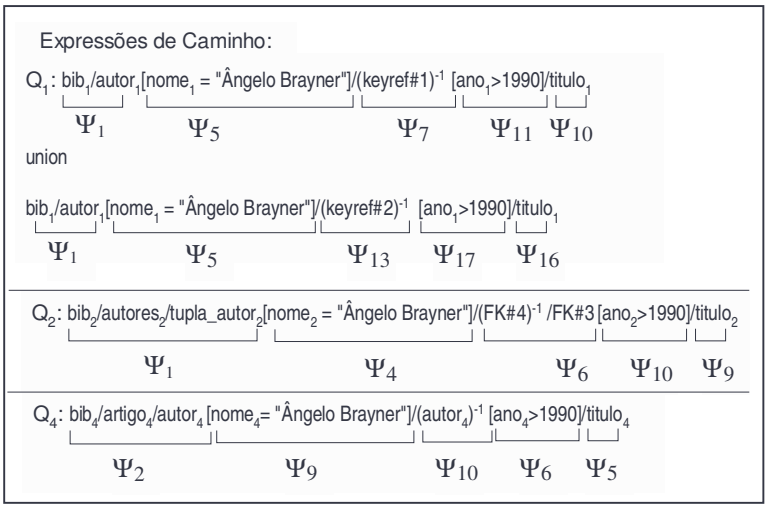


Figura 5.8: ***Sub-consultas Q1, Q2 e Q4 referentes à consulta global Q***

CAPÍTULO 6

O USO DE HEURÍSTICAS NO *MATCHING* DE ESQUEMAS XMLS+

Cada passo do XMLS+Matcher é apoiado por um conjunto de heurísticas e regras de inferência que auxiliam o Engenheiro do Conhecimento na identificação das assertivas. Neste Capítulo, apresentamos essas heurísticas e regras. Na Seção 6.1, usamos uma taxonomia das técnicas para identificação de correspondências para comentar alguns trabalhos relacionados ao nosso enfoque. Na Seção 6.2, descrevemos o uso de dicionários semânticos para apoiar na identificação de afinidade terminológica entre os elementos dos esquemas. Na Seção 6.3, detalhamos cada um dos passos do XMLS+Matcher, discutido no Capítulo 5 e mostramos como os passos utilizam as heurísticas e as regras de inferência.

6.1 Uma Taxonomia das Técnicas para Identificação de Correspondências

A fim de comparar os enfoques propostos na literatura para identificação de correspondências, usamos uma taxonomia das técnicas para *matching* de esquemas como base para esta comparação. Os itens desta taxonomia são baseados nas taxonomias apresentadas em [MCH02, RB01]. Para cada item, analisamos o suporte fornecido pelo XMLS+Matcher. As técnicas para identificação de correspondências podem ser classificadas pelos seguintes critérios:

- Enfoque Semi-automatizado

  Abordagens que identificam correspondências semi-automaticamente utilizam heurísticas e regras de inferência para apoiar a identificação.

Como mostra a Figura 6.1, XMLS+Matcher é apoiado por uma base de conhecimentos que contém um conjunto de heurísticas e um conjunto de regras de inferência. As heurísticas utilizam as informações dos esquemas comparados para determinar quando dois elementos possuem probabilidade de estarem relacionados através de uma Assertiva de Correspondência (AC). As regras de inferência inferem outras correspondências com base nas correspondências identificadas anteriormente. Assim, essas regras servem para identificar novas correspondências, assim como para validar as correspondências já identificadas.

- Baseada em Esquema

  Abordagens baseadas em esquema consideram as informações dos esquemas a serem integrados para identificar correspondências. Estas abordagens podem ser:

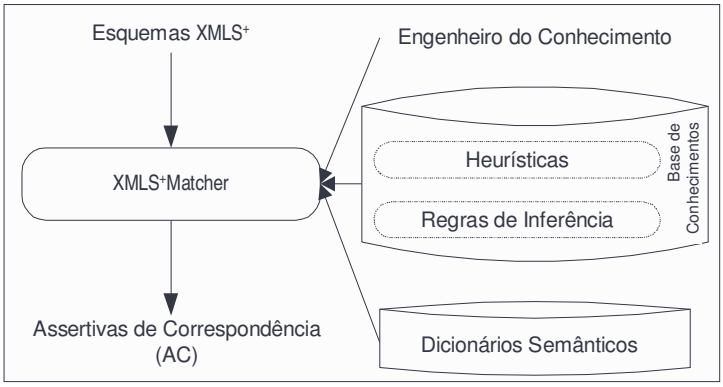


Figura 6.1: ***Enfoque Semi-automatizado do XMLS+Matcher***

- Baseada em Lingüística

  Abordagens baseadas em lingüística identificam correspondências a partir dos nomes e de outras descrições textuais dos elementos dos esquemas. Para isso, algumas abordagens utilizam dicionários semânticos sofisticados, como *thesauri* e ontologias [BBVC98, Mir97, RR95], que contêm relacionamentos semânticos entre termos.

Nossas heurísticas utilizam os nomes dos elementos para auxiliar na identificação da afinidade terminológica e, assim, identificar elementos com probabilidade de estarem relacionados. Esta afinidade ocorre quando dois termos diferentes são usados para descrever um mesmo conceito ou conceitos relacionados. Dicionários semânticos são utilizados para verificar a afinidade terminológica (vide Figura 6.1), como veremos na Seção 6.2.

- Baseada em Restrição

  Abordagens baseadas em restrição identificam correspondências a partir das restrições associadas aos componentes dos esquemas.

  Nossas heurísticas usam informações sobre o tipo e as restrições de ocorrência, de chave e referenciais dos elementos para auxiliar na identificação de elementos com probabilidade de estarem relacionados.

- Baseada em Instância

  Abordagens baseadas em instância identificam correspondências a partir das informações das instâncias persistentes nas fontes de informação. Estas abordagens geralmente utilizam técnicas de aprendizagem de máquina [DDH01] e *data mining* [SHKC93].

  Em XMLS+Matcher, após serem identificados dois elementos com probabilidade de estarem relacionados, o Engenheiro do Conhecimento (EgC) é consultado para informar sobre a afinidade extensional entre os mesmos, ou seja, se os elementos possuem instâncias equivalentes ou relacionadas (vide Figura 6.1). Com base nesta informação, é gerada a correspondência precisa entre os elementos.

  A seguir, comentamos alguns trabalhos relacionados.

ARTEMIS é a ferramenta para identificação de correspondências do MOMIS [CA97]. Para identificar correspondências, os esquemas locais são mapeados para um modelo orientado a objeto. Esta ferramenta identifica correspondências entre classes e atributos baseada na afinidade terminológica, estrutural e extensional. Para auxiliar na identificação de classes com afinidade terminológica, ARTEMIS consulta um *thesaurus*. As classes dos esquemas locais que possuem afinidade são agrupadas para obter as classes globais do esquema global. ARTEMIS especifica apenas as correspondências de Equivalência, Subconjunto e Disjunção entre as classes, e assume que quando duas classes não possuem uma correspondência, estas são sobrepostas. O nosso formalismo permite especificar mais quatro tipos de correspondências (Diferença, União, Interseção e Sobreposição).

Os enfoques adotados por Cupid [MBR01] e Xyleme [DRRS+02] identificam correspondências entre elementos dos esquemas baseado na afinidade terminológica e estrutural.

Estes enfoques têm como entrada esquemas representados em estrutura de árvore. Para auxiliar na identificação da afinidade terminológica, ambos consultam um *thesaurus*. Uma desvantagem destes enfoques é que apenas identificam a existência de uma correspondência entre dois elementos, não sendo precisos o suficiente para especificar o tipo da correspondência identificada, além de não permitirem especificar correspondências entre esquemas com estruturas diferentes. Especificar uma correspondência precisa é muito importante, visto que esta será utilizada no momento da reformulação de consultas das visões virtuais ou da manutenção das visões materializadas. Como discutido anteriormente, o nosso formalismo permite definir de forma precisa as correspondências entre os esquemas.

Dos três enfoques comentados acima, o que mais se assemelha à nossa abordagem é o ARTEMIS. Isto porque, apesar do ARTEMIS utilizar um modelo orientado a objetos, é o único que identifica correspondências baseadas, também, na afinidade extensional, podendo, desta forma, especificar correspondências precisas entre os esquemas.

6.2 Usando Dicionários Semânticos para Apoiar a Identificação de Afinidade Terminológica

Para auxiliar na identificação de afinidade terminológica, e, assim, identificar elementos com probabilidade de estarem relacionados entre os esquemas, utilizamos a semântica fornecida pelos nomes dos elementos (também chamados de *termos*). Os termos são associados aos elementos pelos projetistas[4] das fontes de informação. É tarefa do projetista associar termos que sejam expressivos de forma que permitam uma interpretação correta do domínio do sistema de integração. O uso de termos não-expressivos pode contribuir para uma interpretação incorreta. Mesmo assim, utilizar a semântica fornecida pelos termos dos elementos é uma oportunidade que deve ser explorada na identificação de correspondências entre os mesmos.

Neste trabalho, usamos um dicionário semântico independente de domínio, chamado *Dicionário Semântico Externo* (DSE)[5] para apoiar a identificação da afinidade terminológica. O DSE deve conter um conjunto de relacionamentos semânticos entre termos. Este tipo de

[4] O termo *projetista* é utilizado nesta dissertação para referenciar a pessoa responsável em projetar uma fonte de informação.
[5] Um exemplo de DSE é o WordNet [Mil95], desenvolvido pela Universidade de Princeton e disponível na Internet.

relacionamento é definido através do significado dos termos. Os relacionamentos semânticos que são fornecidos pelo DSE e que são de interesse neste trabalho são:

- **Sinônimo**: definido entre dois termos t_i e t_j, com $t_i \neq t_j$, tal que t_i tem o mesmo significado que t_j (lê-se t_i é sinônimo de t_j).
  Por exemplo: o termo *Empregado* é sinônimo do termo *Funcionário*.
- **Hiperônimo**: definido entre dois termos t_i e t_j, tal que t_i tem um significado mais amplo que t_j (lê-se t_i é hiperônimo de t_j). Neste caso, podemos dizer que t_j tem um significado mais restrito que t_i. Assim t_j é **hipônimo** de t_i.
  Por exemplo: o termo *Médico* é um hiperônimo do termo *Endocrinologista*, conseqüentemente, o termo *Endocrinologista* é um hipônimo do termo *Médico*.
- **Termos Relacionados Através de um Hiperônimo Comum**: definido entre dois termos t_j e t_k, tais que ambos são termos com significados mais restritos de um mesmo hiperônimo (lê-se t_j está relacionado à t_k através de um hiperônimo comum).
  Por exemplo: o termo *Endocrinologista* é um hipônimo do termo *Médico* e o termo *Cardiologista* também é um hipônimo do termo *Médico*, conseqüentemente o termo *Endocrinologista* está relacionado ao termo *Cardiologista* através do termo hiperônimo *Médico*.

Além do DSE, usamos, neste trabalho, um dicionário semântico específico do domínio da aplicação, chamado *Dicionário Semântico Interno* (DSI). O DSI é similar ao que alguns trabalhos chamam de ontologia do domínio da aplicação [KMAA+01]. O DSI é construído de forma incremental, à medida que são identificadas as correspondências entre elementos dos esquemas durante o processo de *matching*. Ao ser identificada uma correspondência entre dois elementos, os termos destes elementos são armazenados no DSI. Caso já exista um relacionamento semântico entre estes termos no DSE, o relacionamento semântico é incluído juntamente com os termos, caso contrário, os termos são incluídos no DSI como termos correlatos. Assim, o DSI contém tanto os termos relacionados no DSE como os termos correlatos.

A Figura 6.2 mostra o pseudo-código da função Verifica_Afinidade_Terminológica que é usada pelo XMLS+Matcher para verificar a existência de afinidade terminológica entre termos. Esta função recebe como entrada dois termos e retorna "verdadeiro", se os termos tiverem afinidade terminológica ou "falso", caso contrário. Para isso, esta função usa o DSI e o DSE. Antes de iniciar a comparação dos termos, é realizada a normalização de termos. A normalização é necessária já que elementos similares em diferentes esquemas podem ter termos que diferem em decorrência do uso

de abreviações, preposições, etc. Assim, a normalização inclui: (i) expansão, que substitui uma abreviação pelo termo por extenso correspondente; e (ii) eliminação, que descarta preposições, artigos, números além dos termos colocados durante a etapa de mapeamento de esquemas, como por exemplo, o termo tupla, no caso do mapeamento de um esquema relacional (vide Capítulo 4).

```
Verifica_Afinidade_Terminológica (Termo1, Termo2)
Normaliza_Termos (Termo1, Termo2)
Se Termo1 e Termo2 forem iguais
        Retorna(Verdadeiro)
Senão
    Se Termo1 e Termo2 estão relacionados no DSI
        Retorna(Verdadeiro)
    Senão
        Se Termo1 e Termo2 estão relacionados no DSE
            Retorna(Verdadeiro)
        Senão
            Retorna(Falso)
```

Figura 6.2: ***Pseudo-Código da Função Verifica_Afinidade_Terminológica***

6.3 Identificando Correspondências entre Esquemas XMLS+

Nesta Seção, utilizamos *workflows* [EL95] para detalhar os passos do XMLS+Matcher. Um *workflow* consiste de um conjunto de atividades que são executadas em uma determinada sequência. Isto significa que a atividade A_i não pode começar até que a atividade A_{i-1} tenha terminado. Além das atividades, um *workflow* consiste de um conjunto de condições, as quais restringem a execução de determinadas atividades.

Como mostrado no *Workflow* W_1 na Figura 6.3(a), o XMLS+Matcher consiste de duas atividades principais. A atividade A_1 trata da identificação das ACs de Coleções Globais. Esta atividade é detalhada na Seção 6.3.1. A atividade A_2 trata da identificação das ACs de Elementos e Caminhos entre os tipos das coleções Globais.

A identificação das ACs de Elementos e Caminhos consiste de três atividades, que estão descritas no *Workflow* W_2, mostrado na Figura 6.3(b). A atividade $A_{2.1}$ trata da identificação das ACs de Elementos, a qual é detalhada na Seção 6.3.3. A atividade $A_{2.2}$ trata da identificação das ACs de Caminhos que é detalhada na Seção 6.3.2. Ao ser identificada uma AC de Caminhos, esta é encaminhada para a atividade $A_{2.3}$ que verifica as seguintes condições: (i) se os caminhos forem multi-ocorrência e os tipos dos caminhos forem complexos (condição $C_{1.1}$), é requisitada a atividade

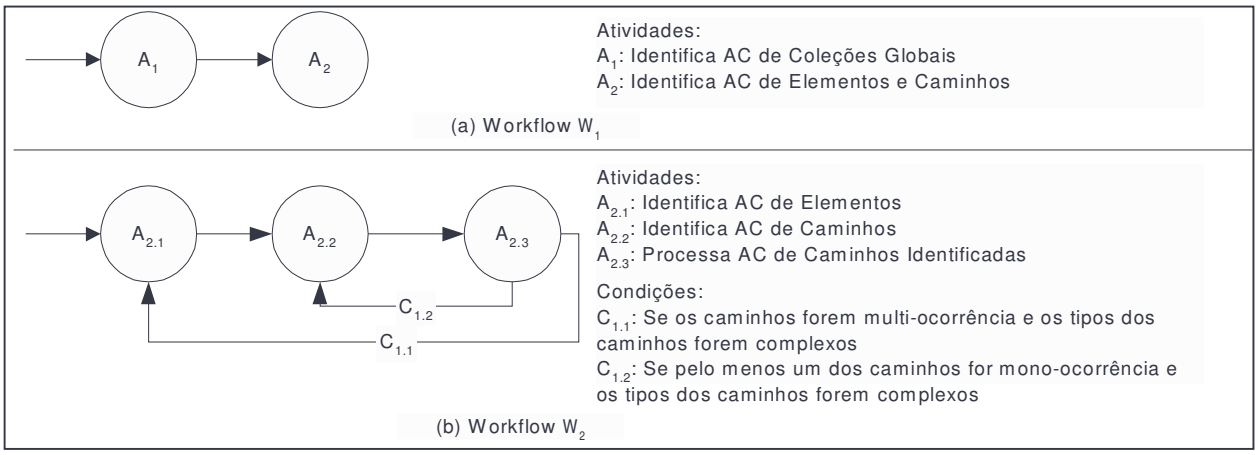


Figura 6.3: ***Workflows W1 e W2***

$A_{2.1}$; (ii) se pelo menos um caminho for mono-ocorrência e os tipos dos caminhos forem complexos (condição $C_{1.2}$), é requisitada a atividade $A_{2.2}$.

A seguir detalhamos as atividades A_1, $A_{2.1}$ e $A_{2.2}$. Para facilitar a identificação de correspondências entre caminhos usamos duas tabelas (Tabela 1 e Tabela 2), as quais são geradas durante o processo de *matching*. A Tabela 1 contém todos os pares de tipos que já foram identificados como sendo semanticamente relacionados (vide definição na Seção 5.3.2 do Capítulo 5). A Tabela 2 contém todos os pares de tipos que já foram identificados como não sendo semanticamente relacionados.

6.3.1 Identificando ACs de Coleções Globais

Os métodos propostos para *matching* e integração de esquemas orientados a objetos [BBVC98, RR95], utilizam uma heurística baseada na afinidade terminológica e estrutural para identificar correspondências entre as classes (extensões de classes). Como discutido no Capítulo 4, as extensões das classes correspondem às coleções globais nos esquema XMLS+. Dessa forma, propomos uma heurística similar para determinar quando duas coleções globais de esquemas XMLS+ têm afinidade semântica, i.é, possuem probabilidade de estarem relacionadas.

No restante desta Seção, considere $Traiz\_S_1$ tipo raiz do esquema S_1 e $Traiz\_S_2$ tipo raiz do esquema S_2. Sejam $\delta_1 = \ell_{11}/\ell_{12}/\ldots/\ell_{1n}$, e $\delta_2 = \ell_{21}/\ell_{22}/\ldots/\ell_{2m}$ caminhos multi-ocorrência de S_1 e S_2. Suponha $C_1=\$S_1/\delta_1$ e $C_2=\$S_2/\delta_2$ coleções globais de S_1 e S_2.

Antes de apresentarmos a heurística, definimos as condições necessárias para que as coleções globais C_1 e C_2 tenham afinidade terminológica e afinidade estrutural.

Definição 6.1: *As coleções globais C_1 e C_2 têm afinidade terminológica se os termos correspondentes às ligações l_{1n} e l_{2m} forem semanticamente relacionados.*

Definição 6.2: *As coleções globais C_1 e C_2 têm afinidade estrutural se os tipos $T\delta_1$ e $T\delta_2$ tiverem estruturas compatíveis (se $T\delta_1$ e $T\delta_2$ tiverem elementos com nome e tipo similares).*

Heurística 1: *As coleções globais C_1 e C_2 têm afinidade semântica se as mesmas tiverem afinidade terminológica ou estrutural.*

O *Workflow* W_3, apresentado na Figura 6.4, detalha a atividade A_1 que trata da identificação de correspondências entre as coleções globais dos esquemas S_1 e S_2. No caso em que um dos esquemas é um esquema de mediador, então deve-se considerar S_1 o esquema mediador. Assim as assertivas geradas constituem as Assertivas de Correspondência do Mediador, as quais serão usadas na reformulação de consultas. Um exemplo desta situação é apresentado no final desta Seção.

A seguir descrevemos cada atividade do *Workflow* W_3.

- $A_{1.1}$: *Seleciona Caminhos Multi-ocorrência de $Traiz\_S_1$*

  Esta atividade seleciona caminhos multi-ocorrência de $Traiz\_S_1$. Para selecionar um caminho, as árvores dos tipos devem ser percorridas a partir da árvore do tipo $Traiz\_S_1$ de acordo com um algoritmo *breadth-first* e *top-down*.

  Para cada caminho $\delta_1 = l_{11}/l_{12}/.../l_{1n}$ selecionado (condição $C_{1.1}$), é requisitada a atividade $A_{1.2}$. Após todos os caminhos multi-ocorrência de $Traiz\_S_1$ terem sido selecionados e comparados (condição $C_{1.2}$), é requisitada a atividade $A_{1.7}$.

- $A_{1.2}$: *Seleciona Caminhos Multi-ocorrência de $Traiz\_S_2$*

  Esta atividade seleciona caminhos multi-ocorrência de $Traiz\_S_2$. Para selecionar um caminho, as árvores dos tipos também devem ser percorridas, como descrito na atividade $A_{1.1}$. Para cada caminho $\delta_2 = l_{21}/l_{22}/.../l_{2m}$ selecionado (condição $C_{1.3}$), é requisitada a atividade $A_{1.3}$.

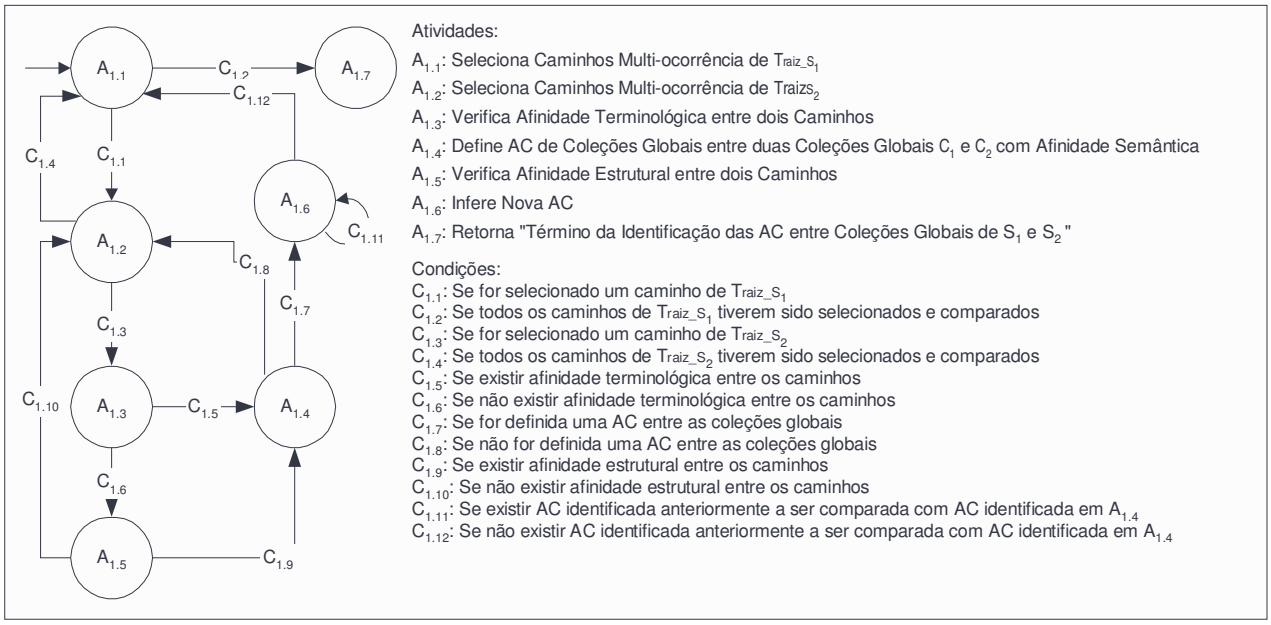


Figura 6.4: ***Workflow W3***

Após todos os caminhos multi-ocorrência de $Traiz\_S_2$ terem sido selecionados e comparados (condição $C_{1.4}$), é requisitada a atividade $A_{1.1}$.

- $A_{1.3}$: *Verifica Afinidade Terminológica entre dois Caminhos*

Esta atividade verifica se existe afinidade terminológica entre os caminhos δ_1 e δ_2. Para isso, deve-se verificar se existe uma afinidade terminológica entre termos associados às ligações l_{1n} de δ_1 e l_{2m} de δ_2 (usa a função Verifica_Afinidade_Terminológica, mostrada na Figura 6.2). Se existir (condição $C_{1.5}$), é requisitada a atividade $A_{1.4}$, senão (condição $C_{1.6}$), é requisitada a atividade $A_{1.5}$.

- $A_{1.4}$: *Define AC de Coleções Globais entre duas Coleções Globais C_1 e C_2 com Afinidade Semântica*

Dado que as coleções globais $C_1 = \$S_1/\delta_1$ e $C_2 = \$S_2/\delta_2$ têm afinidade semântica, conforme Heurística 1, o EgC deve informar qual das ACs se aplica às coleções:

a) C_1 e C_2 são equivalentes
b) C_1 está contido em C_2
c) C_1 contém C_2
d) C_1 e C_2 são sobrepostas
e) C_1 e C_2 são disjuntas, porém relacionadas
f) Não existe AC

Se possivel, o EgC poderá consultar as instâncias das coleções C_1 e C_2 nas suas fontes locais para validar a correspondência.

Se for identificada uma AC entre C_1 e C_2, (condição $C_{1.7}$) é requisitada a atividade $A_{1.6}$. Além disso:

- se os termos das ligações l_{1n} e l_{2m} não estiverem contidos no DSI, estes termos, juntamente com o relacionamento semântico entre os mesmos, são armazenados no DSI, e os tipos $T\delta_1$ e $T\delta_2$ não precisarão mais serem percorridos pelas atividades $A_{1.1}$ e $A_{1.2}$, respectivamente, otimizando a identificação das ACs de Coleções Globais.
- os tipos $T\delta_1$ e $T\delta_2$ são incluídos na Tabela 1 (contém os pares de tipos semanticamente relacionados).

Se não for identificada uma AC entre C_1 e C_2 (condição $C_{1.8}$), é requisitada a atividade $A_{1.2}$. Além disso, os tipos $T\delta_1$ e $T\delta_2$ são incluídos na Tabela 2 (contém os pares de tipos que não são semanticamente relacionados).

- $A_{1.5}$: *Verifica Afinidade Estrutural entre dois Caminhos*

Esta atividade verifica se os caminhos δ_1 e δ_2 têm afinidade estrutural. Para isso, seus tipos, $T\delta_1$ e $T\delta_2$, são comparados para verificar se possuem elementos em comum (com nomes e tipos similares). Se existir afinidade estrutural entre os caminhos δ_1 e δ_2 (condição $C_{1.9}$), é requisitada a atividade $A_{1.4}$. Senão (condição $C_{1.10}$), é requisitada a atividade $A_{1.2}$.

- $A_{1.6}$: *Infere Nova AC*

As regras de inferência, apresentadas na Seção 6.3.4, são aplicadas para inferir nova AC a partir da AC identificada na atividade $A_{1.4}$ e de uma AC identificada anteriormente. Após ser inferida, a AC de Coleções Globais é apresentada ao EgC para que seja validada.

Esta atividade é realizada enquanto existir AC que tenha sido identificada anteriormente e que não tenha sido comparada com a AC identificada na atividade $A_{1.4}$ (condição $C_{1.11}$). Se não existirem mais assertivas a serem comparadas (condição $C_{1.12}$), é requisitada a atividade $A_{1.1}$.

As coleções globais que tiverem AC inferidas e validadas pelo EgC não serão mais selecionadas para serem comparadas com outras coleções globais. Consequentemente, os tipos destas coleções não precisarão mais ser percorridos. Desta forma, o processo para identificação de AC de Coleções Globais é otimizado. Além disso, os termos dos tipos destas coleções serão incluídos na Tabela 1.

- $A_{1.7}$: *Retorna "Término da Identificação das ACs entre Coleções Globais de S_1 e S_2"*

Esta atividade indica a finalização da identificação das ACs entre coleções globais de S_1 e S_2. Em seguida, o EgC poderá gerar as correspondências entre as coleções globais que não foram identificadas semi-automaticamente e, assim, finalizar a identificação das ACs de Coleções Globais. Os tipos das coleções globais que tiverem AC geradas pelo EgC são incluídos na Tabela 1.

A seguir, exemplificamos a identificação das ACs de Coleções Globais.

Exemplo 6.1: Consideremos os esquemas Med e Bib_1, apresentados nas Figuras 5.4 e 4.7, respectivamente. Inicialmente, a atividade $A_{1.1}$ seleciona o caminho $med/autor de $T_{mediador}$ e a atividade $A_{1.2}$ seleciona o caminho $med/autor$_1$ de T_{bib_1}. A atividade $A_{1.3}$ verifica que existe afinidade terminológica (e consequentemente semântica) entre os caminhos bib_1$/autor$_1$ e $med/autor. Assim, o EgC deve informar qual das ACs se aplica às coleções $med/autor e bib_1$/autor$_1$. O EgC verifica que a coleção $med/autor contém a coleção bib_1$/autor$_1$. Desta forma, é gerada a seguinte AC de Subconjunto:

Ψ: [$med/autor] $\supset$ [bib_1$/autor$_1$].

Exemplo 6.2: Consideremos os esquemas Med e Bib_2 (apresentado na Figura 4.14). A atividade $A_{1.1}$ seleciona o caminho $med/publicação de $T_{mediador}$ e a atividade $A_{1.2}$ seleciona um caminho de T_{bib_2}. Como todos os elementos contidos no tipo T_{bib_2} são mono-ocorrência, as árvores dos tipos destes elementos são percorridas a procura de caminhos multi-ocorrência. Analisando o tipo $T_{artigos_2}$, é identificado o elemento multi-ocorrência tupla_artigo$_2$. Assim, o caminho multi-ocorrência bib_2$/artigos$_2$/tupla_artigo$_2$ é selecionado. A atividade $A_{1.3}$ verifica que existe afinidade semântica entre os caminhos $med/publicação e bib_2$/artigos$_2$/tupla_artigo$_2$. Com isso, o EgC deve informar qual das ACs se aplica a estas coleções. O EgC verifica que a coleção $med/publicação contém a coleção bib_2$/artigos$_2$/tupla_artigo$_2$. Desta forma, é gerada a seguinte AC de Subconjunto:

Ψ: [$med/publicação] $\supset$ [bib_2$/artigos$_2$/tupla_artigo$_2$].

Exemplo 6.3: Consideremos os esquemas XMLS+ F_1 e F_2 da Figura 6.5. Considerando F_1 como S_1, é identificada a seguinte AC: Ψ:[$raiz_F$_1$/livro$_1$]$\equiv$[$raiz_F$_2$/autores$_2$/autor$_2$/publicações$_2$/livros$_2$/livro$_2$]. Considerando F_1 como S_2, é identificada a AC: Ψ: [$raiz_F$_2$/autores$_2$/autor$_2$] $\equiv$ [$raiz_F$_1$/livro$_1$/autor$_1$].

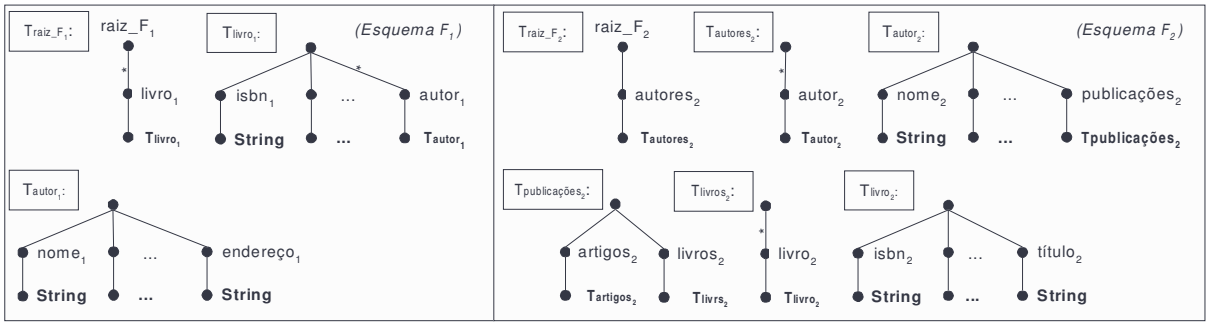


Figura 6.5: ***Esquemas XMLS+ de F_1 e F_2***

Como vemos, as ACs geradas podem ser diferentes, dependendo do esquema que é considerado como o S_1. Supondo que F_2 seja um mediador (esquema S_1), então temos que gerar as ACs de F_2 com relação à F_1 e, portanto, F_1 deve ser o esquema S_2. Estas assertivas é que definem precisamente como sintetizar os elementos do mediador a partir dos elementos da fonte local F_1.

6.3.2 Identificando ACs de Caminhos

A identificação das AC de Caminhos relacionando caminhos dos tipos T_{11} e T_{21} consiste de quatro atividades, vide *Workflow* W_4 na Figura 6.6. A atividade $A_{2.2.1}$ identifica as ACCs relacionando uma ligação de T_{11} e uma ligação de T_{21}. A atividade $A_{2.2.2}$ identifica as ACCs relacionando uma ligação de T_{11} e um caminho de T_{21}. A atividade $A_{2.2.3}$ identifica as ACCs relacionando uma ligação de T_{21} e um caminho de T_{11}. A atividade $A_{2.2.4}$ identifica as ACCs relacionando um caminho de T_{11} e um caminho de T_{21}.

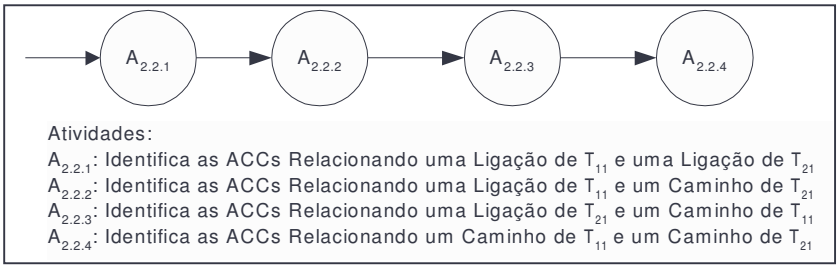


Figura 6.6: ***Workflow W_4***

No restante desta Seção, considere ℓ_{11}: $T_{11} \rightarrow T_{12}$, ℓ_{12}: $T_{12} \rightarrow T_{13}$, …, ℓ_{1n-1}: $T_{1n-1} \rightarrow T_{1n}$ ligações de S_1 e ℓ_{21}: $T_{21} \rightarrow T_{22}$, ℓ_{22}: $T_{22} \rightarrow T_{23}$, …, ℓ_{2m-1}: $T_{2m-1} \rightarrow T_{2m}$ ligações de S_2. Sejam $\delta_1 = \ell_{11}/\ell_{12}/\ldots/\ell_{1n}$ um caminho de T_{11} e $\delta_2=\ell_{21}/\ell_{22}/\ldots/\ell_{2m}$ um caminho de T_{21}.

A seguir, propomos uma heurística para determinar quando os caminhos δ_1 e δ_2 têm afinidade semântica, o que indica a possível existência de uma AC de Caminhos entre δ_1 e δ_2.

Antes de apresentarmos a heurística, definimos as condições necessárias para que os caminhos δ_1 e δ_2 tenham afinidade de tipos e afinidade terminológica.

Definição 6.3: *Os caminhos δ_1 e δ_2 têm afinidade de tipos se os tipos T_{1n} e T_{2m} forem semanticamente relacionados ou compatíveis.*

Definição 6.4: *Os caminhos δ_1 e δ_2 têm afinidade terminológica se os termos das ligações ℓ_{1n} e ℓ_{2m} forem semanticamente relacionados ou, no caso dos tipos T_{1n} e T_{2m} serem complexos, se os termos de T_{1n} e T_{2m} forem semanticamente relacionados.*

Heurística 2: *Os caminhos δ_1 e δ_2 têm afinidade semântica se os mesmos tiverem afinidade de tipos ou terminológica.*

A seguir detalhamos as atividades $A_{2.2.1}$, $A_{2.2.2}$, $A_{2.2.3}$ e $A_{2.2.4}$.

Atividade $A_{2.2.1}$: Identifica as ACCs Relacionando uma Ligação de T_{11} e uma Ligação de T_{21}

O *Workflow* W_5, apresentado na Figura 6.7, detalha a atividade $A_{2.2.1}$. As atividades do *Workflow* W_5 são apresentadas a seguir.

- $A_{2.2.1.1}$: *Seleciona Ligações de T_{11}*

  Esta atividade seleciona ligações do tipo T_{11}. Para cada ligação ℓ_{11} selecionada (condição $C_{2.2.1.1}$), é requisitada a atividade $A_{2.2.1.2}$. Após todas as ligações de T_{11} terem sido selecionadas (condição $C_{2.2.1.2}$), é requisitada a atividade $A_{2.2.1.3}$.

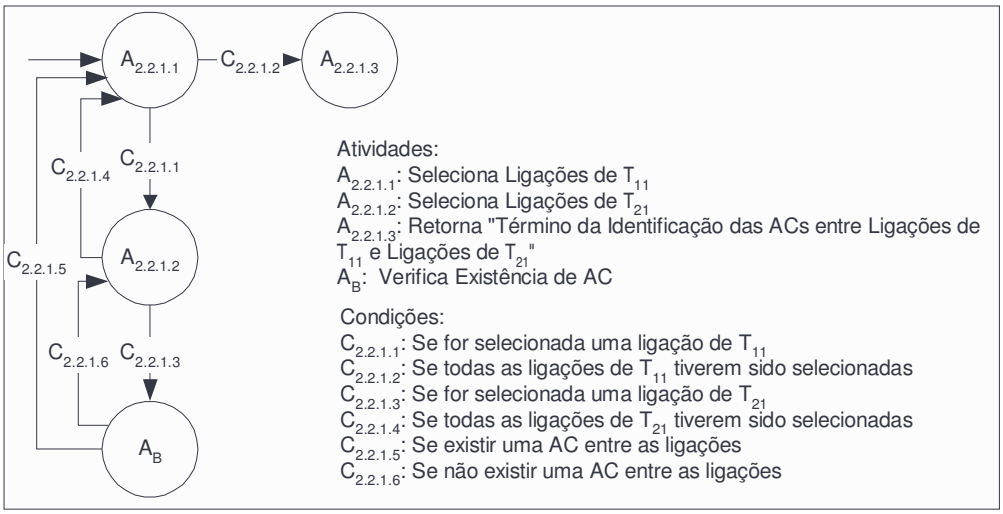


Figura 6.7: *Workflow W5*

- $A_{2.2.1.2}$: *Seleciona Ligações de T_{21}*

Esta atividade seleciona ligações do tipo T_{21}. Para cada ligação l_{21} selecionada (condição $C_{2.2.1.3}$), é requisitada a atividade A_B. Após todas as ligações de T_{21} terem sido selecionadas (condição $C_{2.2.1.4}$), é requisitada a atividade $A_{2.2.1.1}$.

- $A_{2.2.1.3}$: *Retorna "Término da Identificação das ACs entre Ligações de T_{11} e Ligações de T_{21}"*

Esta atividade indica a finalização da identificação das ACs entre as ligações do tipo T_{11} e as ligações do tipo T_{21}. Se ainda existir uma ligação de um tipo que não possua AC com ligações do outro tipo, a atividade $A_{2.2.2}$ será iniciada.

- A_B: *Verifica Existência de AC*

Esta atividade verifica a existência de AC entre as ligações l_{11} e l_{21}. Se existir uma AC entre as ligações (condição $C_{2.2.1.5}$), é requisitada a atividade $A_{2.2.1.1}$. Caso contrário (condição $C_{2.2.1.6}$), é requisitada a atividade $A_{2.2.1.2}$. Esta atividade é descrita no *Workflow W8*, apresentado na Figura 6.8.

O *Workflow W6* detalha a atividade A_B que verifica a existência de uma AC entre caminhos. É importante lembrar que uma ligação é um caminho (composto de apenas uma ligação). Esta atividade utiliza dicionários semânticos para apoiar a identificação da afinidade terminológica entre os caminhos e a Tabela 1 e a Tabela 2 para auxiliar na identificação da afinidade de tipos. A

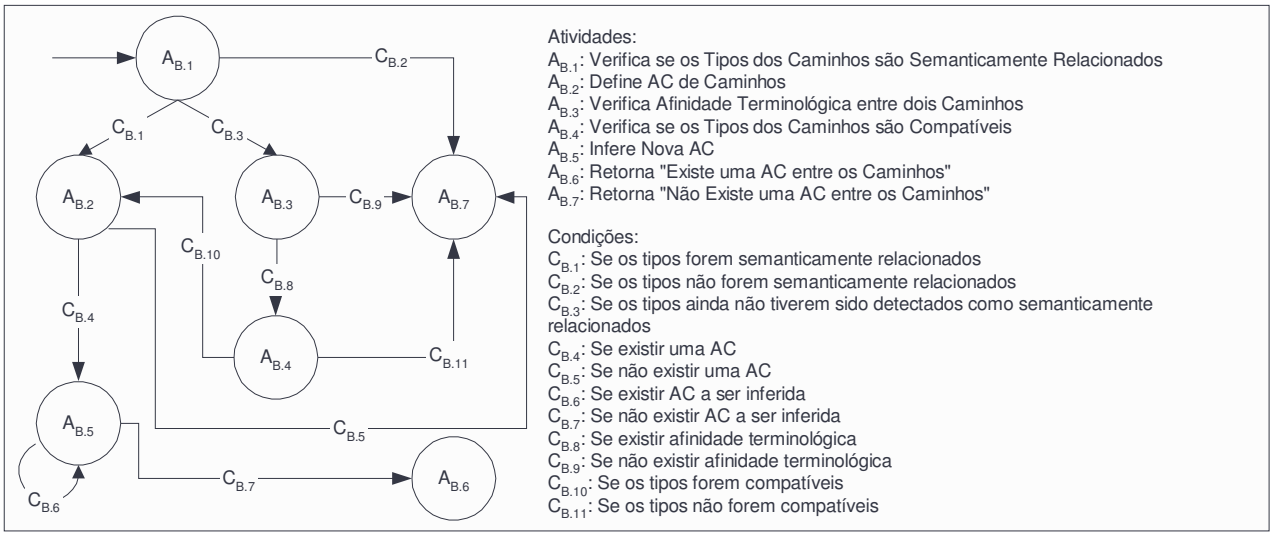


Figura 6.8: ***Workflow W6***

verificação da existência de uma AC entre caminhos é uma subatividade das atividades $A_{2.2.1}$, $A_{2.2.2}$, $A_{2.2.3}$ e $A_{2.2.4}$. A seguir, descrevemos cada atividade do *Workflow* W_6.

- $A_{B.1}$: *Verifica se os Tipos dos Caminhos são Semanticamente Relacionados*

Esta atividade verifica se os tipos dos caminhos δ_1 e δ_2 são semanticamente relacionados. Para isso, deve ser verificado se os tipos destes caminhos estão na Tabela 1. Se estiverem (condição $C_{B.1}$), é requisitada a atividade $A_{B.2}$. Senão, deve ser verificado se os tipos destes caminhos estão na Tabela 2. Se estiverem (condição $C_{B.2}$), é requisitada a atividade $A_{B.7}$. Senão (condição $C_{B.3}$), é requisitada a atividade $A_{B.3}$.

- $A_{B.2}$: *Define AC de Caminhos*

Dado que os caminhos δ_1 e δ_2 têm afinidade semântica, conforme a Heurística 2, a restrição de ocorrência de δ_1 e δ_2 é analisada para verificar o tipo de correspondência possível entre os mesmos. A Tabela 6.1 mostra quais os possíveis tipos de correspondências de acordo com a restrição de ocorrência de cada caminho. O EgC deve confirmar qual das ACs se aplica a δ_1 e δ_2. Para isso, o mesmo poderá verificar nas fontes correspondentes aos esquemas S_1 e S_2 o relacionamento entre as instâncias destes.

Se for identificada uma AC de Caminhos (condição $C_{B.3}$), é requisitada a atividade $A_{B.7}$. Além disso:

- se os termos de δ_1 e δ_2, ou dos seus tipos, não estiverem contidos no DSI, estes termos, juntamente com o relacionamento semântico entre os mesmos, são incluídos no DSI.

Restrições de Ocorrência	*Tipo da Correspondência entre Caminhos*
δ_1 e δ_2 forem mono-ocorrência	δ_1 e δ_2 são equivalentes Não existe AC entre δ_1 e δ_2
δ_1 for mono-ocorrência e δ_2 for multi-ocorrência	δ_1 está contido em δ_2 Não existe AC entre δ_1 e δ_2
δ_1 for multi-ocorrência e δ_2 for mono-ocorrência	δ_1 contém δ_2 Não existe AC entre δ_1 e δ_2
δ_1 e δ_2 forem multi-ocorrência	δ_1 e δ_2 são equivalentes δ_1 está contido em δ_2 δ_1 contém δ_2 δ_1 e δ_2 são sobrepostos δ_1 e δ_2 são disjuntos, porém relacionados Não existe AC entre δ_1 e δ_2

Tabela 6.1: ***Análise das Restrições de Ocorrência entre Caminhos***

- se os tipos de δ_1 e δ_2 forem complexos e não estiverem na Tabela 1, estes são incluídos na Tabela 1.

Se não for identificada uma AC (condição $C_{B.5}$), é requisitada a atividade $A_{B.7}$. Além disso, se os tipos de δ_1 e δ_2 forem complexos, estes são incluídos na Tabela 2.

- $A_{B.3}$: *Verifica Afinidade Terminológica entre dois Caminhos*

Esta atividade verifica se os caminhos δ_1 e δ_2 possuem afinidade terminológica. Para isso, inicialmente é verificado se os termos destes caminhos possuem afinidade terminológica. No caso de uma ligação, o termo é o nome da ligação. No caso de um caminho, o termo é o nome da última ligação do caminho. Se os termos de δ_1 e δ_2 não possuírem afinidade terminológica e os tipos de δ_1 e δ_2 forem complexos, é verificado se os termos dos tipos possuem afinidade terminológica. Para verificar a afinidade terminológica, é usada a função Verifica_Afinidade_Terminológica, mostrada na Figura 6.2. Se existir (condição $C_{B.8}$), é requisitada a atividade $A_{B.4}$, senão (condição $C_{B.9}$), é requisitada a atividade $A_{B.7}$.

- $A_{B.4}$: *Verifica se os Tipos dos Caminhos são Compatíveis*

Esta atividade verifica se os tipos dos caminhos δ_1 e δ_2 são compatíveis. Para isso, a natureza dos tipos de δ_1 e δ_2 são comparadas:

- No caso dos tipos serem simples, é verificado se são tipos simples compatíveis;
- No caso dos tipos serem complexos, é verificado se os mesmos possuem elementos em comum, com nomes e tipos similares;

- No caso de, pelo menos, um dos tipos ser complexo, o EgC é consultado para informar se estes tipos são compatíveis.

Se δ_1 e δ_2 possuírem tipos compatíveis (condição $C_{B.10}$), é requisitada a atividade $A_{B.2}$. Senão (condição $C_{B.11}$), é requisitada a atividade $A_{B.7}$.

- $A_{B.5}$: *Infere Nova AC*

As regras de inferência, apresentadas na Seção 6.3.4, são aplicadas para inferir nova AC a partir da AC identificada na atividade $A_{B.2}$ e de uma AC identificada anteriormente. Após ser inferida, a AC de Caminhos é apresentada ao EgC para que seja validada. Se for validada e se os tipos de A e B forem complexos, estes são incluídos na Tabela 1.

Esta atividade é realizada enquanto existir AC que tenha sido identificada anteriormente e que não tenha sido comparada com a AC identificada na atividade $A_{B.2}$ (condição $C_{B.6}$). Se não existirem mais assertivas a serem comparadas (condição $C_{B.7}$), é requisitada a atividade $A_{B.6}$.

- $A_{B.6}$: *Retorna "Existe uma AC entre os Caminhos"*

Esta atividade retorna que existe uma AC entre os elementos comparados.

- $A_{B.7}$: *Retorna "Não Existe uma AC entre os Caminhos"*

Esta atividade retorna que não existe uma AC entre os elementos comparados.

A seguir, exemplificamos a identificação de correspondências entre ligações de tipos.

Exemplo 6.4: Considere a comparação dos tipos T_{autor} do esquema Med e T_{autor_1} do esquema Bib_1. É verificado que as ligações publicação de T_{autor} e $(keyref\#1)^{-1}$ de T_{autor_1} possuem tipos semanticamente relacionados. Como as ligações publicação e $(keyref\#1)^{-1}$ são multi-ocorrência, o EgC é consultado para informar sobre o relacionamento entre as instâncias destas ligações. O EgC informa que as instâncias da ligação publicação contém as instâncias da ligação $(keyref\#1)^{-1}$. Desta forma, é gerada a seguinte AC de Subconjunto:

Ψ:[T_{autor}/publicação →]⊃[T_{autor_1}/$(keyref\#1)^{-1}$].

Como existe afinidade terminológica entre as ligações nome e $nome_1$ e os tipos dessas ligações são compatíveis, é gerada a seguinte AC de Equivalência:
Ψ:[Tautor/nome] ≡ [T_{autor_1}/$nome_1$].

Ainda comparando os tipos Tautor e T_{autor_1}, é verificado que existe uma afinidade terminológica entre as ligações instituição e @instituiçãoref$_1$. Entretanto, como o valor de instituição é um tipo simples e o valor de @instituiçãoref$_1$ é uma referência a um tipo complexo, o EgC é consultado para informar se as ligações instituição e @instituiçãoref$_1$ estão relacionadas. O EgC informa que a ligação instituição está relacionada com a ligação $nome_1$ do tipo $T_{instituição_1}$. Assim, é gerada a seguinte AC de Equivalência:
Ψ:[Tautor/instituição] ≡ [T_{autor_1}/@instituiçãoref$_1$ → /$nome_1$] .

Atividade $A_{2.2.2}$: Identifica as ACCs Relacionando uma Ligação de T_{11} e um Caminho de T_{21}

O *Workflow* W_7, apresentado na Figura 6.9, detalha a atividade $A_{2.2.2}$. As atividades do *Workflow* W_7 são apresentadas a seguir.

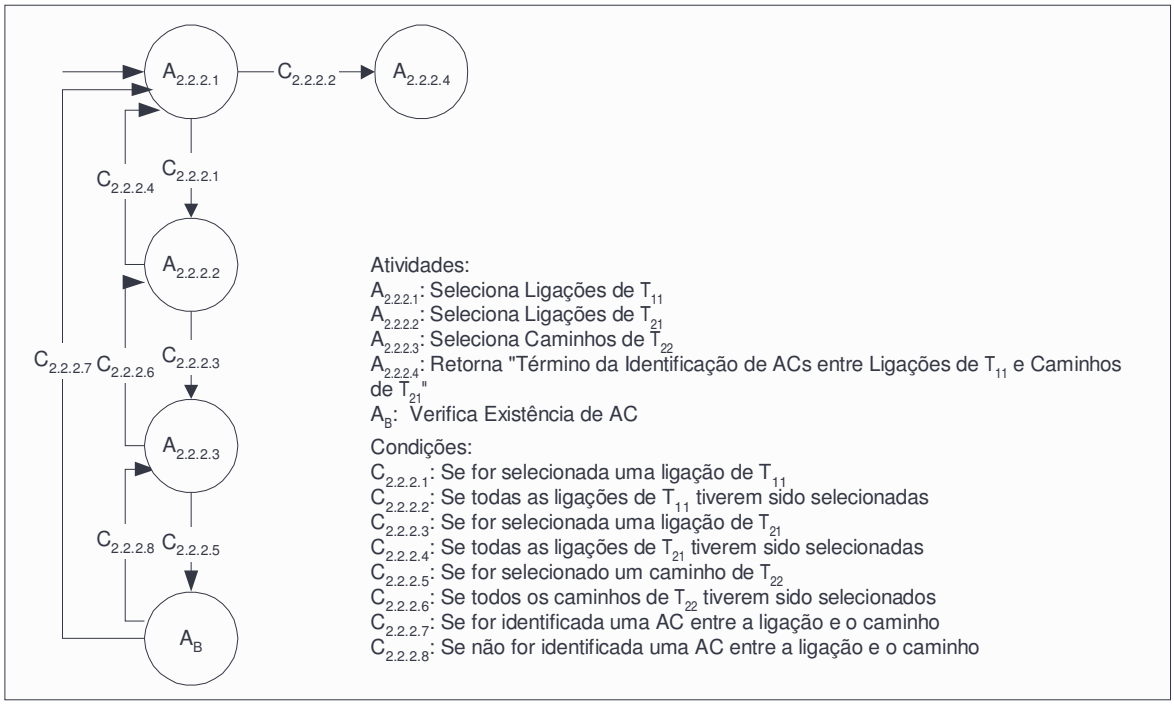


Figura 6.9: *Workflow* W_7

- $A_{2.2.2.1}$: *Seleciona Ligações de* T_{11}

Esta atividade seleciona ligações do tipo T_{11} que ainda não possua AC com ligações de T_{21}. Para cada ligação l_{11} selecionada (condição $C_{2.2.2.1}$), é requisitada a atividade $A_{2.2.2.2}$. Após

todas as ligações de T_{11} terem sido selecionadas (condição $C_{2.2.2.2}$), a atividade $A_{2.2.2.4}$ é requisitada.

- $A_{2.2.2.2}$: *Seleciona Ligações de T_{21}*

Esta atividade seleciona ligações do tipo T_{21}, tal que o tipo da ligação (T_{22}) seja um tipo complexo e a ligação ainda não possua AC com ligações de T_{11}. Para cada ligação ℓ_{21} selecionada (condição $C_{2.2.2.3}$), é requisitada a atividade $A_{2.2.2.3}$. Após todas as ligações de T_{21} terem sido selecionadas (condição $C_{2.2.2.4}$), a atividade $A_{2.2.2.1}$ é requisitada.

- $A_{2.2.2.3}$: *Seleciona Caminhos de T_{22}*

Esta atividade seleciona caminhos de T_{22}. Para cada caminho δ_2 selecionado (condição $C_{2.2.2.5}$), é requisitada a atividade A_B. Após todos os caminhos de T_{22} terem sido selecionados (condição $C_{2.2.2.6}$), a atividade $A_{2.2.2.2}$ é requisitada.

- $A_{2.2.2.4}$: *Retorna "Término da Identificação das ACs entre Ligações de T_{11} e Caminhos de T_{21}"*

Esta atividade indica a finalização da identificação das ACs entre ligações do tipo T_{11} com caminhos do tipo T_{21}. Se existir uma ligação do tipo T_{21} que não possua AC com ligações do tipo T_{11}, a atividade $A_{2.2.3}$ poderá ser iniciada.

- A_B: *Verifica Existência de AC*

Esta atividade verifica a existência de AC entre a ligação ℓ_{11} e o caminho δ_2. Se existir uma AC entre a ligação e caminho (condição $C_{2.2.2.7}$), é requisitada a atividade $A_{2.2.2.1}$, senão (condição $C_{2.2.2.8}$), é requisitada a atividade $A_{2.2.2.3}$. Esta atividade é descrita no *Workflow W_6*, apresentado na Figura 6.8.

A atividade $A_{2.2.3}$ é similar a atividade $A_{2.2.2}$. Finalizada a atividade $A_{2.2.3}$, a atividade $A_{2.2.4}$ pode ser iniciada.

A seguir, exemplificamos a identificação de correspondências entre ligações e caminhos de tipos.

Exemplo 6.5: Considere a comparação dos tipos T_{autor} do esquema Med e $T_{tupla\_autor_2}$ do esquema Bib_2. É verificado que a ligação publicação→ de T_{autor} e o caminho $(FK\#4)^{-1}/FK\#3$ de $T_{tupla\_autor_2}$

possuem tipos semanticamente relacionados. Como a ligação publicação e o caminho $(FK\#4)^{-1}/FK\#3$, são multi-ocorrência, o EgC é consultado para informar sobre o relacionamento entre os mesmos. O EgC informa que as instâncias da ligação publicação→ contém as instâncias do caminho $(FK\#4)^{-1}/FK\#3$. Desta forma, é gerada a AC de Subconjunto:

Ψ:[Tautor/publicação →]⊃[$T_{tupla\_autor_2}$/$(FK\#4)^{-1}$/FK#3].

Atividade $A_{2.2.4}$: Identifica as ACCs Relacionando um Caminho de T_{11} e um Caminho de T_{21}

O *Workflow* W_8, apresentado na Figura 6.10, detalha a atividade $A_{2.2.4}$. As atividades do *Workflow* W_8 são apresentadas a seguir.

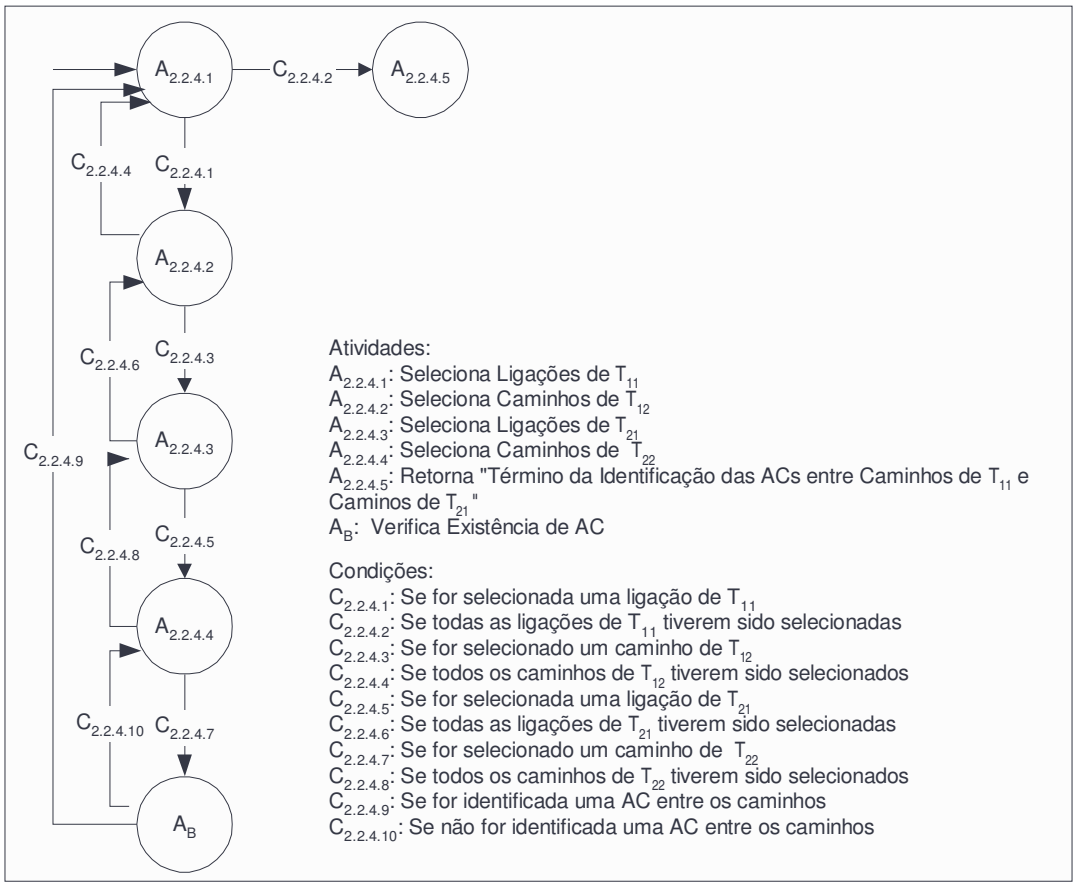


Figura 6.10: ***Workflow W_8***

- $A_{2.2.4.1}$: *Seleciona Ligações de T_{11}*

Esta atividade seleciona ligações de T_{11}, tal que o tipo da ligação (T_{12}) seja um tipo complexo e a ligação ainda não possua AC com ligações de T_{21}. Para cada ligação l_{11} selecionada (condição $C_{2.2.4.1}$), é requisitada a atividade $A_{2.2.4.2}$. Após todas as ligações de T_{11} terem sido selecionadas (condição $C_{2.2.4.2}$), a atividade $A_{2.2.4.5}$ é requisitada.

- $A_{2.2.4.2}$: *Seleciona Caminhos de* T_{12}

Esta atividade seleciona caminhos do tipo T_{12}. Para cada caminho δ_1 selecionado (condição $C_{2.2.4.3}$), é requisitada a atividade $A_{2.2.4.3}$. Após todos os caminhos de T_{12} terem sido selecionados (condição $C_{2.2.4.4}$), a atividade $A_{2.2.4.1}$ é requisitada.

- $A_{2.2.4.3}$: *Seleciona Ligações de* T_{21}

Esta atividade seleciona ligações de T_{21}, tal que o tipo da ligação (T_{22}) seja um tipo complexo e a ligação ainda não possua AC com ligações de T_{11}. Para cada ligação l_{21} selecionada (condição $C_{2.2.4.5}$), é requisitada a atividade $A_{2.2.4.4}$. Após todas as ligações de T_{21} terem sido selecionadas (condição $C_{2.2.4.6}$), a atividade $A_{2.2.4.2}$ é requisitada.

- $A_{2.2.4.4}$: *Seleciona Caminhos de* T_{22}

Esta atividade seleciona caminhos do tipo T_{22}. Para cada caminho δ_2 selecionado (condição $C_{2.2.4.7}$), é requisitada a atividade A_B. Após todos os caminhos de T_{22} terem sido selecionados (condição $C_{2.2.4.8}$), a atividade $A_{2.2.4.3}$ é requisitada.

- $A_{2.2.4.5}$: *Retorna "Término da Identificação das ACs entre Caminhos de* T_{11} *e Caminhos de* T_{21}*"*

Esta atividade indica a finalização da identificação das ACs entre caminhos do tipo T_{11} e caminhos do tipo T_{21}. Em seguida, o EgC poderá gerar as correspondências entre os caminhos que não foram identificadas semi-automaticamente e, assim, finalizar a geração das ACs de Caminhos. No caso do EgC gerar correspondências entre caminhos cujos os tipos sejam complexos, estes são incluídos na Tabela 1.

- A_B: *Verifica Existência de AC*

Esta atividade verifica a existência de AC entre os caminhos δ_1 e δ_2. Se existir uma AC entre os caminhos (condição $C_{2.2.4.9}$), é requisitada a atividade $A_{2.2.4.1}$. Caso contrário (condição $C_{2.2.4.10}$), é requisitada a atividade $A_{2.2.4.4}$. Esta atividade é descrita no *Workflow* W_6, apresentado na Figura 6.8.

A seguir, exemplificamos a identificação de correspondências entre caminhos de tipos.

Exemplo 6.6: Considere a comparação dos tipos $T_{empregado_1}$ do esquema S_1 e $T_{empregado_2}$ do esquema S_2, apresentados na Figura 6.11. Supondo que a ligação $dept_1$ de $T_{empregado_1}$ não possui AC com nenhuma ligação ou caminho de $T_{empregado_2}$ e que a ligação $projeto_2$ de $T_{empregado_2}$ não possui AC com nenhuma ligação ou caminho de $T_{empregado_1}$, os caminhos destas ligações são percorridos para verificar se as mesmas possuem caminhos com afinidade semântica.

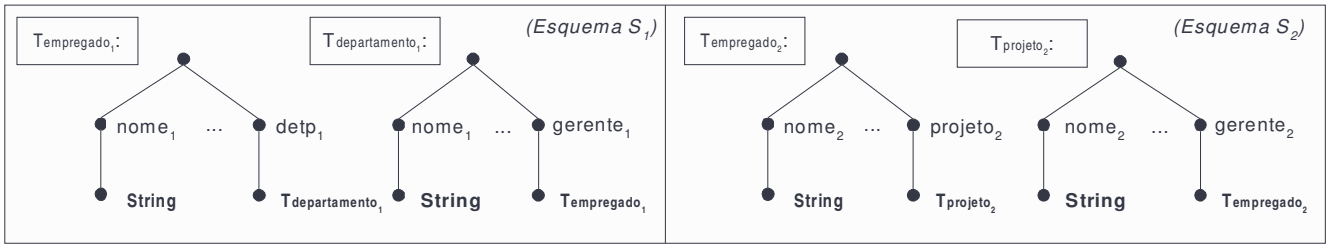


Figura 6.11: ***Tempregado1/dept1/gerente1 corresponde a Tempregado2/projeto2/gerente2***

É verificado que o caminho $dept_1/gerente_1$ de $T_{empregado_1}$ e o caminho $projeto_2/gerente_2$ de $T_{empregado_2}$ têm afinidade semântica. Assim, o EgC é consultado para informar sobre o relacionamento entre os mesmos. O EgC informa que as caminhos são equivalentes. Desta forma, é gerada a AC de Equivalência:

Ψ:[$T_{empregado_1}/dept_1/gerente_1$] ≡ [$T_{empregado_2}/projeto_2/gerente_2$].

Como podemos observar, a heurística definida nesta Seção permite capturar correspondências precisas entre os elementos dos esquemas, mesmo quando esses estão estruturados de formas diferentes.

6.3.3 Identificando ACs de Elementos

Para combinar informações de múltiplas fontes de informação, é necessário determinar quando representações distintas de elementos em diferentes fontes correspondem ao mesmo elemento no mundo real [ZHKF95]. Por exemplo, suponha duas fontes de informação, Estudante_FI e Empregado_FI, as quais contêm informações sobre estudantes e empregados de uma determinada universidade. Um mediador, Mediador_Est&Emp, deverá manter informações sobre pessoas cujos elementos são obtidos da "fusão" dos elementos em Estudante_FI com elementos correspondentes em Empregado_FI. Para isso, precisamos definir uma função de mapeamento (ou função de *matching*) que especifique quando um elemento ***estudante*** na fonte Estudante_FI e um elemento ***empregado*** na fonte Empregado_FI são considerados semanticamente equivalentes, i.é,

correspondem ao mesmo objeto do mundo real. Neste trabalho, usamos as Assertivas de Correspondência de Elementos para definir funções de *matching*.

De acordo com o XMLS+Matcher, uma AC de Elementos deve ser especificada para coleções globais, que estão relacionadas através de uma AC de Coleções Globais, e para coleções aninhadas, que estão relacionadas através de uma AC de Caminhos. Em geral, estas coleções (globais ou aninhadas) possuem uma chave comum a qual é usada como função de *matching*. Dessa forma, dadas suas coleções C_1 e C_2 relacionadas através de uma ACCG ou ACC, com chaves $K_1 = \{\delta_{11}, ..., \delta_{1n}\}$ e $K_2 = \{\delta_{21}, ..., \delta_{2n}\}$ tais que $T_1/\delta_{1i} \equiv T_2/\delta_{2i}$, $1 \leq i \leq n$, então é definida a AC de Elementos dada por $[T_1,\{\delta_{11}, ..., \delta_{1n}\}] \equiv [T_2,\{\delta_{21}, ..., \delta_{2n}\}]$, onde T_1 e T_2 são os tipos dos elementos de C_1 e C_2, respectivamente.

A seguir, apresentamos alguns exemplos de AC de Elementos.

Exemplo 6.7: Considere as coleções globais $med/publicação do esquema Med e $bib_1/livro_1$ do esquema Bib_1. Como estas coleções estão relacionadas através de uma AC de Coleções Globais, as chaves associadas aos elementos destas coleções devem ser comparadas. A chave da coleção $bib_1/livro_1$ é o elemento $isbn_1$. Entretando, não existe uma chave associada à coleção $med/publicação. Assim, o EgC é consultado para informar a chave desta coleção. O EgC informa que a chave é o elemento isbn. Em seguida, as chaves são comparadas e é identificada AC de Elementos abaixo:
Ψ: [$T_{publicação}$, {isbn}] ≡ [T_{livro_1}, {$isbn_1$}].

Exemplo 6.8: No caso de uma coleção aninhada que referencia elementos de uma coleção global, a chave da coleção aninhada pode ser especificada como um subconjunto da chave da coleção global.

Para exemplificar esta situação, considere o tipo raiz Tempresa que contém as coleções globais pessoa do tipo Tpessoa e dependente do tipo Tdependente, e a coleção aninhada dep-pessoa contida na definição do tipo Tpessoa, apresentados na Figura 6.12. A chave da coleção global pessoa é [Tpessoa, {id_pessoa}] e a chave da coleção global dependente é [Tdependente, {id_pessoa, id_dependente}]. Como os dependentes de uma

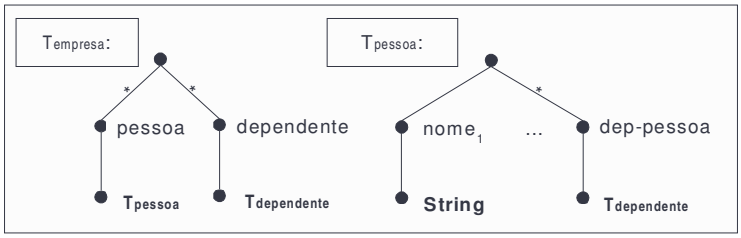


Figura 6.12: *Tpessoa contém a Coleção Aninhada dep-pessoa*

pessoa fazem parte da coleção global dependente, a chave gerada para a coleção aninhada dep-pessoa contida no tipo Tpessoa será um subconjunto da chave da coleção global dependente, ou seja, a chave da coleção aninhada será [Tdependente, {id_dependente}].

6.3.4 Regras de Inferência

Nesta Seção, apresentamos as regras de inferência, as quais, com base em AC identificadas anteriormente, podem inferir outras ACs. Assim, estas regras servem para: (i) otimizar o processo para identificação de correspondências, visto que identificam novas correspondências, e (ii) checar a consistência das correspondências identificadas. Estas regras podem ser utilizadas tanto para inferir correspondências entre os esquemas locais quanto correspondências entre esquemas locais e o esquema do mediador.

Considere A, B e C coleções globais ou caminhos em esquemas XMLS+. Dado que A e B e A e C estão relacionados, as regras apresentadas abaixo inferem as possíveis correspondências entre B e C.

Regra de Inferência 1: Se A ≡ B e A ≡ C, então B ≡ C;
Regra de Inferência 2: Se A ≡ B e A ⊂ C, então B ⊂ C;
Regra de Inferência 3: Se A ≡ B e A Φ C, então B Φ C;
Regra de Inferência 4: Se A ≡ B e A | C, então B | C;
Regra de Inferência 5: Se A ⊂ B e A ⊂ C, então B ⊂ C ou C ⊂ B ou B ≡ C ou B Φ C;
Regra de Inferência 6: Se B ⊂ A e C ⊂ A, então C ⊂ B ou B ⊂ C ou B | C ou B ≡ C ou B Φ C;
Regra de Inferência 7: Se A ⊂ B e A Φ C, então B Φ C ou C ⊂ B;
Regra de Inferência 8: Se A ⊂ B e A | C, então B Φ C ou C ⊂ B ou B | C;
Regra de Inferência 9: Se A Φ B e A Φ C, então BΦ C ou B | C ou C ⊂ B ou B ⊂ C ou C ≡ B;
Regra de Inferência 10: Se A Φ B e A | C, então B Φ C ou C ⊂ B ou B | C.

CAPÍTULO 7

CONCLUSÕES

Neste trabalho propomos um método, denominado XMLS+Matcher, para *matching* de esquemas XMLS+. As principais contribuições do trabalho são:

(1) Propomos uma notação gráfica, denominada XML Schema Semântico (XMLS+), para representar graficamente os esquemas XML Schema. XMLS+ incorpora mais semântica aos esquemas, uma vez que permite representar relacionamentos semânticos implícitos entre os tipos XML, tais como associação e especialização. A representação destes relacionamentos é fundamental para a identificação de correspondencias entre elementos e caminhos dos tipos.

(2) No enfoque proposto, antes de iniciar o processo de *matching*, cada esquema deve ser mapeado em XMLS+. Nesse trabalho, descrevemos os passos do mapeamento para gerar o esquema XMLS+ para um esquema XML Schema, para um esquema relacional e para um esquema orientado a objetos.

(3) Apresentamos a definição formal dos vários tipos de Assertiva de Correspondência que são utilizados para especificar formalmente as correspondências entre esquemas XMLS+. A vantagem desse formalismo é que permite definir de forma precisa as correspondências entre o esquema do mediador e os esquemas locais. O formalismo proposto permite especificar várias formas de correspondências, inclusive casos onde os esquemas têm estruturas diferentes. Os sistemas de integração utilizam as correspondências entre o esquema do mediador e os esquemas locais para determinar como as consultas definidas na visão global serão respondidas a partir de consultas nas fontes locais.

(4) Propusemos um conjunto de heurísticas que auxiliam o Engenheiro do Conhecimento na identificação das assertivas, quando estas não forem de fácil identificação. Para isso, as heurísticas utilizam as informações disponíveis nos esquemas, tais como nome, tipo e restrição de ocorrência dos elementos, assim como as informações provenientes de dicionários semânticos. Os dicionários semânticos são consultados para otimizar o processo de *matching*, identificando os elementos com maior probabilidade de estarem relacionados. Além das heurísticas, cada passo do XMLS+Matcher também é apoiado por um conjunto de regras de inferência. Estas regras inferem outras correspondências com base nas correspondências identificadas anteriormente. Desta forma, servem tanto para identificar novas correspondências, otimizando o processo, como para validar as correspondências já identificadas, checando a consistência das correspondências identificadas.

Apresentamos um estudo de caso onde usamos o XMLS+Matcher para identificar as Assertivas de Correspondência para um mediador que integra dados de duas fontes XML com os de uma fonte relacional. Como vimos, apesar dos esquemas possuírem estruturas diferentes, o formalismo proposto permitiu especificar precisamente as correspondências entre estes. Mostramos também um exemplo de como as Assertivas de Correspondência do Mediador são usadas para identificar os caminhos nas fontes locais que contêm informações relevantes para uma consulta global.

Comparamos nosso trabalho com outros enfoques para identificação de correspondências [BBVC98, DRRS+02, MBR01, MCH02] usando uma taxonomia das técnicas para *matching* de esquemas. A Tabela 7.1 mostra um resumo desta comparação.

	ARTEMIS [BBVC98] e [MCH02]	**Cupid [MBR01] e Xyleme [DRRS+02]**	**XMLS+Matcher**
Representação Comum	Modelo Semântico baseado no OO	Estrutura de Árvore	Modelo Semântico baseado em Árvore
Baseada em Esquema	Sim	Sim	Sim
Baseada em Instâncias	Sim	Não	Sim
Tipo das Correspondências Identificadas	De Classes e de Atributos	De Caminhos (Não diferencia classe de atributo)	Coleções Globais, Caminhos e Elementos

Tabela 7.1: ***Comparação entre XMLS+Matcher e Outros Enfoques para Matching de Esquemas***

Como trabalhos futuros, pretendemos: (i) Estender o XMLS+ com novos tipos de Assertivas de Correspondência de forma a permitir especificar outras formas de correspondência entre esquemas. (ii) Enriquecer o dicionário semântico interno com outros tipos de relacionamento semântico, tais como o Merônimo e Holônimo [Mil95]; (iii) Fazer uso de técnicas de análise das instâncias persistentes nas fontes para auxiliar na identificação da afinidade extensional entre os esquemas; (iv) Identificar novas heurísticas e regras de inferência; (v) Desenvolver um ferramenta semi-automatizada para suportar o XMLS+Matcher. Esta ferramenta é parte essencial do ambiente para manutenção e geração de mediadores proposto no projeto ADaWeb.

REFERÊNCIAS BIBLIOGRÁFICAS

[Abi97] S. Abiteboul. **Querying Semi-Structured Data**. In Proc. of the 6th Int'l Conf. on Database Theory, p. 1-18, Delphi, Grécia, Janeiro 1997.

[Abr02] F. Abreu. **Reformulação de Consultas em Sistemas de Integração de Dados baseados em XML**. Dissertação de Mestrado, Universidade Federal do Ceará, Fortaleza, Ceará, Brasil, *em andamento*, 2002.

[ABS00] S. Abiteboul, P. Buneman e D. Suciu. **Data on the Web: from Relations to Semistructured Data and XML**. 1ª ed, Morgan Kaufmann Publishers, San Francisco, California, USA, 2000.

[ADaWeb01] **ADaWeb: Acesso a Dados Web - Modelos, Técnicas e Ferramentas**, 2001. Projeto da Universidade Federal do Ceará em parceria com a Universidade de Fortaleza e a Universidade Federal de Pernambuco. Maiores informações: http://www.mcc.ufc.br/arida/adaweb/.

[AKMP01] J. Ambite, C. Knoblock, I. Muslea e A. Philpot. **Compiling Source Description for Efficient and Flexible Information Integration**. Journal of Intelligent Information Systems, vol. 16, n. 2, Março, 2001.

[BBVC98] S. Bergamaschi, D. Beneventano, M. Vincini e S. Castano. **MOMIS: An intelligent system for the integration of semi-structured, structured sources**. Technical Report, MO Universita' degli Studi di Modena, Novembro 1998.

[BBC+01] A. Berglund, S. Boag, D. Chamberlin, M. Fernandez, M. Kay, J. Robie e J. Siméon. **XML Path Language (XPath) 2.0**. W3C Working Draft, Dezembro 2001. Disponível em: http://www.w3.org/TR/xpath20.

[BBM01] D. Beneventano, S. Bergamaschi e F. Mandreoli. **Extensional Knowledge for Semantic Query Optimization in a Mediator Based System**. In Proc. of Int'l Workshop on Foundations of Models for Information Integration, Setembro, 2001.

[BCFF+02] S. Boag, D. Chamberlin, M. Fernandez, D. Florescu, J. Robie, J. Siméon e M. Stefanescu. **XQuery 1.0: An XML Query Language**. W3C Working Draft, Abril 2002. Disponível em: http://www.w3.org/TR/xquery/.

[BG00] D. Brickley e R. Guha. **Resource Description Framework (RDF) Schema Specification 1.0**. W3C Candidate Recommendation, Março 2000. Disponível em: http://www.w3.org/TR/rdf-schema.

[BGL+99] C. Baru, A. Gupta, B. Ludascher, R. Marciano, Y. Papakostatinou, P. Velikhov e V. Chu. **XML-Based Information Mediation with MIX**. In Proc. of ACM SIGMOD Conf. on Management of Data, p. 597-599, Philadelphia, Pennsylvania, USA, Junho 1999.

[BL01] A. Bonifati e D. Lee. **Technical Survey of XML Schema, Query Languages**. Disponível em http://www-rocq.inria.fr/~bonifati/Pubs/bole01.pdf, Agosto 2002.

[BLN86] C. Batini, M. Lenzerini e S. B. Navathe. **A Comparative Analysis of Methodologies for Database Schema Integration**. ACM Computing Surveys, p. 323-364, vol. 18 nº 4, Dezembro 1986.

[BM01] P. Biron e A. Malhotra. **W3C Recommendation - XML Schema Part 2: Datatypes**. http://www.w3.org/TR/xmlschema-2/, Maio 2001.

[BPMM00] T. Bray, J. Paoli, C. Sperberg-McQueen e E. Maler. **Extensible Markup Language (XML) 1.0 (Second Edition)**. W3C Recommendation. Outubro 2000.

[Bun97] P. Buneman. **Semi-structured data**. In Proc. of ACM Symposium on Principles of Database Systems. Tucson, Arizona, p.117-121, Tucson, Arizona, Maio 1997.

[CA99] S. Castano e V. De Antonellis. **Building Views over Semistructured Data Sources**. In Proc. of Conf on Conceptual Modeling, p. 146-160, Paris, France, Novembro 1999.

[CA97] S. Castano e V. De Antonellis. **Deriving Global Conceptual Views from Multiple Information Sources**. In Proc. of Conf on Conceptual Modeling, p. 44-55, Los Angeles, California, USA, 1997.

[Cat99] T. Catarci. **Web-based Information Access**. In Proc. of the Int'l Conf. on Cooperative Information Systems, p. 10-19, Edinburgh, Scotland, Setembro 1999.

[CBB+97] R. Cattell, D. Barry, D. Bartels, M. Berler, J. Eastman, S. Gamerman, D. Jordan, A. Spreinger, H. Strickland e D. Wade. **The Object Database Standard ODMG 2.0**. Morgan Kaufman Publishers. 1997.

[CCS00] V. Christophides, S. Cluet e J. Siméon. **On Wrapping Query Languages, Efficient XML Integration**. In Proc. of ACM SIGMOD Conf. on Management of Data, p. 141-152, Dallas, Texas, USA, Maio 2000.

[CDSS98] S. Cluet, C. Delobel, J. Siméon e K. Smaga. **Your Mediators Need Data Conversion!**. In Proc. of ACM SIGMOD Conf. on Management of Data, p. 177-188, Seattle, Washington, USA, Junho 1998.

[CGMH94] S. Chawathe, H. Garcia-Molina e J. Hammer. **The TSIMMIS Project: Integration of Heterogeneous Information Sources**. In Proc. of Meeting of the Information Processing Society of Japan, p.7-18, Tokyo, Japan, Outubro 1994.

[Che76] P. Chen. **The Entity-Relationship Model: Toward a Unified View of Data**. ACM Transactions on Database Systems, p. 9-36, vol. 1, n. 1, Março 1976.

[CS00] S. Cluet e J. Siméon. **YATL: A Functional, Declarative Language for XML**. Draft Manuscritp. Março 2000. Disponível em http://www-db.research.bell-labs.com/user/simeon/icfp.ps, Agosto 2002.

[DDH01] A. Doan, P. Domingos e A. Halevy. **Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach**. SIGMOD p. 509-520, Santa Barbara, CA, USA, Maio 2001.

[DFF+99] A. Deutsch, M. Fernandez, D. Florescu, A. Levy e D. Suciu. **XML-QL: A Query Language for XML**. In Proc. of Int'l World Wide Web Conf., Toronto, Maio 1999.

[DRRS+02] C. Delobel, C. Reynaud, M. Rousset, J. Sirot e D. Vodislav. **Semantic Integration in Xyleme: a Uniform Tree-Based Approach,** Data and Knowledge Engineering Review, 2002.

[EL95] J. Eder e W. Liebhart. **The Workflow Activity Model WAMO**. In Proc. of the Int'l Conf. on Cooperative Information Systems, Vienna, Austria, May 1995.

[EN00] R. Elmasri e S. Navathe. **Fundamentals of Database Systems**. Addison-Wesley 3ª ed, 2000.

[ERS99] A. Elmagarmid, M. Rusinkeiwicz e A. Sheth. **Management of Heterogeneous, Autonomous Database Systems**. Morgam Kaufmann Publishers 1ª ed, 1999.

[Fal01] D. Fallside. **W3C Recommendation - XML Schema Part 0: Primer**. Disponível em: http://www.w3.org/TR/xmlschema-0/, Maio 2001.

[FLM98] D. Florescu, A. Levy e A. Mendelzon. **Database Techniques for the World Wide Web: A Survey**. In ACM SIGMOD Record, p. 59-74, vol. 27 n. 3, Setembro 1998.

[FR01] M. Fernandez e J. Robie. **XML Query Data Model**. Working Draft. Disponível em http://www.w3.org/TR/2001/WD-query-datamodel-20010215/, Fevereiro 2001.

[FSW00] M. Fernandez, J. Siméon e P. Wadler. **XML Query Languages: Experiences, Exemplars**. Disponível http://www-db.research.bell-labs.com/user/simeon/xquery.html, Dezembro 2000.

[GMW99] R. Goldman, J. McHugh e J. Widom. **From Semistructured Data to XML: Migrating the Lore Data Model, Query Language.** Technical Report, Stanford University - Departament of Computer Science, Março 1999.

[GSN99] G. Gardarin, F. Sha e T. Ngoc. **XML-based Components for Federating Multiple Heterogeneous Data Sources**. In Proc. of Int'l Conf. on Conceptual Modeling, p. 506-519, Paris, France, Novembro 1999.

[HMN+99] L. Haas, R. Miller, B. Niswonger, M. Roth, P. Schwarz e E. Wimmers. **Transforming Heterogeneous Data with Database Middleware: Beyond Integration**. Data Engineering Bulletin, p. 31-36, vol. 22, n. 1, Março 1999.

[IFF+99] Z. Ives, D. Florescu, M. Friedman, A. Levy e D. Weld. **An Adaptive Query Execution System for Data Integration**. In Proc. of ACM SIGMOD Int'l Conf. on Management of Data, p. 299-310, Philadelphia, Pennsylvania, USA, Junho 1999.

[KMAA+01] C. Knoblock, S. Minton, J. Ambite, N. Ashish, A. Philpot e S. Tejada. **The Ariadne Approach to Web-Based Information Integration**. In Int'l Journal of Cooperative Information Systems, p. 145-169, vol. 10, n. 1-2, 2001.

[Len02] M. Lenzerini. **Data Integration: A Theoretical Perspective**. In Proc. of Symposium on Principles of Database Systems, p. 233-246, Madison, Winsconsin, USA, Junho 2002.

[LC00] D. Lee e W. Chu. **Comparative Analysis of Six XML Schema Languages**. In ACM SIGMOD Record, p. 76-87, vol. 29, n. 3, Setembro 2000.

[Lós98] B. Lóscio. **Atualização de Múltiplas Bases de Dados através de Mediadores**. Dissertação de Mestrado, Universidade Federal do Ceará, 1998.

[MBR01] J. Madhavan, P. Bernstein e E. Rahm. **Generic Schema Matching with Cupid**. In Proc. Int'l Conf. of Very Large Data Bases, p. 49-58, Roma, Itália, Setembro 2001.

[MCH02] R. Mello, S. Castano e C. Heuser. **A Method for the Unification of XML Schemata**. Information & Software Technology, p. 241-249, vol. 44, n. 4, Março 2002.

[Mil95] A. Miller. **WordNet: A Lexical Database for English**. Communications of the ACM, p. 39-41, vol. 38, n. 11, Novembro 1995.

[MHH+01] R. Miller, M. Hernández, L. Haas, L. Yan, C. Ho, R. Fagin e L. Popa. **The Clio Project: Managing Heterogeneity**. In ACM SIGMOD Record, p. 78-83, vol. 30, n. 1, Março 2001.

[Mir97] I. Mirbel. **Semantic Integration of Conceptual Schemas**. Data & Knowledge Engineering, p. 183-195, vol. 21, n. 1, Janeiro 1997.

[MRJ99] P. Missier, M. Rusinkiewicz e W. Jin. **Multidatabase Languages**. In A. Elmagarmid, M. Rusinkeiwicz, A. Sheth, editors, Management of Heterogeneous, Autonomous Database Systems, chapter 7, p. 175-216. Morgam Kaufmann Publishers, 1ª ed, 1999

[MK99] A. Morishima e H. Kitagawa. **InfoWeaver: Dynamic, Tailor-Made Integration of Structured Documents, Web, Databases**. In Proc. of the 4th ACM Conf. on Digital Libraries, p. 235-236, Berkeley, CA USA, Agosto 1999.

[OV99] M. Ozsu e P. Valduriez. **Principles of Distributed Database Systems**. Prentice Hall, 2ª ed, 1999.

[Peq00] V. Pequeno. **Auto-Manutenção de Classes de Fusão em Visões de Integração de Dados**. Dissertação de Mestrado, Universidade Federal do Ceará, Fevereiro 2000.

[RB01] E. Rahm e P. Bernstein. **On Matching Schemas Automatically**. Microsoft Research Technical Report 17, University of Leipzig, Fevereiro 2001.

[RR95] V. Ramesh e S. Ram. **A Methodology for Interschema Relationship Identification in Heterogeneous Databases**. In Proc. of the Annual Hawai Int'l Conf. on System Sciences, p. 263-272, 1995.

[SHKC93] S. Shekhar, B. Hamidzadeh, A. Kohli e M. Coyle. **Learning Transformation Rules for Semantic Query Optimization: A Data-driven Approach**. IEEE Transactions on Knowledge, Data Engineering, p. 950-964, vol. 5, n. 6, Dezembro 1993.

[SP94] S. Spaccapietra e C. Parent. **View Integration: a Step Forward in Solving Structural Conflicts**. IEEE Trans. on Software Engineering, p. 258-274, v. 6, n. 2, 1994.

[ST98] I. Schmitt e C. Türker. **An Incremental Approach to Schema Integration by Refining Extensional Relationships**. In Proc. of ACM CIKM Int'l Conf. on Information and Knowledge Management, p. 322-330, Bethesda, Maryland, USA, November 1998.

[TBMM01] H. Thompson, D. Beech, M. Maloney e N. Mendelsohn. **W3C Recommendation - XML Schema Part 1: Structures**. Disponível em: http://www.w3.org/TR/xmlschema-1/, Maio 2001.

[Vid02a] V. Vidal. **Generating Self-Maintainable Outer Union Views in XML Based Mediator**. *submetido*, 2002.

[Vid02b] V. Vidal. **Efficient Maintenance of XML-based Mediated Views using the View´s Correspondence Assertions**. *submetido*, 2002.

[VBO01] V. Vidal, A. Brayner e A. Oliveira. **Self-Maintenance of Materialized Views in XML-Based Mediator**. In Informal Proc. of 1th Int'l Workshop on Data Integration Over the Web (DIWeb), p. 32-46, Interlaken, Switzerland, Junho 2001.

[VL97] V. Vidal e B. Lóscio. **Especificação de Mediadores para Acesso e Atualização de Múltiplas Bases de Dados**. In Proc. of Simpósio Brasileiro de Banco de Dados, Fortaleza, 1997.

[VL99] V. Vidal e B. Lóscio. **Solving the Problem of Semantic Heterogneity in Defining Mediator Update Translators**. In Proc. of Int'l Conf.. on Conceptual Modeling, p. 293-308, Paris, France, 1999.

[VLS01] V. Vidal, B. Lóscio e A. Salgado. **Using Correspondence Assertion for Specifying the Semantics of XML-Based Mediators**. In Proc. of Workshop on Integration of Information on the Web, Rio de Janeiro, Abril 2001.

[W3C] World Wide Web Consortium. Disponível em: http://www.w3.org.

[Wie92] G. Wiederhold. **Mediators in the Arquitecture of Future Information Systems**. IEEE Computer, p. 38-49, vol. 25, n.3, Março 1992.

[ZHKF95] G. Zhou, R. Hull, R. King e J. Franchitti. **Using Object Matching and Materialization to Integrate Heterogeneous Databases**. In Proc. of the Int'l Conf. on Cooperative Information Systems, p. 4-18, Vienna, Austria, Maio 1995.

[ZHK96] G. Zhou, R. Hull e R. King. **Generating Data Integration Mediators that Use Materialization**. Journal of Intelligent Information Systems, p. 199-221, vol. 6, n. 2/3, Maio 1996.

APÊNDICE A

A.1 XML Schema da Árvore do Mediador

```
<xs:schema>
    <xs:element name="ACE" type="xs:string"/>
    <xs:element name="ACs_MedxLocal" type="TACs_MedxLocal"/>
    <xs:complexType name="TACs_MedxLocal">
        <xs:sequence>
            <xs:element name="ACs_ColeçõesGlobais" type="TACCGs"/>
            <xs:element name="ACs_Caminhos" type="TACsC"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="TACCGs">
        <xs:sequence>
            <xs:element name="ColeçãoGlobal" type="TACsCG" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="TACsCG">
        <xs:sequence>
            <xs:element name="nome" type="xs:string"/>
            <xs:element name="ACCG" type="TACCG" minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="TACCG" mixed="true">
        <xs:sequence>
            <xs:element ref="ACE" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="ACCs_ref" type="xs:IDREF" use="required"/>
    </xs:complexType>
    <xs:complexType name="TACsC">
        <xs:sequence>
            <xs:element name="ACCs" type="TACCs" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="TACCs">
        <xs:sequence>
            <xs:element name="ACC" type="TACC" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="tipos" type="xs:ID" use="required"/>
    </xs:complexType>
    <xs:complexType name="TACC">
        <xs:sequence>
            <xs:element name="Ligação" type="xs:string"/>
            <xs:element name="Expressão" type="xs:string" minOccurs="0"/>
            <xs:element ref="ACE" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="ACCs_ref" type="xs:IDREF" use="optional"/>
 </xs:complexType>
</xs:schema>
```

A.2 Documento XML da Árvore do Mediador com Bib_1

```
<ACs_MedxLocal xsi:noNamespaceSchemaLocation="XMLSchema-ArvoredeMed.xsd">
    <ACs_ColeçõesGlobais>
        <ColeçãoGlobal>
            <nome> Autor </nome>
            <ACCG ACCs_ref="TautorxTautor1">Ψ1: [$med/autor] ⊃ [$bib1/autor1]
                <ACE>Ψ4: [Tautor, {nome, e-mail}] ≡ [Tautor1, {nome1, e-mail1}] </ACE>
            </ACCG>
        </ColeçãoGlobal>
        <ColeçãoGlobal>
            <nome> Publicação </nome>
            <ACCG ACCs_ref="TpublicaçãoxTartigo1"> Ψ2: [$med/publicação] ⊃ [$bib1/artigo1]
                <ACE>Ψ20: [Tpublicação, {isbn, título}] ≡ [Tartigo1, {isbn1, título1}] </ACE>
            </ACCG>
            <ACCG ACCs_ref="TpublicaçãoxTlivro1">Ψ3: [$med/publicação] ⊃ [$bib1/livro1]
                <ACE>Ψ21: [Tpublicação, {isbn}] ≡ [Tlivro1, {isbn1}] </ACE>
            </ACCG>
        </ColeçãoGlobal>
    </ACs_ColeçõesGlobais>
    <ACs_Caminhos>
        <ACCs tipos="TautorxTautor1">
            <ACC>
                <Ligação> nome </Ligação>
                <Expressão> Ψ5: [Tautor/nome] ≡ [Tautor1/nome1] </Expressão>
            </ACC>
            <ACC>
                <Ligação> e-mail </Ligação>
                <Expressão> Ψ6: [Tautor/e-mail] ≡ [Tautor1/e-mail1] </Expressão>
            </ACC>
            <ACC ACCs_ref="TpublicaçãoxTartigo1">
                <Ligação> publicação </Ligação>
                <Expressão> Ψ7: [Tautor/publicação → ] ⊃ [Tautor1/(keyref#1)-1] </Expressão>
                <ACE>Ψ8: [Tautor/publicação →, {isbn, título}] ≡ [Tautor1/(keyref#1)-1, {isbn1, título1}]</ACE>
            </ACC>
            <ACC ACCs_ref="TpublicaçãoxTlivro1">
                <Ligação> publicação </Ligação>
                <Expressão> Ψ13: [Tautor/publicação →] ⊃ [Tautor1/(keyref#2) -1] </Expressão>
                <ACE>Ψ14: [Tautor/publicação →, {isbn}] ≡ [Tautor1/(keyref#2)-1, {isbn1}] </ACE>
            </ACC>
            <ACC>
                <Ligação> instituição </Ligação>
                <Expressão> Ψ19: [Tautor/instituição] ≡ [Tautor1/instituiçãoref1 --> /nome1] </Expressão>
            </ACC>
        </ACCs>
        <ACCs tipos="TpublicaçãoxTartigo1">
            <ACC>
                <Ligação> isbn </Ligação>
                <Expressão> Ψ9: [Tpublicação/isbn] ≡ [Tartigo1/isbn1] </Expressão>
            </ACC>
            <ACC>
                <Ligação> título </Ligação>
                <Expressão> Ψ10: [Tpublicação/título] ≡ [Tartigo1/título1] </Expressão>
            </ACC>
            <ACC>
                <Ligação> ano </Ligação>
                <Expressão> Ψ11: [Tpublicação/ano] ≡ [Tartigo1/ano1] </Expressão>
            </ACC>
```

```
            <ACC>
                <Ligação> local-publicação </Ligação>
                <Expressão> Ψ12: [Tpublicação/local-publicação] ≡ [Tartigo1/local-publicação1] </Expressão>
            </ACC>
        </ACCs>
        <ACCs tipos="TpublicaçãoxTlivro1">
            <ACC>
                <Ligação> isbn </Ligação>
                <Expressão> Ψ15: [Tpublicação/isbn] ≡ [Tlivro1/isbn1]  </Expressão>
            </ACC>
            <ACC>
                <Ligação> título </Ligação>
                <Expressão> Ψ16: [Tpublicação/título] ≡ [Tlivro1/título1] </Expressão>
            </ACC>
            <ACC>
                <Ligação> ano </Ligação>
                <Expressão> Ψ17: [Tpublicação/ano] ≡ [Tlivro1/ano1] </Expressão>
            </ACC>
            <ACC>
                <Ligação> editora </Ligação>
                <Expressão> Ψ18: [Tpublicação/editora] ≡ [Tlivro1/editora1] </Expressão>
            </ACC>
        </ACCs>
    </ACs_Caminhos>
</ACs_MedxLocal>
```

A.3 Documento XML da Árvore do Mediador com Bib_2

```
<ACs_MedxLocal xsi:noNamespaceSchemaLocation="XMLSchema-ArvoredeMed.xsd">
    <ACs_ColeçõesGlobais>
        <ColeçãoGlobal>
            <nome> Autor </nome>
            <ACCG ACCs_ref="TautorxTautor2">Ψ1: [$med/autor] ⊃ [$bib2/autores2/tupla_autor2]
                <ACE>Ψ4: [Tautor, {nome, e-mail}] ≡ [Ttupla_autor2, {nome2, e-mail2}] </ACE>
            </ACCG>
        </ColeçãoGlobal>
        <ColeçãoGlobal>
            <nome> Publicação </nome>
            <ACCG ACCs_ref="TpublicaçãoxTtupla_livro2">Ψ2: [$med/publicação] ⊃ [$bib2/livros2/tupla_livro2]
                <ACE>Ψ19: [Tpublicação, {isbn}] ≡ [Ttupla_livro2, {isbn2}] </ACE>
            </ACCG>
            <ACCG ACCs_ref="TpublicaçãoxTtupla_publicação2">
                Ψ3: [$med/publicação] ⊃ [$bib2/publicações2/tupla_publicação2]
                <ACE>Ψ20: [Tpublicação, {isbn}] ≡ [Ttupla_publicação2, {isbn2}] </ACE>
            </ACCG>
        </ColeçãoGlobal>
    </ACs_ColeçõesGlobais>
    <ACs_Caminhos>
        <ACCs tipos="TautorxTtupla_autor2">
            <ACC>
                <Ligação> nome </Ligação>
                <Expressão> Ψ5: [Tautor/nome] ≡ [Ttupla_autor2/nome2] </Expressão>
            </ACC>
            <ACC>
                <Ligação> e-mail </Ligação>
                <Expressão> Ψ6: [Tautor/e-mail] ≡ [Ttupla_autor2/e-mail2] </Expressão>
            </ACC>
```

```
            <ACC ACCs_ref="TpublicaçãoxTtupla_publicação2">
                <Ligação> publicação </Ligação>
                <Expressão> Ψ7: [Tautor/publicação →] ⊃ [Ttupla_autor2/(FK#4)-1/FK#3] </Expressão>
                <ACE>Ψ8: [Tautor/publicação →, {isbn}] ≡ [Ttupla_autor2/(FK#4)-1/FK#3, {isbn2}] </ACE>
            </ACC>
            <ACC ACCs_ref="TpublicaçãoxTtupla_livro2">
                <Ligação> publicação </Ligação>
                <Expressão> Ψ12: [Tautor/publicação →] ⊃ [Ttupla_autor2/(FK#4)-1/FK#3] </Expressão>
                <ACE>Ψ13: [Tautor/publicação →, {isbn}] ≡ [Ttupla_autor2/(FK#4)-1/FK#3, {isbn2}] </ACE>
            </ACC>
            <ACC>
                <Ligação> instituição </Ligação>
                <Expressão> Ψ18: [Tautor/instituição] ≡ [Ttupla_autor2/FK#2/nome2] </Expressão>
            </ACC>
        </ACCs>
        <ACCs tipos="TpublicaçãoxTtupla_publicação2">
            <ACC>
                <Ligação> isbn </Ligação>
                <Expressão> Ψ9: [Tpublicação/isbn] ≡ [Ttupla_publicação2/isbn2 ] </Expressão>
            </ACC>
            <ACC>
                <Ligação> título </Ligação>
                <Expressão> Ψ10: [Tpublicação/título] ≡ [Ttupla_publicação2/título2] </Expressão>
            </ACC>
            <ACC>
                <Ligação> ano </Ligação>
                <Expressão> Ψ11: [Tpublicação/ano] ≡ [Ttupla_publicação2/ano2] </Expressão>
            </ACC>
        </ACCs>
        <ACCs tipos="TpublicaçãoxTtupla_livro2">
            <ACC>
                <Ligação> isbn </Ligação>
                <Expressão> Ψ14: [Tpublicação/isbn] ≡ [Ttupla_livro2/isbn2 ] </Expressão>
            </ACC>
            <ACC>
                <Ligação> título </Ligação>
                <Expressão> Ψ15: [Tpublicação/título] ≡ [Ttupla_livro2/título2] </Expressão>
            </ACC>
            <ACC>
                <Ligação> ano </Ligação>
                <Expressão> Ψ16: [Tpublicação/ano] ≡ [Ttupla_livro2/ano2] </Expressão>
            </ACC>
            <ACC>
                <Ligação> editora </Ligação>
                <Expressão> Ψ17: [Tpublicação/editora] ≡ [Ttupla_livro2/editora2] </Expressão>
            </ACC>
        </ACCs>
    </ACs_Caminhos>
</ACs_MedxLocal>
```

A.4 Documento XML da Árvore do Mediador com Bib_4

```
<ACs_MedxLocal xsi:noNamespaceSchemaLocation="XMLSchema-ArvoredeMed.xsd">
    <ACs_ColeçõesGlobais>
        <ColeçãoGlobal>
            <nome> Publicação </nome>
            <ACCG ACCs_ref="TpublicaçãoxTartigo4">Ψ1: [$med/publicação] ⊃ [$bib4/artigo4]
                <ACE>Ψ3: [Tpublicação, {isbn, título}] ≡ [Tartigo4, {isbn4, título4}] </ACE>
            </ACCG>
        </ColeçãoGlobal>
```

```
        <ColeçãoGlobal>
            <nome> Autor </nome>
            <ACCG ACCs_ref="TautorxTautor4">Ψ2: [$med/autor] ⊃ [$bib4/artigo4/autor4]
                <ACE>Ψ8: [Tautor, {nome}] ≡ [Tartigo4/autor4, {nome4}] </ACE>
            </ACCG>
        </ColeçãoGlobal>
    </ACs_ColeçõesGlobais>
    <ACs_Caminhos>
        <ACCs tipos="TpublicaçãoxTartigo4">
            <ACC>
                <Ligação> isbn </Ligação>
                <Expressão> Ψ4: [Tpublicação/isbn] ≡ [Tartigo4/isbn4]  </Expressão>
            </ACC>
            <ACC>
                <Ligação> título </Ligação>
                <Expressão> Ψ5: [Tpublicação/título] ≡ [Tartigo4/título4] </Expressão>
            </ACC>
            <ACC>
                <Ligação> ano </Ligação>
                <Expressão> Ψ6: [Tpublicação/ano] ≡ [Tartigo4/ano4] </Expressão>
            </ACC>
            <ACC>
                <Ligação> local-publicação </Ligação>
                <Expressão> Ψ7: [Tpublicação/local-publicação] ≡ [Tartigo4/local-publicação4] </Expressão>
            </ACC>
        </ACCs>
        <ACCs tipos="TautorxTautor4">
            <ACC>
                <Ligação> nome </Ligação>
                <Expressão> Ψ9: [Tautor/nome] ≡ [Tautor4/nome4] </Expressão>
            </ACC>
            <ACC ACCs_ref="TpublicaçãoxTlivro2">
                <Ligação> publicação </Ligação>
                <Expressão> Ψ10: [Tautor/publicação →] ⊃ [Tautor4/(autor4)-1 ] </Expressão>
                <ACE>Ψ11: [Tautor/publicação →, {isbn, título}] ≡ [Tautor4/(autor4)-1, {isbn4, título4}] </ACE>
            </ACC>
            <ACC>
                <Ligação> instituição </Ligação>
                <Expressão> Ψ12: [Tautor/instituição] ≡ [Tautor4/instituição4] </Expressão>
            </ACC>
        </ACCs>
    </ACs_Caminhos>
</ACs_MedxLocal>
```