PRESERVANDO A SEMÂNTICA DE ATUALIZAÇÕES NA INTEGRAÇÃO DE VISÕES

Vânia Maria Ponte Vidal

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO EM ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Jano Moreira de Souza, Ph.D.
(Presidente)

Profa. Marianne Winslett, Ph.D.

Profa. Ana Regina C. da Rocha, D.Sc.

Profa. Marta Lima de Queirós Mattoso, D.Sc.

Prof. Rubens Nascimento Melo, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
MAIO DE 1994

VIDAL, VÂNIA MARIA PONTE

Preservando a Semântica de Atualizações na Integração de Visões (Rio de Janeiro) 1994.

xi, 155p. 29,7 cm (COPPE/UFRJ, D.Sc. Engenharia de Sistemas e Computação, 1994)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. banco de dados 2. integração de visões 3. modelo de dados semânticos 4. tradução de atualizações em visões.

I. COPPE/UFRJ II. Título (série)

Aos meus filhos,

Gustavo e Ana Carolina

Agradecimentos,

Gostaria de agradecer às pessoas e instituições que me apoiaram durante o meu trabalho de doutorado na COPPE-UFRJ e na Universidade de Illinois.

Ao Professor Jano Moreira de Souza, pela disponibilidade em aceitar orientar essa tese.

À minha co-orientadora Professora Marianne Winslett, por suas valiosas sugestões e o cuidado que teve ao ler cada detalhe desta tese durante a minha estadia na Universidade de Illinois.

Aos membros da minha banca de tese, Professora Ana Regina Cavalcanti da Rocha, Professor Rubens Nascimento Melo e Professora Marta Lima de Queiróz Mattoso, pelas valiosas sugestões a meu trabalho.

A meu marido, Creto Augusto, por sua ajuda na tradução da tese de inglês para português e por seu amor, amizade e encorajamento ao longo desses anos.

A minha amiga e colega Luciana Ferraz Tomé, pelo seu carinho e incentivo

À Universidade de Illinois em Urbana-Champaign por ter permitido que eu usasse suas excelentes instalações e recursos computacionais.

À UFF pela liberação para a estada em Illinois.

Ao CNPq pelo suporte financeiro, incluindo os recursos necessários para trazer a minha co-orientadora para o Brasil.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para obtenção do grau de Doutor em Ciências (D.Sc.).

PRESERVANDO A SEMÂNTICA DE ATUALIZAÇÕES NA INTEGRAÇÃO DE VISÕES

Vânia Maria Ponte Vidal

MAIO, 1994

Orientador: Jano Moreira de Souza

Programa: Engenharia de Sistemas e Computação

Nessa tese, propomos uma metodologia para integração de visões onde a semântica das atualizações são preservadas durante o processo de integração. Integração de visões ou, como é freqüentemente chamada, integração de esquemas, é um processo que recebe um conjunto de visões projetadas por grupos separados de usuários, e as integra em um esquema conceitual global para toda a organização. Nós propomos dividir integração de visões em três passos: *combinação*, *reestruturação* e *otimização*. No passo de combinação das visões, definimos o *esquema combinado* que contém todas as visões originais e um novo conjunto de restrições as quais expressam de que maneira os dados das diferentes visões estão relacionados. No passo de reestruturação, as visões são reestruturadas de forma que a fusão das partes comuns seja possível. No passo de otimização, reduzimos as redundâncias e o tamanho do esquema.

Nossa metodologia define um conjunto de primitivas de transformação que permitem que a integração de esquemas seja realizada de forma segura e eficiente. Nas primitivas de transformação propostas, o relacionamento entre o esquema original e o esquema transformado é formalmente especificado pelo *mapeamento de instâncias* e pelo *mapeamento de atualizações*. O mapeamento de instâncias especifica de que forma as instâncias do esquema original são mapeadas nas instâncias do esquema transformado e vice-versa. O mapeamento de atualizações especifica de que maneira cada operação de atualização definida sobre o esquema original é transformada em uma operação de atualização definida sobre o esquema

transformado. No nosso enfoque, uma tranformação deve preservar a informação e a semântica das atualizações. Uma transformação *preserva a informação* quando o esquema original e o esquema transformado têm capacidades de informação equivalentes. Uma transformação *preserva a semântica das atualizações* quando qualquer atualização especificada no esquema transformado tem o mesmo efeito de como se o usuário tivesse executado a operação diretamente no esquema original. A preservação da semântica das atualizações é necessária para que os relacionamentos entre as visões e o esquema global, originados durante o processo de integração, reflitam exatamente os relacionamentos entre as visões da forma com foi definido pelo esquema combinado. A garantia formal da preservação da semântica das atualizações é uma das inovações do nosso enfoque e é o objeto central dessa tese.

No nosso enfoque, além de alcançarmos a integração das visões no esquema global obtido, nós também obtemos os mapeadores de instâncias e os tradutores de atualizações das visões. Os mapeadores de instâncias e os tradutores de atualizações das visões são diretamente definidos a partir dos mapeamentos de instâncias e de atualizações entre os vários esquemas intermediários gerados durante o proceso de integração. Nós definimos formalmente as condições para a validação das traduções de atualizações em visões na presença de interferência entre visões e mostramos que, como as tranformações de esquemas na nossa metodologia preservam a informação e a semântica das atualizações, podemos então provar formalmente que as traduções das atualizações nas visões definidas a partir dos mapeamentos de atualizações são válidas.

Nós propomos uma *forma nomal decomposta* para esquemas a qual especifica as propriedades que um esquema deve ter para que o problema de "incompatibilidade de fusão" capturado por determinadas restrições de dependências existenciais não ocorra, e apresentamos um algorítimo para transformar determinados tipos de esquemas em esquemas equivalentes que estão em forma normal decomposta.

Descrevemos também com projetar uma representação orientada a objetos para um esquema conceitual obtido com a nossa metodologia e como se deve traduzir as definições conceituais das visões em definições orientadas a objetos.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfilment of the requirements for the degree od Doctor of Science (D.Sc.).

PRESERVING UPDATE SEMANTICS IN VIEW INTEGRATION

Vânia Maria Ponte Vidal

MAY, 1994

Thesis Supervisor: Jano Moreira de Souza

Department: Engenharia de Sistemas e Computação

In this thesis, we propose a methodology for schema integration where the semantics of the updates are preserved during the view integration process. View integration or, as it is often called, schema integration, is a process that takes a set of views designed by separate user groups, and integrates them into a global conceptual schema for the entire organization. We propose to divide view integration into three steps: combination, restructuring, and optimization. In the view combination step, we define the *combined schema* that contains all original views, plus a new set of constraints that express how data in distinct views are interrelated. The restructuring step is devoted to normalizing the views so that merging becomes possible. The optimization step tries to reduce redundancy and the size of the schema.

Our methodology defines a set of transformation primitives that allows schema integration to be realized in a safe (information preserving) and algorithmic way. In the proposed transformation primitives, the relationship between the original and transformed schema is formally specified by the *instance* and *update mappings*. In the proposed transformation primitives, the relationship between the original and transformed schema is formally specified by the *instance* and *update mappings*. The instance mapping specifies how instances of the original schema are mapped to instances of the transformed schema and vice-versa. The update mapping specifies how each update operation defined under the original schema is transformed into an update operation defined under the transformed schema. In our approach, a transformation must be *information* and *update semantics preserving*. A transformation is *information preserving* when the original and transformed schemas

are of equivalent information capacity. A transformation is *update semantics preserving* when any update specified against the transformed schema has the same effect as if the user had performed the update directly on the original schema. The formal guarantee of preservation of update semantics is a novel feature of our approach and is the focus of the thesis. This is necessary so that we can guarantee that the relationships between each view and the global schema, originated during the view integration process, reflect exactly the relationships between the views as defined by the combined schema.

In our approach, in addition to achieving a schema integration at the schema level, the view instance mappings and the view update translator can be directly defined from the instance and update mappings between the different intermediate schemas generated during the view integration process. We formally define the conditions for the validation of view update translations in the presence of view interference, and we show that, since schema transformation under our methodology is information and update semantics preserving, we can formally prove that a view update translation defined from the update mappings is a valid translation.

We propose a *decomposable normal form* for schemas that specifies which properties a schema should have so that the problem of "merge incompatibility" captured by certain existence dependency constraints do not occur. We also present an algorithm to transform restricted class of schemas into equivalent schemas which are in decomposable normal form.

We also describe how to design an object-oriented representation for a conceptual schema obtained with our methodology, and how to translate conceptual view definitions into object-oriented view definitions.

ÍNDICE

1. INTRODUÇÃO

O projeto de banco de dados pode ser descrito como a atividade de construir uma *representação* de um dado domínio de aplicação em um dado formalismo. Esta representação ou modelo precisa capturar os objetos que estão envolvidos no domínio de aplicação, e as operações que, de alguma forma, afetam os objetos. O processo de projeto de banco de dados consiste de duas fases (vide Figura 1.1):

(i) A fase de *projeto do esquema conceitual,* na qual o projetista especifica o banco de dados em termos de um modelo semântico. O resultado dessa fase é o *esquema conceitual* do banco de dados.

(ii) A fase de *projeto do esquema lógico,* na qual o esquema conceitual é traduzido em um dos modelos tradicionais. O resultado dessa fase constitui o *esquema lógico* do banco de dados.

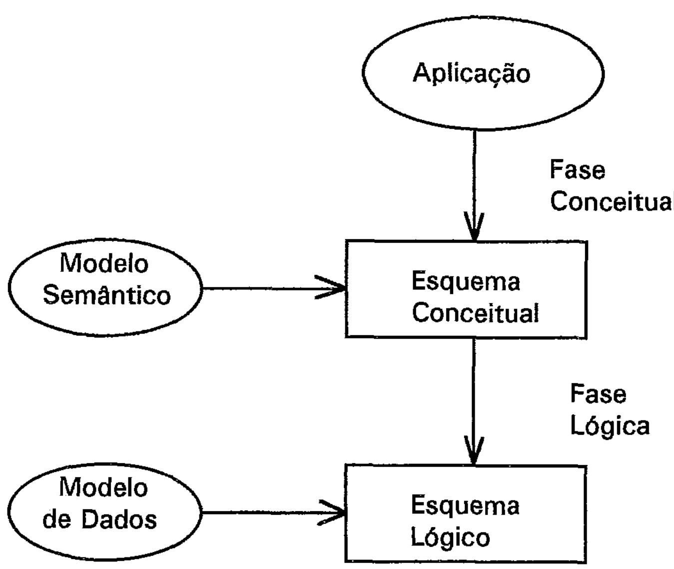


Figura 1.1: Fases do Projeto de Banco de Dados

Durante o processo do projeto do esquema conceitual as visões dos usuários são abstraídas e representadas. Cada grupo de usuários analisa seus requisitos e especifica suas visões dos dados. Em seguida as várias visões dos usuários são integradas em um esquema conceitual global que satisfaz os requisitos de toda a organização (integração das visões). Em nosso enfoque o processo de integração das visões é dividido em três passos:

(1) ***Combinação***. Durante este passo, as várias visões dos usuários são analisadas e comparadas para determinar correspondência entre conceitos, identificar relacionamentos entre as visões e detectar possíveis conflitos. O resultado desse passo é o *esquema combinado* o qual contém todas as visões originais e um conjunto de restrições de integridade que expressam de que maneira os dados estão relacionados nas várias visões.

(2) ***Reestruturação***: Os objetivos do passo de reestruturação de visões são:
(i) compatibilizar as visões de forma que a fusão das partes comuns seja possível, e,
(ii) aumentar a clareza e poder de expressão do esquema, expressando, tanto quanto possível, as restrições de integridade através da estrutura do esquema.

(3) ***Otimização***: O objetivo do passo de otimização é reduzir o tamanho do esquema através da fusão das partes comuns.

Qualquer metodologia para integração de visões deve ser baseada em algum modelo de dados. Um modelo de dados é a base para especificarmos as visões e os relacionamentos semânticos entre os conceitos nas diferentes visões. Adotamos, nesse trabalho, um modelo de dados semântico que é fortemente influenciado pelos modelos SDM [Hammer81], o modelo funcional [Shipman81] e TAXIS [Mylopoulos80]. Nosso modelo suporta formas mais genéricas de restrições de dependências existenciais que são necessárias para dar uma fundamentação teórica sólida à integração de visões. No nosso modelo a declaração de uma restrição pode ser estendida com informações extras para indicar o que deve ser feito quando uma atualização viola essa restrição (semântica de atualização). Essa informação é requerida quando mais de uma alternativa de correção for possível. Nosso modelo, também, oferece uma facilidade para a definição conceitual das visões em termos de um dado esquema conceitual.

Nossa metodologia define um conjunto de primitivas de transformação as quais permitem que a integração de esquemas seja realizada de forma segura e eficiente. Nas primitivas de transformação propostas, o relacionamento entre o esquema original e o esquema transformado é formalmente especificado pelo *mapeamento de instâncias* e pelo *mapeamento de atualizações*. O mapeamento de instâncias especifica de que forma as instâncias do esquema original são mapeadas nas instâncias do esquema transformado e vice-versa. O mapeamento de atualizações especifica de que maneira cada operação de atualização definida sobre o esquema original é transformada numa operação de atualização definida sobre o esquema transformado. Nesse trabalho consideramos que uma tranformação deve preservar a informação e a semântica das atualizações. Uma transformação *preserva a informação* quando o esquema original e o esquema transformado têm capacidades de informação equivalentes [Miller93]. Uma transformação *preserva a semântica das atualizações* quando qualquer atualização especificada no esquema transformado tem o mesmo efeito que o da operação sendo executada pelo usuário diretamente no esquema original. A garantia formal da preservação da semântica das atualizações é uma das inovações do nosso enfoque e é o objeto central dessa tese.

O enfoque que apresentamos pode ser sumarizado como se segue: primeiro definimos o *esquema combinado* que contém todas as visões originais, mais um novo conjunto de restrições que expressam como os dados das diferentes visões estão relacionados. Se necessário, a declaração de uma restrição é estendida com informação extra que especifica a semântica de atualização para suportar essa restrição. Depois, quando transformações de esquemas ocorrem durante os passos de reestruturação e otimização, a semântica de atualização das restrições, no contexto do esquema original, deve ser "mapeada" na semântica de atualização das restrições, no contexto do esquema transformado, de forma que a semântica das atualizações seja preservada. Como o esquema combinado contém todas as visões originais, o mapeador de instâncias e o tradutor de atualizações das visões são diretamente definidos a partir dos mapeamentos de instâncias e de atualizações entre os vários esquemas intermediários gerados durante o proceso de integração. Usando transformações que preservam a informação e a semântica das atualizações, podemos garantir formalmente que os relacionamentos entre as visões e o esquema global, gerados durante o processo de integração, reflete exatamente os relacionamentos entre as visões da forma como foram definidos pelo esquema combinado. Pode-se, também, usar esse enfoque para, incrementalmente, identificar relacionamentos entre visões e integrá-los, com a mesma garantia formal de corretude.

Um problema de integração de visões é que não se pode processar diretamente a fusão de conceitos que sejam os mesmos mas tenham representações diferentes. Esse problema pode ser resolvido através da transformação do esquema, de forma que as visões sejam compatibilizadas e a fusão das classes e atributos comuns se torne possível (reestruturação de esquemas). Nas metodologias para integração de visões existentes [Batini84, Navathe86, Gottard92], a reestruturação de esquemas não é tratada com rigor. Isso, em parte, deve-se ao fato de que o modelo semântico usado por estas metodologias não pode expressar a equivalência de conceitos com representações diferentes. Portanto, essas metodologias não podem formalmente definir as transformações em que estas restrições devem ser preservadas. Nosso trabalho resolve parte dessa limitação com o suporte de formas mais gerais de restrições de dependências existenciais, as quais permitem a definição formal dos tipos mais comuns de transformações que são requeridas durante a reestruturação de esquemas. Nós propomos uma *forma normal decomposta* para esquemas a qual especifica as propriedades que um esquema deve ter para que o problema de "incompatibilidade de fusão" capturado por determinadas restrições de dependências existenciais não ocorra, e apresentamos um algorítimo para transformar determinados tipos de esquemas em esquemas equivalentes que estejam em forma normal decomposta.

A maior parte das metodologias para integração de visões existentes não tem como justificar a corretude das suas transformações. Como observado em [Miller93], como essas metodologias não definem formalmente o tipo de equivalência de esquemas que utilizam, algumas anomalias podem ocorrer devido a transformações de esquemas incorretas. [Miller93] explora o uso da equivalência de capacidade de informação como base para a validação das transformações na integração e tradução de esquemas. No nosso enfoque, como a seqüência de transformações de esquema começa com o esquema combinado, que assumimos conter todas as informações sobre as visões e os relacionamentos ente as visões, exigimos que os esquemas original e transformado tenham a mesma capacidade de informação. Nosso trabalho introduz um novo critério para a validação das tranformações, que é: *a semântica das atualizações no esquema original deve ser preservada pelo esquema transformado.* Isto é necessário para que possamos garantir que são válidas as traduções das atualizações de visões definidas, diretamente, a partir dos mapeamentos de atualizações entre os vários esquemas intermediários gerados durante o proceso de integração.

O problema de definir traduções para atualizações de visões tem sido considerado por muitos pesquisadores, incluindo [Bancilhon81, Dayal82, Furtado85, Medeiros85, Keller86]. O enfoque de definir procedimentos gerais de tradução [Dayal82, Furtado85] somente pode ser aplicado a uma classe restrita de atualizações de visões, porque alguns tipos de atualizações de visões requerem que mais semântica seja fornecida pelo usuáro, para eliminar ambigüidades nas traduções das atualizações em visões [Keller86, Medeiros85]. O enfoque de Keller fornece ao usuário uma enumeração completa de todas as traduções possíveis para as atualizações da visão, e usa a semântica fornecida pelo usuário para escolher um tradutor de atualizações da visão. Keller assume que "o entendimento da interação entre cada visão e o banco de dados nos leva a entender a interação entre as visões". Acreditamos que não é possível relacionar corretamente cada visão com o banco de dados, a menos que se entenda primeiro como as visões originais estão relacionadas entre si. Diferentemente de todos os trabalhos citados acima, este conhecimento nos permite provar formalmente que uma tradução de atualização de uma visão tem o efeito desejado nas outras visões. Em nosso trabalho, o relacionamento entre as visões é explicitamente especificado no esquema combinado, onde a semântica necessária para eliminar ambiguidades é obtida durante o passo de combinação das visões e é explicitamente especificada pela semântica de atualização das restrições do esquema combinado.

Nessa tese nós definimos formalmente as condições para a validação das traduções de atualizações de visões na presença de interferência entre visões e mostramos que, como as transformações de esquemas na nossa metodologia preservam a informação e a semântica das atualizações, podemos provar formalmente que as traduções das atualizações de visões definidas a partir dos mapeamentos de atualizações são válidas.

A tese é dividida em oito capítulos como se segue. No capítulo 2, apresentamos o modelo de dados semântico que será usado para a modelagem e integração das visões. No capítulo 3, descrevemos as características gerais da metodologia proposta e a comparamos com o estado da arte. No capítulo 4, discutimos o passo de reestruturação. No capítulo 5, discutimos o passo de otimização. No capítulo 6, apresentamos um exemplo do processo de integração de visões com a metodologia proposta. No capítulo 7, descrevemos como projetar uma representação orientada a objetos para um esquema conceitual obtido com a nossa metodologia e como se deve traduzir as definições conceituais das visões em definições orientadas a objetos. O capítulo 8 contém conclusões e direções para futuras pesquisas.

2.0 O MODELO SEMÂNTICO

Nesse capítulo apresentamos o modelo semântico utilizado para a modelagem e integração das visões. Todas as atividades do projeto do esquema conceitual são fortemente influenciadas pelo modelo semântico utilizado para expressar o esquema das visões. O sucesso da integração de esquemas depende da compreensão da semântica dos componentes do esquema (por exemplo, atributos, classes de entidades, relacionamentos) e da habilidade de descobrir e representar como os componentes dos diversos esquemas das visões dos usuários estão semanticamente relacionados. Sem a habilidade de descobrir e representar relacionamentos entre os componentes do esquema não poderemos integrá-los eficientemente.

Esse capítulo é organizado como se segue. Na seção 2.1, apresentamos os conceitos preliminares. Na seção 2.2, definimos os vários tipos de restrições de integridade que são suportadas pelo modelo. Na seção 2.3, discutimos o enfoque proposto nesse trabalho, para manutenção das restrições de integridade. Na seção 2.4, apresentamos um mecanismo para definição conceitual de visões.

2.1 CONCEITOS PRELIMINARES

Nós adotamos um modelo semântico que é fortemente influenciado pelos modelos SDM [Hammer81], Funcional [Shipman81] and TAXIS [Mylopoulos80]. Nesse modelo os objetos conceituais são modelados como *entidades*. O modelo suporta a noção de *classe*, que permite o agrupamento de entidades de acordo com suas similaridades. Nesse modelo, um esquema é uma tripla $\boldsymbol{S}=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ onde $\mathcal{C}$ é um conjunto de classes, $\mathcal{A}$ é um conjunto de atributos das classes em $\mathcal{C}$, e $\mathcal{R}$ é um conjunto de restrições de integridade sobre as classes em $\mathcal{C}$ e os atributos em $\mathcal{A}$.

Definições de atributos são relacionamentos funcionais entre entidades em uma classe e outras entidades que são instâncias do contra-domínio do atributo. Atributos podem ser expressos como $\boldsymbol{f}:\boldsymbol{A}\rightarrow\boldsymbol{B}$, onde $\boldsymbol{f}$ é o nome do atributo cujo domínio consiste das entidades na classe $\boldsymbol{A}$ e cujo contra-domínio consiste das entidades na classe $\boldsymbol{B}$. Por exemplo, o atributo

departamento*:*ESTUDANTES*→*DEPARTAMENTOS

relaciona entidades da classe ***ESTUDANTES*** com entidades da classe ***DEPARTAMENTOS***.

Atributos são divididos em duas categorias:

(1) *Atributo de valor.* O contra-domínio de um atributo de valor é definido em uma classe pré-definida pelo sistema (classe "built-in"). Por exemplo, o atributo

CR*:*ESTUDANTES* → *REAL

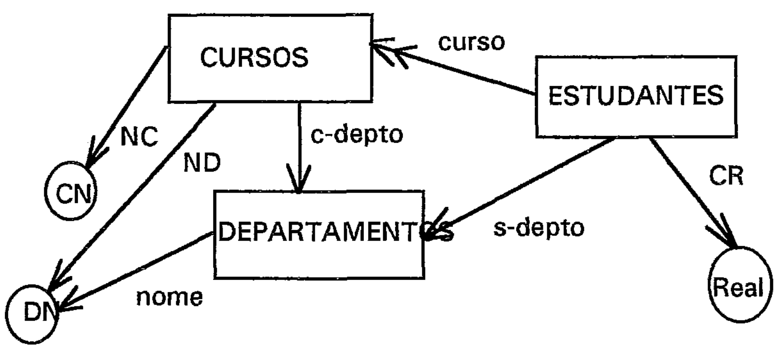
é um atributo que mapeia uma entidade da classe ***ESTUDANTES*** em instâncias da classe ***REAL*** que é uma classe pré-definida pelo sistema.

(2) *Atributo de referência.* O contra-domínio de um atributo de referência é uma classe definida pelo usuário. Por exemplo, o atributo

departamento*:*ESTUDANTES*→*DEPARTAMENTOS

é um atributo de referência.

Também distinguimos entre atributos monovalorados e multivalorados. O valor de um atributo monovalorado tem um único elemento, enquanto que o valor de um atributo multivalorado pode ter mais de um elemento. Na notação gráfica da figura 2.1, retângulos representam classes e círculos representam classes pré-definidas pelo sistema. Setas simples representam atributos monovalorados e setas duplas representam atributos multivalorados. Maiores detalhes da notação gráfica utilizada serão apresentados à medida que novos conceitos forem sendo introduzidos.



<u>Abreviações</u>:**NC**= Número do curso **ND**= Nome do departamento
CR= Coeficiente de Redimento

Figura 2.1: Representação gráfica de classes e atributos

<u>Definição 2.1</u>: Uma *instância* $\mathcal{D}$ de um esquema $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ é uma função definida nos conjuntos $\mathcal{C}$ e $\mathcal{A}$ como se segue:

(i) Para qualquer classe $A \in \mathcal{C}$, $\mathcal{D}$ atribui um conjunto específico de entidades a A (denominado $\mathcal{D}(A)$). Dizemos que $\mathcal{D}(A)$ é a *extensão* de A na instância $\mathcal{D}$.

(ii) Para qualquer atributo $f: A \rightarrow B \in \mathcal{A}$, $\mathcal{J}(f)$ é uma função de $\mathcal{J}(A)$ para $\mathcal{J}(B)$, onde:

- Para qualquer entidade $a \in \mathcal{J}(A)$, $\mathcal{J}(f)$ atribui um conjunto específico de entidades em $\mathcal{J}(B)$(denominado $\mathcal{J}(f)(a)$). Dizemos que $\mathcal{J}(f)(a)$ é o *valor* de $f(a)$ na instância $\mathcal{J}$.
- Para qualquer conjunto de entidades $\{a_1, \ldots, a_n\} \subseteq \mathcal{J}(B)$, temos: $\mathcal{J}(f)(\{a_1, \ldots, a_n\}) = \{\mathcal{J}(f)(a_1), \ldots, \mathcal{J}(f)(a_n)\}$.

<u>Definição 2.2:</u> Sejam $\mathcal{J}_1$ e $\mathcal{J}_2$ instâncias do esquema S. $\mathcal{J}_1 = \mathcal{J}_2$ sss

(i) para qualquer classe C de S, $\mathcal{J}_1(C) = (\mathcal{J}_2)(C)$ e

(ii) para qualquer atributo $f: C \rightarrow A$ de S, para qualquer $e \in \mathcal{J}_1(C)$ $\mathcal{J}_1(f)(e) = \mathcal{J}_2(f)(e)$.

<u>Definição 2.3:</u> Suponha que $S = (\mathcal{C}, \mathcal{A}, \mathcal{R})$ seja um esquema e $\mathcal{J}$ seja uma instância de S. O conjunto de relacionamentos de uma classe $A \in \mathcal{C}$, denominado $rel(A)$, é o menor conjunto tal que:

(i) Se o atributo $f: A \rightarrow B \in \mathcal{A}$ então $f: A \rightarrow B \in rel(A)$

(ii) Se o atributo de referência $f: B \rightarrow A \in \mathcal{A}$ então $f^{-1}: A \rightarrow B \in rel(A)$, onde para qualquer instância $\mathcal{J}$ de S e qualquer $a \in \mathcal{J}(A)$, $\mathcal{J}(f^{-1})(a) = \{b \mid b \in \mathcal{J}(B) \text{ e } a \in \mathcal{J}(f)(b)\}$.

(iii) Se $f: A \rightarrow B \in rel(A)$, $g: B \rightarrow C \in rel(B)$, então $g \circ f: A \rightarrow C \in rel(A)$, onde para qualquer instância $\mathcal{J}$ de S e qualquer $a \in \mathcal{J}(A)$, $\mathcal{J}(g \circ f)(a) = \mathcal{J}(g)(\mathcal{J}(f)(a))$.

Por exemplo, os relacionamentos das classe *CURSOS* (*rel*(*CURSOS*)) no esquema da Figura 2.1 são: *NC*, *ND*, *c-depto*, *c-depto*$\circ$*nome*, *c-depto*$\circ$*s-depto*$^{-1}$, *c-depto*$\circ$*s-depto*$^{-1}\circ$*cursos*, *cursos*$^{-1}$, *cursos*$^{-1}\circ$*s-depto*, *cursos*$^{-1}\circ$*s-depto*$\circ$*c-depto*$^{-1}$.

2.2 RESTRIÇÕES DE INTEGRIDADE

O modelo dá suporte aos seguintes tipos de restrições de integridade: restrição de dependência funcional (DF), restrição de dependência funcional de união (DFU), restrição de chave, restrição de cardinalidade, restrição referencial, restrição referencial total, restrição de equivalência_de_relacionamentos, restrição de dependência existencial (DE), e restrição estrutural. Tipos adicionais de restrições podem ser adicionados se desejado. Nesta seção suponha que $S = (\mathcal{C}, \mathcal{A}, \mathcal{R})$ seja um esquema e que $\mathcal{J}$ seja uma instância de S.

2.2.1 RESTRIÇÃO DE DEPENDÊNCIA FUNCIONAL (DF)

Definição 2.4: A *restrição de dependência funcional* $C([f_1 \ldots f_n] \rightarrow f)$, onde $C \in \mathcal{C}$ e $f_1,\ldots, f_n$, $f \in rel(C)$, é válida em $\mathcal{D}$ sss para qualquer e_1, $e_2 \in C$, se para todos $1 \leq i \leq n$ $\mathcal{D}(f_i)(e_1)) = \mathcal{D}(f_i)(e_2))$, então $\mathcal{D}(f)(e_1)) = \mathcal{D}(f)(e_2))$.

Exemplo 2.1: Considere o esquema da Figura 2.2 as restrições de dependência funcionais sobre a classe ***MATRÍCULAS*** são:

(i) ***MATRÍCULAS*** ([***estudante***, ***curso***] $\rightarrow$ ***professor***)

(ii) ***MATRÍCULAS*** ([***estudante∘nome***, ***curso∘número***] $\rightarrow$ ***professor***)

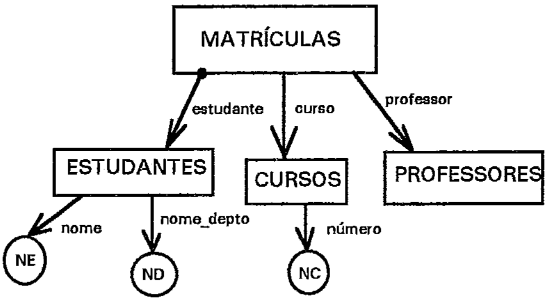


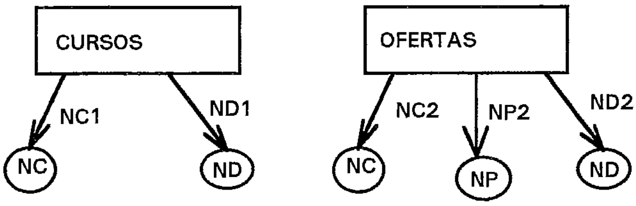
Figura 2.2: Esquema do Exemplo 2.1

É importante notar que a Definicão 2.1 propõe uma forma de restrição de dependência funcional mais geral do que as dependências funcionais tradicionais. Os componentes de dependências funcionais tradicionais só podem ser atributos, enquanto que, na Definição 2.1, os componentes são relacionamentos (i.e., atributos, inversas de atributos ou composição de atributos). A vantagem dessa forma generalizada de dependência funcional é que ela aumenta o poder de expressão do modelo, uma vez que existem restrições de dependência funcional que não podem ser representadas por nenhuma combinação de dependências funcionais simples em que todos os componentes são atributos. Limitando-se os componentes de uma dependência funcional a serem somente atributos diminui-se o poder de expressão.

2.2.2 RESTRICÃO DE DEPENDÊNCIA FUNCIONAL DE UNIÃO (UFD)

Definição 2.5: A *restrição de dependência funcional de união*
$< C_1([f_{i1},\ldots,f_{im}]\rightarrow f_1),\ldots,C_n([f_{n1},\ldots,f_{nm}]\rightarrow f_n) >$, onde $C_i \in \mathcal{C}$, $f_{ik} \in rel(C_i)$, $1 \leq i \leq n$ e $1 \leq k \leq m$, é válida em $\mathcal{D}$ sss $\forall i,j$ em $[1,n]$, $\forall e \in \mathcal{D}(C_i)$ e $\forall e' \in \mathcal{D}(C_j)$, se para todos $1 \leq k \leq m$, $\mathcal{D}(f_{ik})(e) = \mathcal{D}(f_{jk})(e')$, então $\mathcal{D}(f_i)(e) = \mathcal{D}(f_j)(e')$.

Exemplo 2.2: No esquema da figura 2.3, temos a seguinte restrição de dependência funcional de união: $< CURSOS([NC1] \rightarrow ND1),\ OFERTAS([NC2] \rightarrow ND2) >$



Abreviações: NC= Número do curso, NP= Nome do professor, ND= Nome do departamento.

Figura 2.3: Esquema do exemplo 2.2

2.2.3 RESTRIÇÃO DE CHAVE

Definição 2.6: A *restrição de chave* $\{f_1,\ldots, f_n\} \in Chaves(C)$, onde $C \in \mathcal{C}$ e $f_1,\ldots, f_n \in rel\,(C)$ é válida em $\mathcal{B}$ sss para qualquer $e_1,\ e_2 \in C$, se para todos $1 \leq i \leq n$, $\mathcal{B}(f_i)(e_1) = \mathcal{B}(f_i)(e_2))$, então $e_1 = e_2$.

Exemplo 2.3: No esquema da figura 2.2, temos a seguinte restrição de chave:
$\{estudante \circ nome,\ curso \circ número\} \in Chaves(MATRÍCULAS)$.

2.2.4 RESTRIÇÃO DE CARDINALIDADE

Definição 2.7: A *restrição de cardinalidade* $(C\text{-}min(f)=min)$ & $(C\text{-}max(f)=max)$, onde $f \in \mathcal{A}$ e $f \in rel(C)$ e min e max são números inteiros, é válida em $\mathcal{B}$ sss $\forall e \in \mathcal{B}(C)$ $|\mathcal{B}(f)(e)| \leq max$ & $|\mathcal{B}(f)(e)| \geq min$

O modelo distingue entre atributos multivalorados e monovalorados. Um atributo f é monovalorado quando $C\text{-}max(f)=1$, i.e., o valor de um atributo monovalorado é uma única entidade. Um atributo f é multivalorado quando $C\text{-}max(f)>1$, i.e., o valor de um atributo multivalorado é um subconjunto do contra-domínio o qual pode ter mais de um elemento. Atributos monovalorados podem ser definidos como *requeridos* e *não requeridos*. Um atributo f é requerido quando $card\_min(f) \geq 1$, i.e., o valor de um atributo requerido não pode ser nulo. Um atributo f é não requerido quando $card\_min(f)=0$, i.e., o atributo pode ter um valor nulo (não definido). Quando a cardinalidade mínima de um atributo não é especificada assume-se o valor 0 (*min*=0). Quando a cardinalidade máxima de um atributo multivalorado não é especificada assume-se que *max* tem valor ilimitado (*max*= ∞).

2.2.5 RESTRIÇÃO REFERENCIAL

Um atributo de referência $f: A \rightarrow B$ impõe uma dependência existencial (*restrição referencial*) entre as coleções A e B, uma vez que a existência de uma referência para a entidade e é dependente da existência de e em B, como definido a seguir:

Definição 2.8: A *restrição referencial* $f: A \rightarrow B$, onde $f \in \mathcal{A}$ e $A, B \in \mathcal{C}$, é válida em $\mathcal{B}$ sss $\forall a \in \mathcal{B}(A)$ se $b \in \mathcal{B}(f)(a)$ então $b \in \mathcal{B}(B)$.

Exemplo 2.4: No esquema da figura 2.2, temos a seguinte restrição referencial: *professor*: $MATRÍCULAS \rightarrow PROFESSORES$.

2.2.6 RESTRIÇÃO REFERENCIAL TOTAL

Além da restrição referencial, um atributo de referência pode ser restringido como *total*. Dizemos que um atributo de referência $f: A \bullet\!\!\rightarrow B$ é total (restrição referencial total) se todos os elementos de B tiverem que ser referenciados por algum elemento de A, i. e., $card\_min(f^{-1})=1$. Mais formalmente temos:

Definição 2.9: A *restrição referencial total* $f: A \bullet\!\!\rightarrow B$, onde $f \in \mathcal{A}$ e $A, B \in \mathcal{C}$, é válida em $\mathcal{B}$ sss $\forall\ b \in \mathcal{B}(B)$ existe $a \in \mathcal{B}(A)$ tal que $b \in \mathcal{B}(f)(a)$.

Exemplo 2.5: No esquema da figura 2.6, temos a seguinte restrição referencial total: *estudante*: $MATRÍCULAS \bullet\!\!\rightarrow ESTUDANTES$.

2.2.7 RESTRIÇÕES ESTRUTURAIS

As restrições estruturais capturam relacionamentos entre as extensões das classes. O modelo suporta sete tipos diferentes de restrições estruturais que definidos a seguir.

<u>Definição 2.10</u>: A restrição estrutural de *subconjunto* $\boldsymbol{A} \subset \boldsymbol{B}$ (***A*** <u>*está contida em*</u> ***B***, ou ***A*** *é uma subclasse de* ***B***) onde ***A*** e $\boldsymbol{B} \in \mathcal{C}$ é válida em $\mathcal{D}$ sss $\mathcal{D}(\boldsymbol{A}) \subset \mathcal{D}(\boldsymbol{B})$.

<u>Exemplo 2.6:</u> Considere o esquema da figura 2.4. A classe ***ESTUDANTES_PÓS*** está contida na classe ***ESTUDANTES*** ou seja é uma subclasse da classe ***ESTUDANTES*** (***ESTUDANTES_PÓS*** $\subset$ ***ESTUDANTES***). Na nossa notação gráfica, setas sem rótulo representam restrição estrutural de subconjunto.

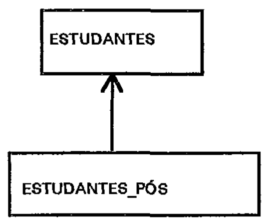


Figura 2.4: Esquema do exemplo 2.6

<u>Definição 2.11</u>: A restrição estrutural de *equivalência* $\boldsymbol{A} \equiv \boldsymbol{B}$ (***A*** e ***B*** <u>*são equivalentes*</u>) é válida em $\mathcal{D}$ sss $\mathcal{D}(\boldsymbol{A}) = \mathcal{D}(\boldsymbol{B})$.

<u>Definição 2.12</u>: A restrição estrutural de *disjunção* $\boldsymbol{A} | \boldsymbol{B}$ (***A*** e ***B*** <u>*são disjuntas*</u>), onde ***A*** e $\boldsymbol{B} \in \mathcal{C}$, é válida em $\mathcal{D}$ sss $\mathcal{D}(\boldsymbol{A}) \cap \mathcal{D}(\boldsymbol{B}) = \emptyset$.

<u>Definição 2.13</u>: A restrição estrutural de *diferença* $\boldsymbol{C} = \boldsymbol{A} - \boldsymbol{B}$ (***C*** <u>*é a diferença*</u> de ***A*** e ***B***), onde ***A***, ***B*** e $\boldsymbol{C} \in \mathcal{C}$, é válida em $\mathcal{D}$ sss $\mathcal{D}(\boldsymbol{C}) = \mathcal{D}(\boldsymbol{A}) - \mathcal{D}(\boldsymbol{B})$.

<u>Definição 2.14</u>: A restrição estrutural de *união* $\boldsymbol{C} = \bigcup_{i=1}^{n} \boldsymbol{C}_i$ (***C*** <u>*é a união*</u> de $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_n$), onde $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_n$ e $\boldsymbol{C} \in \mathcal{C}$, é válida em $\mathcal{D}$ sss $\mathcal{D}(\boldsymbol{C}) = \bigcup_{i=1}^{n} \mathcal{D}(\boldsymbol{C}_i)$.

Definição 2.15: A restrição estrutural de *interseção* $\boldsymbol{C} = \bigcap_{i=1}^{n} C_i$ ($\boldsymbol{C}$ *é a interseção* de $C_1, \ldots, C_n$), onde $C_1, \ldots, C_n$ e $C \in \mathcal{C}$, é válida em $\mathcal{D}$ sss $\mathcal{D}(C) = \bigcap_{i=1}^{n} \mathcal{D}(C_i)$.

Definição 2.16: A restrição estrutural de *generalização* $C = GEN(C_1, \ldots, C_n)$ ($\boldsymbol{C}$ *é a generalização* de $C_1, \ldots, C_n$), onde $C_1, \ldots, C_n$ e $C \in \mathcal{C}$, é válida em $\mathcal{D}$ sss

$$\mathcal{D}(C) = \bigcup_{i=1}^{n} \mathcal{D}(C_i) \quad \text{e} \quad \mathcal{D}(C_i) \cap \mathcal{D}(C_j) = \varnothing \quad \text{para } 1 \leq i,j \leq n \text{ e } i \neq j.$$

Definição 2.17: A restrição estrutural de *seleção* $C = A[\mathcal{P}]$, onde A e $C \in \mathcal{C}$ e $\mathcal{P}$ é um predicado, é válida em $\mathcal{D}$ sss $\mathcal{D}(C) = \{a \mid a \in \mathcal{D}(A)$ e a satisfaz o predicado $\mathcal{P}\}$.

As restrições estruturais particionam as classes em classes de equivalência. Dizemos que as classes $\boldsymbol{A}$ e $\boldsymbol{B}$ estão na mesma classe de equivalência se e somente se existir uma coleção $\boldsymbol{C}$ tal que $\boldsymbol{C}$ esteja relacionada com $\boldsymbol{A}$ e $\boldsymbol{B}$ através de restrições estruturais.

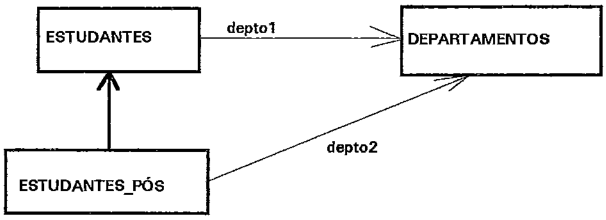


Figura 2.5: Esquema do Exemplo 2.6

Exemplo 2.7: Considere o esquema da figura 2.6. Os relacionamentos ***departamento*** e ***alocação∘orientador*** são sinônimos (***departamento*** ≡ ***alocação∘orientador***).

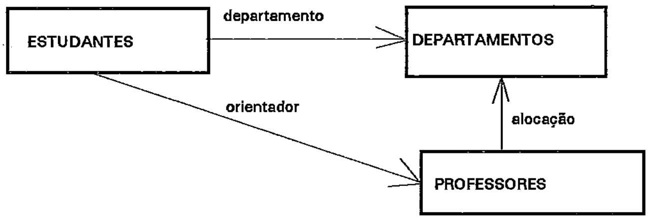


Figura 2.6: Esquema do exemplo 2.7

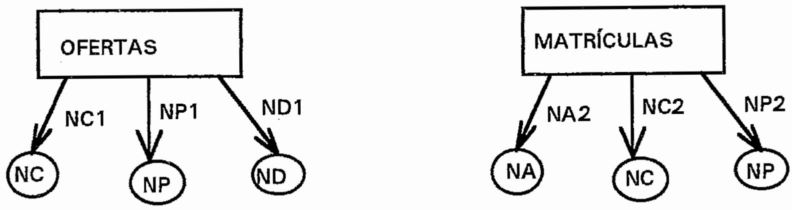
2.2.9 RESTRIÇÕES DE DEPENDÊNCIA EXISTENCIAL (DE)

As restrições de dependência existencial são utilizadas para expressar como a existência de entidades em uma classe depende da existência de entidades em outras classes. Como veremos no Capítulo 4, estas restrições são fundamentais para dar, à reestruturação de esquemas, uma base teórica sólida. A seguir definimos os sete tipos de restrições de dependência existencial que são suportadas pelo modelo.

Para as definições 2.20 a 2.22, suponha que $\boldsymbol{A}$, $\boldsymbol{B} \in \mathcal{C}$, $f_1,..., f_n \in \boldsymbol{rel}(\boldsymbol{A})$ e $g_1,..., g_n \in \boldsymbol{rel}(\boldsymbol{B})$, onde g_i e f_i são monovalorados e têm domínios compatíveis com função de mapeamento de domínio f_i^*, para $1 \leq i \leq n$.

Definição 2.20: A restrição de dependência existencial $A[f_1, ..., f_n] \subset B[g_1, ..., g_n]$ é válida em $\mathcal{D}$ sss para qualquer $a \in \mathcal{D}(A)$, existir $b \in \mathcal{D}(B)$ tal que $\mathcal{D}(f_i^* \circ f_i)(a) = \mathcal{D}(g_i)(b)$ para $1 \leq i \leq n$.

Exemplo 2.8: Considere o esquema da figura 2.7. A restrição de dependência existencial ***MATRÍCULAS***[***NC2***, ***NP2***] ⊂ ***OFERTAS***[***NC1***, ***NP1***] especifica que a existência de uma matrícula ***m*** no curso ***c*** com o professor ***p*** requer a existência de uma oferta ***o*** com curso ***c*** e professor ***p***.



Abreviações: **NC**= Número do Curso, **NP**= Nome do Professor, **ND**= Nome do Departamento, **NA**= Nome do Aluno

Figura 2.7: Esquema do Exemplo 2.8

Definição 2.21: A restrição de dependência existencial $A[f_1,...,f_n]=B[g_1,...,g_n]$ é válida em $\mathcal{D}$ sss $A[f_1, ..., f_n] \subset B[g_1, ..., g_n]$ e $B[g_1, ..., g_n] \subset A[f_1, ..., f_n]$ são válidas em $\mathcal{D}$.

Definição 2.22: A restrição de dependência existencial $A[f_1,...,f_n] \mid B[g_1,...,g_n]$ é válida em $\mathcal{D}$ sss para qualquer $a \in \mathcal{D}(A)$, não existir $b \in \mathcal{D}(B)$ tal que $\mathcal{D}(f_i^* \circ f_i)(a)=\mathcal{D}(g_i)(b)$ para $1 \leq i \leq n$.

Para a definição 2.23, suponha que A, B e $C \in \mathcal{C}$, $h_1,... ,h_n \in rel(C), f_1,..., f_n \in rel(A)$ e $g_1,..., g_n \in rel(B)$, onde h_i, g_i e f_i são monovalorados e têm domínios compatíveis com funções de mapeamento de domínio f_i^* e g_i^* para $1 \leq i \leq n$,

Definição 2.23: A restrição de dependência existencial $C[h_1,... ,h_n]=A[f_1,...f_n] - B[g_1,...,g_n]$ é válida em $\mathcal{D}$ sss ($c \in \mathcal{D}(C)$ sss existe $a \in \mathcal{D}(A)$ tal que $\mathcal{D}(f_i^* \circ f_i)(a) = \mathcal{D}(h_i)(c)$ para $1 \leq i \leq n$, e não existe nenhum $b \in \mathcal{D}(B)$ tal que $\mathcal{D}(g_i^* \circ g_i)(b) = \mathcal{D}(h_i)(c)$, para $1 \leq i \leq n$).

Para as definições 2.24 e 2.25, suponha que $C_1,... , C_m$, e $C \in \mathcal{C}$, $g_1,..., g_n \in rel(C)$, and $f_{i1},...f_{in} \in rel(C_i)$ para $1 \leq i \leq n$, onde g_i e f_{ik} são monovalorados e têm domínios compatíveis com função de mapeamento de domínio f_{ik}^*, para $1 \leq i \leq m$ e $1 \leq k \leq n$.

Definição 2.24: A restrição de dependência existencial $C[g_1,...,g_n]= \bigcup_{i=1}^{m} C_i[f_{i1},...f_{in}]$ é válida em $\mathcal{D}$ sss ($c \in \mathcal{D}(C)$ sss existe $c' \in \mathcal{D}(C_i)$, $i \in [1,m]$, tal que $\mathcal{D}(g_k)(c) = \mathcal{D}(f_{ik}^* \circ f_{ik})(c')$ para $1 \leq k \leq n$).

Definição 2.25: A restrição de dependência existencial $C[g_1, ... ,g_n]= \bigcap_{i=1}^{m} C_i[f_{i1},...f_{in}]$ é válida em $\mathcal{D}$ sss ($c \in \mathcal{D}(C)$ sss existe $c_1,... , c_m$ tal que $c_i \in \mathcal{D}(C_i)$ e $\mathcal{D}(g_k)(c) = \mathcal{D}(f_{ik}^* \circ f_{ik})(c_i)$ para $1 \leq i \leq m$ e $1 \leq k \leq n$).

2.3 MANUTENÇÃO DAS RESTRIÇÕES DE INTEGRIDADE

As restrições de integridade impõem restrições nos estados admissíveis do banco de dados (instâncias). Uma instância $\mathcal{I}$ do esquema S=($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$) é *consistente*, ou é *uma instância válida*, se e somente se para qualquer restrição de integridade $\psi \in \mathcal{R}$ ψ é válida em $\mathcal{I}$. O conjunto de todas as instâncias válidas do esquema S é *instâncias*(S)={$\mathcal{I}$ | $\mathcal{I}$ é uma instância válida de S}. Dado um conjunto de restrições de integridade Σ e uma restrição de integridade ψ sobre S, dizemos que Σ *implica logicamente* ψ se e somente se, para qualquer instância $\mathcal{I}$ de S, se todas as restrições de Σ são válidas em $\mathcal{I}$, então ψ também é válida em $\mathcal{I}$. Um conjunto de restrições $\Sigma \subset \mathcal{R}$ é redundante se qualquer instância satisfazendo todas as restrições em $\mathcal{R}-\Sigma$, também satisfaz todas as restrições Σ. Dizemos que um conjunto de restrições Σ é uma cobertura mínima se não existe nenhum Y subconjunto de Σ tal que $Y \neq \emptyset$ e Y é redundante.

Qualquer operação que modifique o estado do banco de dados deve preservar a consistência, i.e., deve mapear uma instância consistente do banco de dados em outra instância do banco de dados que também seja consistente. Quando uma atualização viola uma restrição, a atualização pode ser rejeitada. Entretanto, em alguns casos, pode ser que outras atualizações sejam requeridas de forma que a consistência seja mantida. Em nosso modelo, para restrições de integridade para as quais existem várias alternativas de correção, o projetista pode estender a declaração das restrições de integridade com informação extra para indicar qual opção deve ser utilizada (semântica de atualização); caso contrário, a opção "default" será assumida. Para especificar a semântica de atualização das restrições de integridade usamos a seguinte notação:

SI(A, ψ): A semântica para inserções na classe A com relação a restrição ψ
SD(A,ψ): A semântica para remoções na classe A com relação a restrição ψ
SM(f, ψ): A semântica para modificações no valor do relacionamento f com relação a restrição ψ.

As alternativas de semântica de atualização para manter a restrição referencial $f{:}A \rightarrow B$ são definidas na Figura 2.8; se desejado pode-se adicionar novas opções. No Apêndice A, definimos as alternativas de semântica de atualização para manter as restrições definidas na Seção 2.2. As semânticas de atualizações, se não forem corretamente especificadas, podem causar problemas de atualização [Markowitz91]. Para evitar esses problemas, deve-se assegurar que as semânticas de atualizações não são conflitantes e não causam "deadlocks". Nessa tese, nós assumimos que as semânticas de atualizações sejam bem-definidas no sentido de que o processamento de uma atualização irá sempre terminar e terá a propriedade de Church-Rosser (i.e., a ordem em que as operações de manutenção de uma restrição são executadas é irrelevante).

Operação de atualização	**Alternativas de semântica de atualização para manter** $\psi = f{:}A \rightarrow B$
Remoção em B	Suponha u="remova b de B ". Seja $\mathcal{B}(A^*)=\{a \mid a \in \mathcal{B}(A)$ and $\mathcal{B}(f)(a)=b\}$. As alternativas de semântica de atualização para u com relação a ψ são: (a) $SD(B, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(A^*)=\emptyset$ então EC="continue". ("EC" indica "**efeito colateral**") (ii) senão, EC="rollback". (b) $SD(B, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*)=\emptyset$ então EC="continue". (ii) se $\mathcal{B}(A^*)=\{a_1, ..., a_n\}\neq\emptyset$ então $EC=\{u_k \mid u_k$="remova a_k em A", para $1 \leq k \leq n\}\}$. (c) $SD(B, \psi)$ = ***nulifica***, determina que: (i) se $\mathcal{B}(A^*)=\emptyset$ então EC="continue". (ii) se $\mathcal{B}(A^*)=\{a_1, ..., a_n\}$ então $EC=\{u_k \mid u_k$="modifique a_k with (f=*nulo*) ", para $1 \leq k \leq n\}\}$.
Adição em A	Suponha u="adicione a a A". Seja $b = \mathcal{B}(f)(a)$. As alternativas de semântica de atualização para u com relação a ψ são: (a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se b=**nulo** or $b \in \mathcal{B}(B)$ então EC="continue" (ii) senão, EC="rollback". (b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se b=**nulo** or $b \in \mathcal{B}(B)$ então EC="continue" (ii) senão, EC={ "adicione b a B "}.
Modificação de f	Suponha u="modifique a com ($f = b$)". As alternativas de semântica de atualização para u com relação a ψ são: (a) $SM(f, \psi)$ = ***rejeita***, determina que: (i) se b=**nulo** or $b \in \mathcal{B}(B)$ então EC="continue". (ii) senão, EC="rollback". (b) $SM(f, \psi)$ = ***propaga***, determina que: (i) se b=**nulo** or $b \in \mathcal{B}(B)$ então EC="continue". (ii) senão, EC={"adicione b a B "}.

Figura 2.8: Alternativas de semântica de atualização para manter $f{:}A \rightarrow B$

A seguir, definimos formalmente o nosso enfoque para manutenção de restrições de integridade. No resto dessa seção, considere $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ um esquema, onde $u \in$ ***atualizações***(S) e $\mathcal{B} \in$ ***instâncias***(S).

Definição 2.26: Uma atualização u é uma função u: $instâncias(S) \rightarrow instâncias(S)$ definida como se segue:

$$\forall \mathcal{D} \in instâncias(S),\ u(\mathcal{D}) = co\text{-}efeito^*(u,\ efeito(u, \mathcal{D})).$$

onde as funções *co-efeito** e *efeito* são definidas a seguir.

As operações de atualização básicas em uma instância $\mathcal{D}$ são:

1) A operação ADICIONE, para adicionar uma entidade a uma classe. Uma operação adicione tem uma das seguintes formas:

a) "adicione e a C ", onde C é uma classe e e é uma entidade de $\mathcal{D}$, ou

b) "adicione $e(f_1=v_1, ..., f_n=v_n)$ a C", onde C é uma classe e $e(f_1=v_1, ..., f_n=v_n)$ é uma especificação_de_entidade, i.e., para $1 \leq j \leq n$, f_j é um atributo de C ou um atributo de uma superclasse de C, e v_j é uma entidade em $\mathcal{D}$, ou uma especificação_de_entidade.

2) A operação REMOVA, para remover uma entidade de uma classe. A operação remova tem a forma "remova e de C" onde C é uma classe e e é uma entidade de $\mathcal{D}(C)$.

3) A operação MODIFIQUE para modificar o valor de um atributo de uma entidade. Uma operação MODIFIQUE tem a sequinte forma "modifique e com $(f=v)$" onde e é uma enitidade em $\mathcal{D}$, $f \in atributos(a, \mathcal{D})$[1] e v é uma entidade em $\mathcal{D}$, ou uma especificação_de_entidade.

Assumimos que a aplicação de uma operação de atualização u em $\mathcal{D}$ é precedida pela definição de uma nova instância $\mathcal{D}'$ onde cada especificação de entidade em u é substituída por uma entidade em $\mathcal{D}'$. Assim sendo, nas definições seguintes só consideramos adicões da forma "adicione e em C" onde e é uma entidade e modificações da forma "modifique e com $(f=v)$" onde v é uma entidade.

Definição 2.27: $efeito(u, \mathcal{D}) = \mathcal{D}'$, onde $\mathcal{D}'$ é uma instância de S (possivelmente não válida) definida como se segue:

(i) Se u = "remova a de A" então $\mathcal{D}'$ é idêntica a $\mathcal{D}$ exceto que:

$$\mathcal{D}'(A) = \mathcal{D}(A) - \{a\}$$

(ii) Se u = "adiciona a em A" então $\mathcal{D}'$ é idêntica a $\mathcal{D}$ exceto que:

$$\mathcal{D}'(A) = \mathcal{D}(A) \cup \{a\},$$

$$atributos(a, \mathcal{D}') = atributos(a, \mathcal{D}) \cup atributos(A)$$

Para cada $g \in atributos(A)$,

(i) se existe $f \in atributos(a, \mathcal{D})$ tal que $g \equiv f$ então $\mathcal{D}'(g)(a) = \mathcal{D}(f)(a)$.

(ii) senão, $\mathcal{D}'(g)(a) = nulo$.

[1] $atributos(a, \mathcal{D})$ é o conjunto de atributos definidos para a entidade a em $\mathcal{D}$.

(iii) Se $\boldsymbol{u}$ = "modifique $\boldsymbol{a}$ com $(f=v)$", então $\mathscr{D}'$ é idêntica a $\mathscr{D}$ exceto que:
$\mathscr{D}'(f)(a) = v$.

Definição 2.28: Seja ψ uma restrição em $\mathcal{C}$. ψ é relevante para $\boldsymbol{u}$ sss existir $\mathscr{D} \in$ *instâncias*(S) tal que *efeito*$(\boldsymbol{u}, \mathscr{D})$ não seja uma instância válida de $\boldsymbol{S}$.

Definição 2.29: *relevantes*$(\boldsymbol{u})=\{\ \psi \mid \psi$ é relevante para $\boldsymbol{u}\ \}$.

Definição 2.30: *co-efeito**$(\boldsymbol{u}, \mathscr{D}) = \mathscr{D}'$ onde $\mathscr{D}'$ é uma instância de $\boldsymbol{S}$ definida como se segue:

(i) se *relevantes*$(u)=\varnothing$ então $\mathscr{D}' = \mathscr{D}$.

(ii) se *relevantes*$(u)=\{\psi_1, \dots, \psi_n\}$ então $\mathscr{D}'$ = *co-efeito*$(\boldsymbol{u}, \mathscr{D}_n, \psi_n)$, onde $\mathscr{D}_1 = \mathscr{D}$, e $\mathscr{D}_i$ = *co-efeito*$(\boldsymbol{u}, \mathscr{D}_{i-1}, \psi_{i-1})$, para $1 < i \leq n$.

Definição 2.31: Seja ψ uma restrição em $\mathcal{C}$ onde ψ é relevante para $\boldsymbol{u}$. Seja $\boldsymbol{EC}$ o "efeito colateral" requerido pela semântica de atualização de $\boldsymbol{u}(\mathscr{D})$ com relação a ψ. *co-efeito*$(\boldsymbol{u}, \mathscr{D}, \psi) = \mathscr{D}'$ onde $\mathscr{D}'$ é uma instância de $\boldsymbol{S}$ definida como se segue:

(i) Se $\boldsymbol{EC}$ ="rollback" então aborta a presente transação.

(ii) Se $\boldsymbol{EC}$ ="continue" então $\mathscr{D}' = \mathscr{D}$.

(iii) Se $\boldsymbol{EC} = \{\boldsymbol{u}_1, \dots, \boldsymbol{u}_n\}$, onde $\boldsymbol{u}_1, \dots, \boldsymbol{u}_n$ são operações de atualização, então $\mathscr{D}' = \boldsymbol{u}_n \circ \dots \circ \boldsymbol{u}_1(\mathscr{D})$.

O Exemplo apresentado a seguir ilustra o enfoque descrito acima.

Exemplo 2.9: Seja $\boldsymbol{S}$ o esquema da figura 2.9 e $\boldsymbol{u}$= "delete $\boldsymbol{o}$ em ***OFERTAS***" uma atualização de $\boldsymbol{S}$. Suponha que $\mathscr{D}$ é uma instância de $\boldsymbol{S}$ tal que $\mathscr{D}(oferta^{-1})(\boldsymbol{o})=\{\boldsymbol{m}_1, \dots, \boldsymbol{m}_n\}$. $\boldsymbol{u}(\mathscr{D})$ é processado com se segue:

- Cálculo de $\boldsymbol{u}(\mathscr{D})$

Seja $\mathscr{D}^1$= *efeito*$(\boldsymbol{u}, \mathscr{D})$. $\mathscr{D}^1$ é idêntica a $\mathscr{D}$ exceto que:

(2.1) $\mathscr{D}^1(\boldsymbol{OFERTAS})=\mathscr{D}(\boldsymbol{OFERTAS}) - \{\boldsymbol{o}\}$

Como *relevantes*$(\boldsymbol{u})=\{\psi\}$, nós temos:

*co-efeito**$(\boldsymbol{u}, \mathscr{D}^1)$ = *co-efeito*$(\boldsymbol{u}, \mathscr{D}^1, \psi)$.

Como $\boldsymbol{EC}=\{\boldsymbol{u}_1$="remova $\boldsymbol{m}_1$ de ***MATRÍCULAS***", ..., $\boldsymbol{u}_n$="remova $\boldsymbol{m}_n$ de ***MATRÍCULAS***"$\}$ é o "efeito colateral" requerido pela semântica de atualização de $\boldsymbol{u}$ com respeito a ψ (*SD*(***OFERTAS***, ψ) = ***propaga***), nós temos:

co-efeito$(\boldsymbol{u}, \mathscr{D}^1, \psi) = \boldsymbol{u}_n \circ \dots \circ \boldsymbol{u}_1(\mathscr{D}^1)$

- Cálculo de $u_k(\mathcal{D}^k)$ onde $\mathcal{D}^k = u_{k-1}(\mathcal{D}^{k-1})$, para $2<k\leq n$.

Seja $\mathcal{D}^{k+1}= \mathit{efeito}(u_k, \mathcal{D}^k)$. $\mathcal{D}^{k+1}$ é identica a $\mathcal{D}^k$ exceto que:

(2.2) $\mathcal{D}^{k+1}(\mathit{MATRÍCULAS})=\mathcal{D}^k(\mathit{MATRÍCULAS}) - \{m_k\}$

Como $\mathit{relevantes}(u_k)=\varnothing$, nós temos:

$$\mathit{co\text{-}efeito}(u_k, \mathcal{D}^{k+1}) = \mathcal{D}^{k+1}.$$

Portanto,

$$u_k(\mathcal{D}^k) = \mathit{co\text{-}efeito}^*(u_k, \mathit{efeito}(u_k, \mathcal{D}^k)) = \mathcal{D}^{k+1}.$$

(Continuando o cálculo de $u(\mathcal{D})$)

$$\mathit{co\text{-}efeito}^*(u, \mathcal{D}^1) = u_n \circ \ldots \circ u_1 (\mathcal{D}^1) = \mathcal{D}^{n+1}.$$

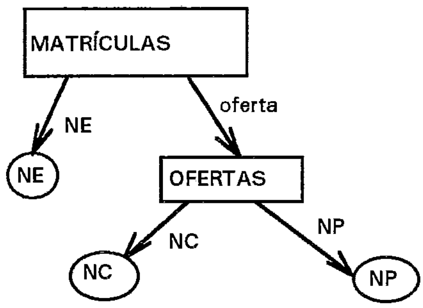
Portanto,

$$u(\mathcal{D}) = \mathit{co\text{-}efeito}(u, \mathit{efeito}(u, \mathcal{D})) = \mathcal{D}^{n+1}.$$

De (2.1) e (2.2) nós temos que $\mathcal{D}^{n+1}$ é idêntica a $\mathcal{D}$ exceto que:

$\mathcal{D}^{n+1}(\mathit{OFERTAS})=\mathcal{D}(\mathit{OFERTAS}) - \{o\}$

$\mathcal{D}^{n+1}(\mathit{MATRÍCULAS})=\mathcal{D}(\mathit{MATRÍCULAS}) - \{m_1, ..., m_n\}$.



A semântica de atualização da restrição referencial ψ = *ofertas*: *MATRÍCULAS* →*OFERTAS* é:

SD(*OFERTAS*, ψ)= ***propaga***

SI(*MATRÍCULAS*, ψ)= ***rejeita***

SM(*ofertas*, ψ)= ***rejeita***

<u>Abreviações</u>:**NC**= Número do Curso, **NP**= Nome do Professor, **ND**= Nome do Departamento, **NE**= Nome do Estudante

Figura 2.9: Esquema para o exemplo 2.9

2.3 DEFINIÇÃO DE VISÕES

Uma definição de visão sobre o esquema S é uma tripla (S_V, σ_V, τ_V), onde $S_V=(\mathcal{C}_V, \mathcal{A}_V, \mathcal{R}_V)$ é um esquema, σ_V é um mapeamento de instâncias de S em instâncias de S_V, e τ_V especifica traduções corretas das operações de atualização sobre S_V para operações de atualização sobre S. Dizemos que σ_V é o *mapeador de instâncias da visão* e τ_V é o *tradutor de atualizações da visão.*

Nessa seção, seja (S_V, σ_V, τ_V) uma definição de visão sobre o esquema $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$. A extensão de uma classe C de S_V é definida a partir de uma ou mais classes de S, chamadas de classes de base de C. Uma operação de atualização sobre uma classe de visão deve ser traduzida para operações de atualização sobre suas classes de base. Baseado no tipo de relacionamento entre uma classe de visão e suas classes de base, definimos cinco tipos diferentes de classes de visão:

Definição 2.32: A classe C de S_V é uma *classe de projeção* sss $\forall \mathcal{S} \in$ **instâncias**(S) $\sigma_V(\mathcal{S})(C)=\mathcal{S}(C')$ onde $C' \in \mathcal{C}$.

Definição 2.33: A classe C de S_V é uma *classe de seleção* sss $\forall \mathcal{S} \in$ **instâncias**(S) $\sigma_V(\mathcal{S})(C)=\{\, e \mid e \in \mathcal{S}(C')$ e e satisfaz $\mathcal{P}\}$onde $C' \in \mathcal{C}$, e $\mathcal{P}$ é um predicado.

Definição 2.34: A classe C de S_V é uma *classe de diferença* sss $\forall \mathcal{S} \in$ **instâncias**(S) $\sigma_V(\mathcal{S})(C) = \mathcal{S}(A) - \mathcal{S}(B)$, onde A e $B \in \mathcal{C}$.

Definição 2.35: A classe C de S_V é uma *classe de união* sss $\forall \mathcal{S} \in$ **instâncias**(S)

$$\sigma_V(\mathcal{S})(C)= \bigcup_{i=1}^{m} \mathcal{S}(C_i) \text{ onde } C_i \in \mathcal{C} \text{ para } 1 \leq i \leq n.$$

Definição 2.36: A classe C de S_V é uma *classe de interseção* sss $\forall \mathcal{S} \in$ **instâncias**(S)

$$\sigma_V(\mathcal{S})(C)= \bigcap_{i=1}^{m} \mathcal{S}(C_i), \text{ onde } C_i \in \mathcal{C} \text{ para } 1 \leq i \leq n.$$

O valor de um atributo f em S_V é definido a partir do valor de um ou mais atributos em S, chamados de atributos de base de f. Uma operação de atualização sobre um atributo de visão deve ser traduzida em operações de atualizações sobre seus atributos de base.

Existem dois tipos de atributos de visão:

(i) *atributo de projeção*: O valor de um atributo de projeção é diretamente definido por meio do valor de um atributo em ***S***, e

(ii) *atributo derivado*: O valor de um atributo derivado é definido por meio da composição de dois ou mais atributos em ***S***.

Dizemos que uma classe de visão é uma *classe de junção* se ela tem um ou mais atributos derivados.

3. CARACTERÍSTICAS GERAIS DA METODOLOGIA

Nesse capítulo, discutimos as características gerais da metodologia. Na Seção 3.1 nós discutimos os três passos do processo de integração de visões. Na Seção 3.2 definimos a noção de transformações que preservam a informação e a semântica das atualizações. Na Seção 3.3 discutimos como os mapeadores de instâncias das visões e os tradutores de atualizações das visões são gerados durante o processo de integração das visões. Na Seção 3.4 comparamos nossa metodologia com outras metodogias para integração de visões existentes.

3.1 O PROCESSO DE INTEGRAÇÃO DE VISÕES

Nós propomos dividir o processo de integração de visão em três passos: Combinação, Reestruturação e Otimização.

• ***Combinação:*** O passo de combinação consiste em analisar e comparar os vários esquemas dos usuários para determinar correspondências entre conceitos, descobrir propriedades inter-esquemas e detectar possíveis conflitos. No modelo semântico proposto, as propriedades inter-esquemas são representadas através das restrições estruturais, restrições de equivalência_de_relacionamentos, restrições de dependências funcionais de união e restrições de dependências existenciais. Como discutimos na Seção 3.4, o modelo resolve limitações que outros modelos semânticos têm para representar relacionamentos entre conceitos similares com formas estruturais diferentes e entre conceitos que, embora não sendo o mesmo, estão de alguma forma relacionados.

Durante a combinação das visões o projetista segue os seguintes passos:

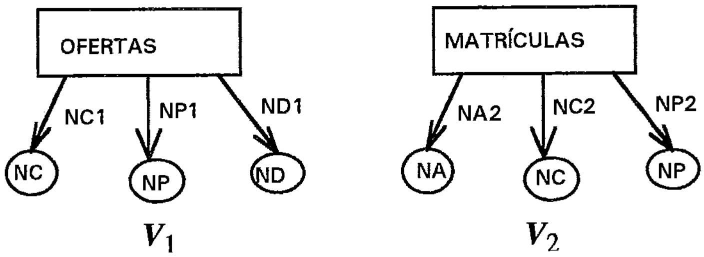
1. Determinação de restrições estruturais.
2. Determinação de restrições de equivalência_de_relacionamentos.
3. Determinação de restrições de dependências existenciais.
4. Determinação de restrições de dependências funcionais de união.

O uso de um modelo de dados com grande poder de abstração não é suficiente para tornar fácil o processo de comparação de visões em aplicações de grande porte. Embora o modelo semântico utilizado permita ao projetista representar a semântica das diversas formas de propriedades inter-esquemas, ele não pode evitar situações onde tais relacionamentos não tenham sido detectados. Muitos pesquisadores, entre estes [Navathe86, Sousa86, Gottard92], têm investigado o uso de ferramentas interativas para ajudar a identificar relacionamentos entre visões, inconsistências e ambigüidades, que podem ser adaptadas para o nosso modelo.

• ***Reestruturação***: Os objetivos do passo de reestruturação são:
(i) Compatibilizar as visões de forma que a integração das partes comuns seja possível, e
(ii) Aumentar a clareza e a expressividade do esquema, expressando, tanto quanto possível, as restrições de integridade através da estrutura do esquema.
Nossa metodologia define um conjunto de primitivas de transformação as quais permitem que a reestruturação das visões seja realizada de forma segura (preserva a informação e a semântica das atualizações) e algorítimica. Essas primitivas de reestruturação são discutidas no Capítulo 4.

• ***Otimização***: O objetivo da otimização é modificar o esquema para reduzir redundâncias e o o tamanho do esquema. Redundâncias podem ser eliminadas através da fusão das classes e atributos. Nossa metodologia define um conjunto de primitivas de transformação que permitem que a otimização do esquema conceitual seja realizada de forma segura e eficiente. Estas primitivas de reestruturação são discutidas no Capítulo 5.

Uma característica importante desta metodologia é que ela dá suporte tanto para o uso do enfoque "top down" quanto para o uso "bottom up" durante a integração das visões. O enfoque *bottom up* é apropriado quando cada grupo de usuários analisa seus requisitos e independentemente especifica sua visão dos dados. As várias visões são então comparadas para identificar similaridades e conflitos (combinação das visões). Durante a fase de combinação das visões, as similaridades entre visões são capturados através das restrições estruturais, das restrições de equivalência_de_relacionamentos, das restrições de dependências funcionais de união e das restrições de dependências existenciais. À medida que restrições inter-visões são identificadas, o projetista tenta identificar possíveis maneiras de reestruturar o esquema de forma que as partes comuns possam ser integradas. Por exemplo, suponha que as visões *V*1 e *V*2 na Figura 3.1 foram independentemente especificadas. Combinando as visões *V*1 e *V*2, o projetista identifica a restrição de dependência existencial ***MATRÍCULAS***[***NC2***, ***NP2***] $\subset$ ***OFERTAS***[***NC1***, ***NP1***], a qual especifica que a existência de uma matrícula ***m*** no curso ***c*** com o professor ***p*** requer a existência de uma oferta ***o*** com curso ***c*** e professor ***p***. A visão *V*2 pode ser reestruturada como na Figura 3.2 o que torna possível a otimização do esquema através da fusão das classes ***OFERTAS*** e ***OFERTAS***$_2$, obtendo-se assim o esquema da Figura 3.3.



<u>Abreviações</u>: **NC**= Número do Curso, **NP**= Nome do Professor, **ND**= Nome do Departamento, **NA**= Nome do Aluno

Figura 3.1: As Visões V_1 e V_2

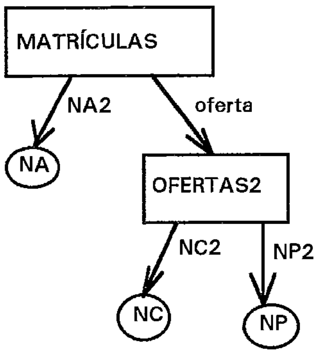


Figura 3.2: Visão V_2 reestruturada

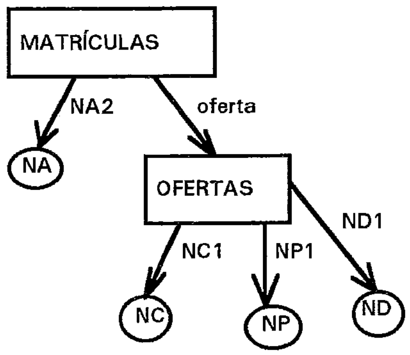


Figura 3.3: Visões V_1 e V_2 integradas

O enfoque <u>*top down*</u> é apropriado quando o projetista identifica similariadades entre as visões antes da suas especificações, i.e., a combinação ocorre juntamente com a modelagem da visão. Por exemplo, o projetista ao especificar os atributos ***NC2*** e ***NP2*** para a classe ***MATRÍCULAS***, já os identifica como atributos da classe **OFERTAS** obtendo, assim, diretamente a estrutura da Figura 3.3. O uso do enfoque top down requer que os relacionamentos entre as visões sejam antecipadamente abstraídos pelo projetista. Isso permite que ele possa definir uma estrutura que capture diretamente a semântica do relacionamento, de forma que reestruturação não seja necessária. Entretanto, o uso do enfoque top-down nem sempre é apropriado. É comum ocorrer a situação em que um relacionamento entre visões só é identificado em um estado mais avançado do projeto(enfoque bottom up). Por isso a importância do uso de um modelo semântico que permita ao projetista intercalar os dois enfoques durante o processo de integração de visões, dependendo de quando os relacionamentos entre visões são identificadas. Na Seção 3.4 discutimos os problemas que as metodologias baseadas no modelo ER e em outros modelos semânticos apresentam para dar suporte ao enfoque bottom up.

3.2 TRANSFORMAÇÕES DE ESQUEMAS

A metodologia proposta define um conjunto de primitivas de transformação que permitem com que transformações de esquemas sejam realizadas de forma segura e eficiente. Uma primitiva de transformação é uma 5-tupla *(**pre**, **pos**, **ϕ**, **σ**, **τ**)* onde:

- **PRE** = ($\mathbf{PRE_s}$, $\mathbf{PRE_{rest}}$) especifica as pré-condições necessárias para a aplicação da transformação. $\mathbf{PRE_s}$ especifica as condições em termos da estrutura original e $\mathbf{PRE_{rest}}$ especifica restrições adicionais que não são representadas em $\mathbf{PRE_s}$.

- **POS** = ($\mathbf{POS_s}$, $\mathbf{POS_{rest}}$) especifica o esquema transformado resultante da aplicação da transformação. $\mathbf{POS_s}$ especifica a estrutura do esquema transformado e $\mathbf{POS_{rest}}$ especifica as transformações de restrições.

- **ϕ** especifica como instâncias do esquema original são mapeadas em instâncias do esquema transformado.

- **σ** especifica como instâncias do esquema transformado são mapeadas em instâncias do esquema original (mapeamentos de instâncias).

- **τ** especifica como as operações de atualizações definidas sobre o esquema original são traduzidas em operações de atualizações sobre o esquema transformado (mapeamentos de atualizações)

As primitivas de transformação que apoiam os passos de reestruturação e otimização são discutidas nos capítulos 4 e 5. Uma transformação de esquema deve preservar a "semântica" do esquema original. No nosso enfoque, uma transformação deve *preservar a informação e a semântica das atualizações.* Dizemos que uma transformação *preserva a informação* quando o esquema original e o esquema transformado têm capacidades de informação equivalentes [Miller93], ou seja não existe perda de informação durante a transformação.

Definição 3.1: Uma transformação **T** = (***pre***, ***pos***, **ϕ**, **σ**, **τ**) *preserva a informação* se dado um esquema S_1 tal que S_1 satisfaça a pré-condição ***pre*** e $\mathbf{T}(S_1)=S_2$, as seguintes condições forem satisfeitas (vide Figura 3.4):

(i) $\forall \mathcal{B}_1 \in$ *instâncias*(S_1) $\phi(\mathcal{B}_1) \in$ *instâncias*(S_2) & $\mathcal{B}_1 = \sigma(\phi(\mathcal{B}_1))$

(ii) $\forall \mathcal{D}_2 \in$ *instâncias*(S_2) $\sigma(\mathcal{D}_2) \in$ *instâncias*(S_1).

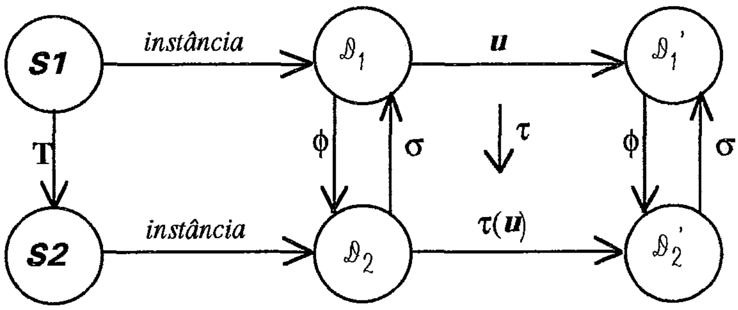


Figura 3.4: *T* preserva a informação e a semântica das atualizações

Dizemos que uma tradução de atualização é uma *tradução correta* (ou válida) se a tradução da atualização especificada no esquema transformado tem o mesmo efeito que o da atualização sendo executada diretamente no esquema original.

<u>Definição 3.2</u>: Seja **T**=(***pre***, ***pos***, ϕ, σ, τ) uma transformação que preserva a informação. Suponha que o esquema S_1 satisfaça a pré-condição ***pre*** e que S_2=**T**(S_1). Seja $u \in$ *atualizações*(S_1) e $u_\tau \in$ *atualizações*(S_2). u_τ é uma *tradução correta de **u*** sss a seguinte condição for satisfeita (vide figura 3.4):

$$\forall \mathcal{D}_1 \in \textit{instâncias}(S_1)\ \ u(\mathcal{D}_1) = \sigma(u_\tau(\phi(\mathcal{D}_1)))$$

Dizemos que uma transformação *preserva a semântica das atualizações* se todos os mapeamentos de atualizações forem traduções corretas.

<u>Definição 3.3:</u> Seja **T**=(***pre***, ***pos***, ϕ, σ, τ) uma trasformação que preserva a informação. **T** *preserva a semântica das atualizações* se dado um esquema S_1 tal que S_1 satisfaça a pré-condição ***pre*** e **T**$(S_1)=S_2$, a seguinte condição for satisfeita (veja figura 3.4):

$$\forall u \in \textit{atualizações}(S_1)\ \tau(u) \text{ é uma tradução correta de } u.$$

Na especificação de uma transformação *T*=(***pre***, ***pos***, ϕ, σ, τ), as pré-condições ***pre*** e ***pos*** consideram somente a parte do esquema que é relevante para ***T***, e especificam somente a semântica de atualizações das restrições para as quais mais de uma alternativa de correção existe. Quando aplicando a tranformação ***T*** no esquema S_1 onde **T**$(S_1)=S_2$, assumimos que as classes e atributos em S_1 que não aparecem em ***pre*** são diretamente transformados em classes e atributos de S_2 como definido a seguir.

Definição 3.4: A classe $\boldsymbol{C}$ de S_1 é diretamente transformada na classe $\boldsymbol{C'}$ de S_2 ($T(C)=C'$) sss:

(i) $\boldsymbol{C}$ e $\boldsymbol{C'}$ têm o mesmo nome,

(ii) $\forall \mathcal{D}_1 \in$ *instâncias*(S_1) $\mathcal{D}_1(C) = \phi(\mathcal{D}_1)(C')$,

(iii) $\forall \mathcal{D}_2 \in$ *instâncias*(S_2) $\mathcal{D}_2(C') = \sigma(\mathcal{D}_2)(C)$,

(iv) τ("adicione $\boldsymbol{e}$ a $\boldsymbol{C}$") = "adicione $\boldsymbol{e}$ a $\boldsymbol{C'}$ " e

(v) τ("remova $\boldsymbol{e}$ de $\boldsymbol{C}$") = "remova $\boldsymbol{e}$ de $\boldsymbol{C'}$ ".

Definição 3.5: O atributo $f{:}A \rightarrow B$ de S_1 é diretamente transformado no atributo $f'{:}A' \rightarrow B'$ de S_2 ($T(f)=f'$) sss:

(i) f e f' têm o mesmo nome,

(ii) $\forall \mathcal{D}_1 \in$ *instâncias*(S_1) $\forall e \in \mathcal{D}_1(A)$ $\mathcal{D}_1(f)(e) = \phi(\mathcal{D}_1)(f')(e)$,

(iii) $\forall \mathcal{D}_2 \in$ *instâncias*(S_2) $\forall e \in \mathcal{D}_2(A')$ $\mathcal{D}_2(f')(e) = \sigma(\mathcal{D}_2)(f)(e)$,

(iv) τ("modifique $\boldsymbol{e}$ com (f=v)") = "modifique $\boldsymbol{e}$ com (f'= v)".

Usamos também a definição acima para especificar implicitamente tranformações de classes e atributos que aparecem em ***pre*** mas são diretamente transformadas em classes e atributos de S_2, assumindo que:

(i) Para qualquer classe $\boldsymbol{C}$ de S_1, se $\boldsymbol{C}$ aparecer em ***pre*** e existir a classe $\boldsymbol{C'}$ em S_2 com o mesmo nome que $\boldsymbol{C}$, então $T(C)= C'$ ($\boldsymbol{C}$ é diretamente transformada em $\boldsymbol{C'}$).

(ii) Para qualquer atributo f de S_1, se f aparecer em ***pre*** e existir um atributo f' em S_2 com o mesmo nome que f, então $T(f)=f'$ (f é diretamente transformada em f').

Assim sendo, a especificação de ϕ, σ e τ para classes e atributos que sejam diretamente transformados em classes e atributos de S_2 podem ser induzidas.

3.3 DEFINIÇÃO CONCEITUAL DAS VISÕES

Após o témino do processo de integração, todas as visões devem ser definidas em termos do esquema conceitual obtido. Como definido na Seção 2.3, a definição conceitual da visão *V* é uma tripla (S_V, σ_V, τ_V) onde S_V é o esquema da visão, σ_V é o mapeador de instâncias da visão e τ_V é o tradutor de atualizações da visão. No nosso enfoque, como o esquema combinado contém todas as visões originais, os mapeadores de instâncias e os tradutores de atualizações das visões são diretamente definidos pelos mapeamentos de instâncias e mapeamentos de atualizações entre os vários esquemas intermediários gerados durante o processo de integração.

Definição 3.6: Suponha que o esquema global *GS* seja obtido a partir do esquema combinado *CS* através da aplicação das transformações T_1, ..., T_m com mapeamentos de instâncias σ_1, ...,σ_m e mapeamentos de atualizações τ_1, ...,τ_m. Suponha que T_1, ..., T_m preservem a informação e a semântica das atualizações. Seja S_V=($\mathcal{C}_V$, $\mathcal{A}_V$, $\mathcal{R}_V$) o esquema da visão *V* em *CS*. Seja $\mathcal{D} \in$ *instâncias*(*GS*).

(i) O mapeador de instâncias de *V* (σ_V) é definido por:

- $\forall\ C \in \mathcal{C}_V,\ \sigma_V(\mathcal{D})(C) = \sigma_1 \circ ... \circ \sigma_m(\mathcal{D})(C)$
- $\forall f{:}C \rightarrow D \in \mathcal{A}_V,\ \ \forall e \in \sigma_V(\mathcal{D})(C),\ \ \sigma_V(\mathcal{D})(f)(e) = \sigma_1 \circ ... \circ \sigma_m(\mathcal{D})(f)(e)$

(ii) O tradutor de atualizações de *V* (τ_V) é definido por:

- $\forall u \in$ *atualizações*(*V*), $\tau_V(u) = \tau_m \circ ... \circ \tau_1(u)$

A seguir, definimos as condições para a validação de traduções de atualizações de visões e mostramos que (ii) define traduções válidas. O capítulo 6 contém um exemplo concreto da prova de que uma tradução é válida.

Para definirmos formalmente as condições para a validação de traduções de atualizações de visões utilizamos a especificação explícita dos relacionamentos entre as visões como especificado pelo esquema combinado. Dizemos que uma tradução de atualização de visão *é uma tradução válida* se e somente se a atualização especificada no esquema global tem o mesmo efeito que o da operação sendo executada pelo usuário diretamente no esquema combinado. Isso pode ser formalizado como se segue.

Definição 3.7: Suponha que o esquema global ***GS*** seja definido para o esquema combinado ***CS*** com visões $\{V_1, ..., V_n\}$. Seja $\sigma_{V1}, ..., \sigma_{Vn}$ os mapeadores de instâncias para $V_1, ..., V_n$. A tradução da atualização ***u*** da visão V_i, $1 \leq i \leq n$, para uma atualização ***T*(*u*)** do esquema global é uma tradução válida se e somente se a seguinte condição for satisfeita (veja figura 3.5):

$$\forall \mathcal{D} \in \textbf{\textit{instâncias}}(GS) \;\; u(\{\sigma_{V1}(\mathcal{D}), ..., \sigma_{Vn}(\mathcal{D})\}) = \{\sigma_{V1}(T(u)(\mathcal{D})), ..., \sigma_{Vn}(T(u)(\mathcal{D}))\}$$

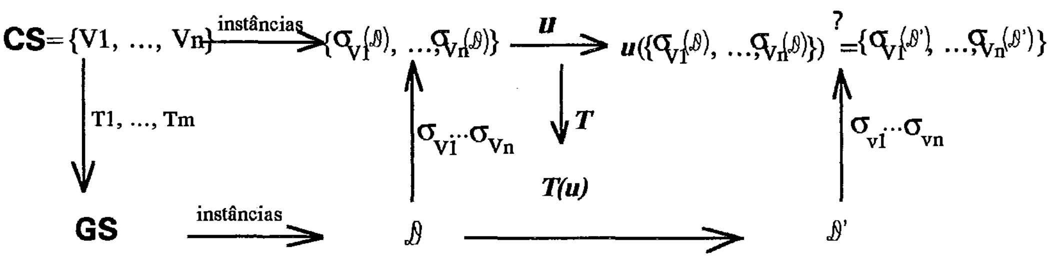


Figura 3.5: ***T*(*u*)** **é uma tradução válida de *u***

De acordo com a definição 3.7, na especificação de uma tradução da atualização ***u*** da visão V_i para uma atualização ***T*(*u*)** do esquema global, não é suficiente provar que ***T*(*u*)** tem o efeito desejável na visão V_i. Precisamos provar também que ***T*(*u*)** tem o efeito desejável nas outras visões como definido pelo esquema combinado. Esta definição difere dos trabalhos citados anteriormente [Bancilhon81, Dayal82, Furtado85, Medeiros85, Keller86] em seu tratamento do problema de interferência entre visões. Para provarmos formalmente que uma tradução de atualização de visão tem o efeito desejado nas outras visões, precisamos conhecer como os dados nas diferentes visões estão relacionados. Este conhecimento é explicitamente especificado no esquema combinado obtido com o nosso enfoque, onde a semântica necessária para eliminar ambigüidades é obtida durante o passo de combinação das visões e é explicitamente especificada através da semântica de atualização das restrições do esquema combinado. Como as transformações, na metodologia proposta, preservam a informação e a semântica das atualizações, da proposição 3.1 a seguir, podemos concluir que a Definição 3.6 define traduções válidas. Isso mostra também que, no nosso enfoque, os relacionamentos entre as visões e o esquema global refletem exatamente os relacionamentos entre as visões como definidos pelo esquema combinado.

Proposição 3.1: Suponha que o esquema global ***GS*** seja obtido a partir do esquema combinado ***CS*** através da aplicação das transformações $T_1, ..., T_m$ com mapeamentos de instâncias $\sigma_1, ..., \sigma_m$ e mapeamentos de atualizações $\tau_1, ..., \tau_m$.

Suponha que T_1, ..., T_m preservam a informação e a semântica das atualizações. Seja σ_{v1},..., σ_{vn} os mapeadores de instâncias para as visões V_1, ..., V_n de CS. Dada uma atualização u da visão V_i, $1 \le i \le n$, a tradução para uma atualização $T(u) = \tau_m \circ \ldots \circ \tau_1(u)$ do esquema global é uma tradução válida.♦

<u>Prova:</u>

Seja $S_1 = T_1(CS)$, $S_i = T_i(CS)$, para $1 \le i \le m$.

Seja $u \in atualizações(CS)$ e $\mathcal{B}_1 \in instâncias(S_1)$.

Como T_1 preserva a informação e a semântica das atualizações, das definiçõe 3.2 and 3.3 temos:

$$(1)\ u(\sigma_1(\mathcal{B}_1)) = \sigma_1(\tau_1(u)(\mathcal{B}_1))$$

Seja $\mathcal{B}_{k-1} \in instâncias(S_{k-1})$ tal que $\sigma_2 \circ \ldots \circ \sigma_{k-1}(\mathcal{B}_{k-1}) = \mathcal{B}_1$, $2 < k < m$.

Suponha que:

$$(2)\ \tau_1(u)(\mathcal{B}_1) = \sigma_2 \circ \ldots \circ \sigma_{k-1}(\tau_{k-1} \circ \ldots \circ \tau_1(u)(\mathcal{B}_{k-1})). \text{ (hipótese de indução)}$$

Seja $\mathcal{B}_k \in instâncias(S_k)$ tal que $\sigma_k(\mathcal{B}_k) = \mathcal{B}_{k-1}$.

Como T_k preserva a informação e a semântica das atualizações, das definições 3.2 and 3.3 temos:

$$\forall u' \in atualizações(S_{k-1})\ \ u'(\mathcal{B}_{k-1}) = \sigma_k(\tau_k(u')(\mathcal{B}_k))$$

Suponha que $u' = \tau_{k-1} \circ \ldots \circ \tau_1(u)$. Portanto,

$$(3)\ \tau_{k-1} \circ \ldots \circ \tau_1(u)(\mathcal{B}_{k-1}) = \sigma_k(\tau_k\ (\tau_{k-1} \circ \ldots \circ \tau_1(u))\ (\mathcal{B}_k)\ ).$$

Substituindo (3) em (2) temos:

$$\tau_1(u)(\mathcal{B}_1) = \sigma_2 \circ \ldots \circ \sigma_{k-1}(\sigma_k\ (\tau_k \circ \ldots \circ \tau_1(u)\ (\mathcal{B}_k)).$$

Logo

$$(4)\ \tau_1(u)(\mathcal{B}_1) = \sigma_2 \circ \ldots \circ \sigma_k\ (\tau_k \circ \ldots \circ \tau_1(u)\ (\mathcal{B}_k)), \text{ for } 2 < k \le m$$

Seja $\mathcal{B}_m \in instâncias(S_m)$ ($S_m = SG$) tal que $\sigma_2 \circ \ldots \circ \sigma_m(\mathcal{B}_m) = \mathcal{B}_1$. De (4) temos:

$$(5)\ \tau_1(u)(\mathcal{B}_1) = \sigma_2 \circ \ldots \circ \sigma_m\ (\tau_m \circ \ldots \circ \tau_1(u)\ (\mathcal{B}_m)).$$

Substituindo (5) em (1) temos:

$$u(\sigma_1 \circ \ldots \circ \sigma_m(\mathcal{B}_m)) = \sigma_1 \circ \ldots \circ \sigma_m\ (\tau_m \circ \ldots \circ \tau_1(u))(\mathcal{B}_m)\ )$$

Mas $\sigma_1 \circ \ldots \circ \sigma_m(\mathcal{B}_m) = \{\sigma_{v1}(\mathcal{B}_m), \ldots, \sigma_{vn}(\mathcal{B}_m)\}$ e

$$\sigma_1 \circ \ldots \circ \sigma_m(\tau_m \circ \ldots \circ \tau_1(u))(\mathcal{B}_m))) = \{\sigma_{v1}(\tau_m \circ \ldots \circ \tau_1(u)(\mathcal{B}_m)), \ldots, \sigma_{vn}(\tau_m \circ \ldots \circ \tau_1(u)(\mathcal{B}_m))\}$$

Logo,

$$u(\{\sigma_{v1}(\mathcal{B}_m), \ldots, \sigma_{vn}(\mathcal{B}_m)\}) = \{\sigma_{v1}(\tau_m \circ \ldots \circ \tau_1(u)(\mathcal{B}_m)), \ldots, \sigma_{vn}(\tau_m \circ \ldots \circ \tau_1(u)(\mathcal{B}_m))\}$$

Portanto, $T(u) = \tau_m \circ \ldots \circ \tau_1(u)$ é uma tradução válida.

3.4 COMPARAÇÃO COM OUTRAS METODOLOGIAS PARA INTEGRAÇÃO DE VISÕES

Nessa seção comparamos a metodologia de integração de visão que propomos, com outras metodologias existentes. Classificando estas metodologias de acordo com o modelo de dados no qual elas se baseiam temos:
(1) Metodologias baseadas no modelo de entidade e relacionamentos (E-R).
(2) Metodologias baseadas no enfoque orientado a objetos

3.4.1 METODOLOGIAS BASEADAS NO MODELO E-R

A metodologia que propomos, assim como outras baseadas no enfoque orientado a objetos, usam atributos para representar relacionamentos entre entidades. Esse enfoque tem quatro vantagens imediatas sobre o uso de construtores explícitos como é feito no modelo de Entidades e Relacionamentos [Chen79]:

1) Atributos são tratados mais uniformemente. Eles podem se referir tanto a valores como a outras entidades, não sendo necessário remodelagem quando atributos viram relacionamentos.

2) Esse enfoque generaliza o conceito de entidade e relacionamento não distinguindo entidades que representam objetos conceituais e entidades que representam relacionamento entre objetos. Assim sendo, relacionamentos podem ser definidos entre relacionamentos sem que seja necessário diferenciar entre integração de entidade e integração de relacionamento [Navathe86] durante o processo de integração de visão.

3) Nosso enfoque permite representar múltiplas perspectivas do mesmo fato enquanto se mantém o controle da equivalência entre as diferentes perspectivas. Alguns usuários podem escolher, por exemplo, representar *a matrícula de um estudante em um curso* como uma entidade tendo suas próprias propriedades, enquanto que para outros pode ser mais natural representar esse relacionamento como um atributo de estudante. No modelo de E-R não podemos representar explicitamente o relacionamento semântico entre conceitos similares mas com estruturas diferentes.

(4) No nosso enfoque, as restrições de dependências existenciais permitem representar relacionamentos entre conceitos que não são exatamente os mesmos mas

estão de alguma forma relacionados. O modelo E_R não oferece nenhum suporte para a integração de conceitos que não são "diretamente integráveis" [Navathe86].

Consideremos um exemplo simples (figura 3.6) tirado de [Navathe86]. Na visão ***V1***, ***R1*** descreve o relacionamento de um fornecedor fornecendo uma peça, e ***R2*** descreve o relacionamento de um comprador comprando uma peça. Na visão ***V2***, ***R3*** descreve o fato de que um comprador compra uma certa peça de um certo fornecedor. ***R1***, ***R2*** e ***R3*** estão relacionados, uma vez que a existência de relacionamento entre um comprador ***c***, uma peça ***p***, e um fornecedor ***f*** em ***R3*** requer a existência de um relacionamento entre ***f*** e ***p*** em ***R1*** e de um relacionamento entre ***c*** e ***p*** em ***R2***. Como ***R1***, ***R2*** e ***R3*** não são diretamente integráveis, essa restrição não pode ser capturada pelo esquema global usando as técnicas tradicionais de integração de visões [Navathe86]. A representação das visões ***V1*** e ***V2*** com o nosso modelo semântico é esquematizada na figura 3.7. As similaridades entre as visões ***V1*** e ***V2*** são expressas através das restrições de dependências existenciais:

(i) ***R3***[***vendedor3***, ***peça3***] $\subset$ ***R1***[***vendedor1***, ***peça1***], e

(ii)***R3***[***comprador3***, ***peça3***] $\subset$ ***R2***[***comprador2***, ***peça2***]

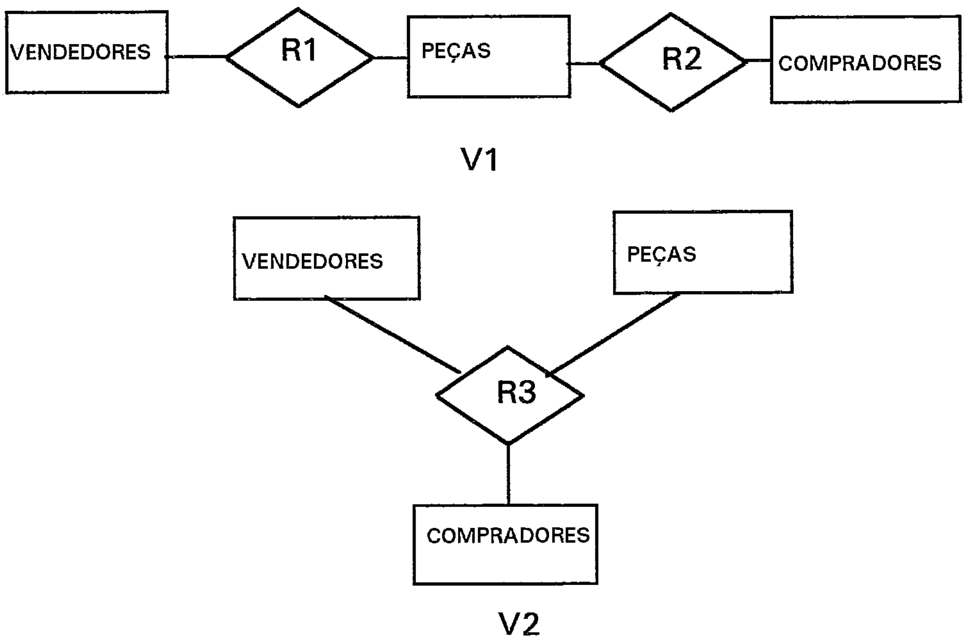


Figura 3.6: Visões V1 e V2 representadas no modelo E-R

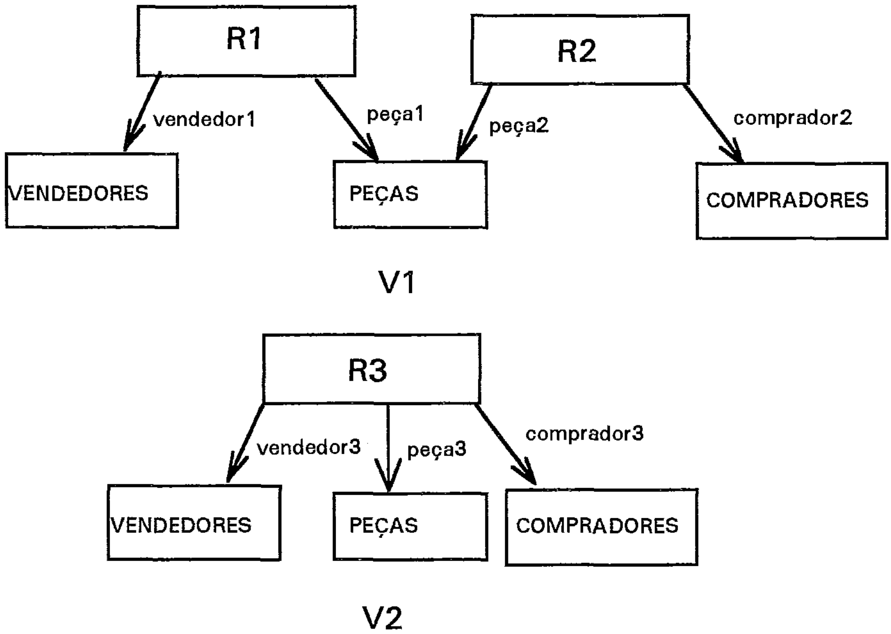


Figura 3.7: Visões V1 e V2 representadas com o nosso modelo semântico.

3.4.2 METODOLOGIAS BASEADAS NO ENFOQUE ORIENTADO A OBJETOS

Nessa seção, discutimos primeiramente o mecanismo de classificação como proposto pelos modelos semânticos [Hammer81, Shipman81, Mylopoulos80] sucessores do modelo E-R e suas limitações para dar suporte ao processo de modelagem e de integração de visões. Em seguida discutimos como o nosso enfoque supera estas limitações.

No mecanismo de classificação usual, uma classe é especificada pela definição do tipo de seus elementos (intensão da classe), e a extensão da classe é definida implicitamente como sendo o conjunto de objetos que são instâncias da classe em um determinado momento. Uma classe pode ser declarada como sendo uma subclasse de outra. Nesse caso, todas as instâncias da subclasse são também instâncias de suas superclasses e uma classe herda todos os atributos definidos por suas superclasses. Este mecanismo de classificação tem as seguintes limitações para dar suporte ao processo de modelagem e integração de visões:

(1) Um objeto só pode ser membro de duas classes diferentes se ele for uma instância de uma classe que seja subclasse de ambas as classes. Por exemplo, para que um objeto seja membro das classe ***ESTUDANTES_POS*** e

ESTUDANTES_INTEGRAL, uma classe *ESTUDANTE_INTEGRAL_POS*, subclasse das classes *ESTUDANTES_POS* e *ESTUDANTES_INTEGRAL*, precisa ser definida.

(2) As metodologias de integração de visões atuais sugerem que um atributo comum de duas classes com entidades comuns ("overlapping") deve migrar para uma superclasse comum. Por exemplo, o atributo **orientador** comum para as classes *ESTUDANTES_INTEGRAL* e *ESTUDANTES_POS* deve ser definido na classe *ESTUDANTES*. Essa solução pode introduzir tanto problema dos valores nulos e dificuldades em modificar o esquema. Além disso, tal solução não é aplicável quando as visões requerem restrições diferentes para o atributo, como acontece, por exemplo, no caso de uma visão requerer que estudantes de pós graduação tenham orientadores que sejam membros do comitê acadêmico, enquanto que, em outras visões, estudantes não estão sujeitos a esta restrição. Em tal situação, a única alternativa seria definir uma classe *ESTUDANTES_INTEGRAL_POS* que fosse subclasse das classes *ESTUDANTES_INTEGRAL* e *ESTUDANTES_POS*. Isso causa o mesmo problema que em (1), proliferação de subclasses.

(3) Esse mecanismo de classificação não permite que uma definição de uma subclasse limite herança a somente os atributos que o usuário da subclasse tem direito a acessar (visibilidade limitada para visões).

(4) Uma subclasse não pode alterar a especificação do atributo herdado. Por exemplo, a subclasse *ESTUDANTES_POS* não pode especificar a restricão "avançado" para o atributo **cursa**[1] herdado da classe *ESTUDANTES*. (TAXIS [Myl80] permite que o contra-domínio de um atributo seja restringido quando uma classe for especializada, mas não permite que outras formas de restrições sejam adicionadas).

(5) A modificação do esquema para adicionar uma nova classe e/ou uma nova restrição pode requerer restruturação da hierarquia de classes. Nesse caso, as classes devem ser remodeladas para refletir a nova estrutura.

(6) Para aplicar com sucesso o conceito de especialização de classes, o projetista precisa conhecer à priori a organização da hierarquia de classes. No entanto, o conhecimento dessa organização é de fato resultado do projeto do esquema conceitual. Assim, sem essa informação, o uso do enfoque top down para projeto do esquema conceitual nem sempre é apropriado ou possível.

---

[1] o atributo ***cursa*** define o conjunto de cursos que o estudante está cursando.

No nosso enfoque uma classe não define um tipo de entidade como no mecanismo de classe descrito acima, mas sim os requisitos mínimos que as entidades membros da classe devem satisfazer. Como uma entidade pode ser membro de mais de uma classe, seu tipo deve satisfazer os requisitos de todas as classes para as quais ela é membro. Podemos definir o ***tipo conceitual*** de uma entidade como a união das restrições de todas as classes das quais ela é membro. O nosso enfoque melhora as outras metodologias baseadas no enfoque orientado a objetos nos seguintes pontos:

1) O modelo evita a proliferação de definições de tipos no nível conceitual (problemas (1) e (2)).

2) Se novas classes e/ou restrições são adicionadas, nenhuma remodelação do esquema conceitual é requerida (Problema 5). Nosso modelo torna os estágios subsequentes do processo de projeto mais um refinamento dos estágios anteriores do que uma reformulação.

3) Como discutido na Seção 3.1, a nossa metodologia suporta tanto o enfoque "top down" quanto o enfoque "bottom up" durante o projeto do esquema conceitual (Problema 6). Isso dá flexibilidade para o projetista direcionar o projeto de acordo com as informações disponíveis em cada instante.

4) Nossa metodologia suporta a integração de atributos sinônimos com restrições diferentes, o que não é tratado pelas metodologias de integração de visões já existentes (problema 2).

5) Como discutido na seção 2.2.8, o nosso modelo permite representar formas mais gerais de equivalência de relacionamentos envolvendo composição e inversa de atributos.

6) Nossa metodologia fornece, também, um conjunto de primitivas de transformação que apoiam o processo de integração de visões e garantem que as transformações preservam a validade dos esquemas originais. As metodologias de integração existentes não têm como justificar a correteza de suas transformações. Conforme discutido em [Miller93], como essas metodologias não definem formalmente que tipo de esquema de equivalência é requerido, algumas anomalias podem ocorrer devido a transformações incorretas do esquema.

7) O problema de reestruturação de esquemas não é tratado com rigor pelas metodologias de integração existentes. Isso deve-se ao fato de que os modelos usados por essas metodologias não podem expressar a equivalência de conceitos similares com representações diferentes. Assim sendo, tais metodologias não podem definir formalmente transformações em que essas restrições precisam ser preservadas. Nosso trabalho resolve parte dessa limitação com o uso de formas mais gerais de restrições de dependências existenciais que permitem a definição formal dos tipos mais comuns de tranformações que são requeridas durante a reestruturação de esquemas.

8) No nosso enfoque, além de alcançarmos a integração das visões no esquema global obtido, nós também obtemos os mapeadores de instâncias e os tradutores de atualizações das visões. Como discutimos na Seção 3.3 os mapeadores de instâncias e os tradutores de atualizações das visões são diretamente definidos, a partir dos mapeamentos de instâncias e de atualizações entre os vários esquemas intermediários gerados durante o proceso de integração.

9) Nosso enfoque introduz um novo critério para a validação de transformações, que é: *a semântica das atualizações do esquema original deve ser preservada no esquema transformado*. Como discutido na Seção 3.3, isso é necessário para que sejam válidas as traduções das atualizações em visões, definidas por meio dos mapeamentos das atualizações entre os diferentes esquemas intermediários gerados durante o processo de integração de visões.

4. O PASSO DE REESTRUTURAÇÃO

Um problema de integração de visões é que algumas formas de redundâncias capturadas por certas dependências existenciais (DEs) não podem ser diretamente integradas. Esse problema pode ser resolvido através de transformações de esquemas que compatibilizem as visões, possibilitando a fusão de classes e atributos. Um outro problema que também pode ser resolvido através de reestruturação de esquemas é o problema de redundâncias e anomalias causadas por certas DFs. O processo de normalização que soluciona esse problema no âmbito de banco de dados relacionais, foi definido originalmente em [Codd 72]. Nesse capítulo, nós definimos formalmente uma *forma normal decomposta* para esquemas a qual especifica as propriedades que um esquema deve ter para que o problema de incompatibilidade de fusão capturado por certas DEs e o problema de redundâncias e anomalias causadas por certas DFs não ocorram. Na Seção 4.1.3, nós apresentamos um algorítimo para transformação de um esquema em outro equivalente em forma normal decomposta.

Dividimos a reestruturação de esquemas em três passos:

1) Normalização;

2) Transformação de restrições de dependências existenciais em restrições estruturais e restrições de equivalência_de_relacionamentos, e

3) Transformação de restrições funcionais de união em restrições de equivalência_de_relacionamentos,

que discutimos nas próximas seções.

4.1 NORMALIZAÇÃO

Nessa seção apresentamos uma forma normal decomposta para esquemas e um algorítimo para transformação de um esquema em outro, equivalente, em forma normal decomposta.

4.1.1 A FORMA NORMAL DECOMPOSTA

Nessa seção consideramos que $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ é um esquema e, sem perda de generalidade, assumimos que $\mathcal{R}$ seja uma cobertura mínima.

<u>Definição 4.1</u>: Seja C uma classe de S e $\{f_1, ..., f_n\} \subseteq rel(C)$. $\{f_1, ..., f_n\} \in S\_chaves(C)$ sss para qualquer $g \in rel(C)$, $\mathcal{R} \models C([f_1, ..., f_n] \rightarrow g)$.

Definição 4.2: Seja C uma classe de S e $\{f_1, ...f_n\} \subseteq rel(C)$. $\{f_1, ...f_n\} \in chaves(C)$ sss:

(i) $\{f_1, ...f_n\} \in S\_chaves(C)$, e

(ii) para qualquer $\{fk_1, ...fk_m\} \subset \{f_1, ...f_n\}$, $\{fk_1, ...fk_m\} \notin S\_chaves(C)$.

Informalmente, dizemos que o esquema S está em forma normal decomposta se as seguintes condições forem satisfeitas: as únicas restrições funcionais (DFs) em $\mathcal{R}$ forem aquelas que implicam em restrições de chaves; as únicas restrições existenciais (DEs) em $\mathcal{R}$ forem aquelas que implicam em restrições estruturais; e as únicas DFUs em $\mathcal{R}$ forem aquelas que implicam em restrições de equivalência_de_relacionamentos. Isso pode ser formalizado como se segue:

Definição 4.3: Dizemos que $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ está em *forma normal decomposta* sss:

(i) Se $C([f_1, ...f_n] \rightarrow g)$ é uma DF em $\mathcal{R}$ então $\{f_1, ...f_n\} \in S\_chaves(C)$.

(ii) Se $C([f_1, ...f_n] \rightarrow g)$ é um membro de uma DFU em $\mathcal{R}$, então $\{f_1, ...f_n\} \in S\_chaves(C)$.

(iii) Se ψ é uma DE em $\mathcal{R}$, tal que $A[f_1, ...f_n]$ e $B[g_1, ...g_n]$ ocorrem em ψ, então

a) $\{f_1, ..., f_n\} \in S\_chaves(A)$,

b) $\{fk_1, ...fk_m\} \in S\_chaves(A)$ onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$ sss $\{gk_1, ...gk_m\} \in S\_chaves(B)$,

c) Se $\{fk_1, ...fk_m\} \in S\_chaves(A)$ onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$ então $\mathcal{R} \models \langle A([fk_1, ...fk_m] \rightarrow f_j), B([gk_1, ...gk_m] \rightarrow g_j) \rangle$, para $1 \leq j \leq n$. ♦

É importante notar que a condição (i) da Definição 4.3 é equivalente à condição imposta pela forma normal BCNF. Podemos mostrar que é sempre possível fazer com que o esquema satisfaça a condição (i) sem que ele perca a habilidade de preservar restrições. Como mostramos na Seção 4.2, quando um esquema está em forma normal decomposta, todas as restrições existenciais podem ser transformadas em restrições estruturais e de equivalência_de_relacionamentos de forma que a fusão é possível. Nesse trabalho, nos concentraremos em uma classe restrita de esquemas para as quais é sempre possível encontrar esquemas equivalentes que estão em forma normal decomposta.

Definição 4.4: Dizemos que o esquema $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ é um esquema restrito sss

(i) Se $C([f_1, ...f_n] \rightarrow g)$ é uma DF em $\mathcal{R}$ ou um membro de uma DFU em $\mathcal{R}$, então $\{f_1, ...f_n, g\} \subseteq atributos(C)$,

(ii) Se ψ é uma DE em $\mathcal{R}$, tal que $C[f_1, ...f_n]$ ocorre em ψ, então $\{f_1, ..., f_n\} \subseteq atributos(C)$,

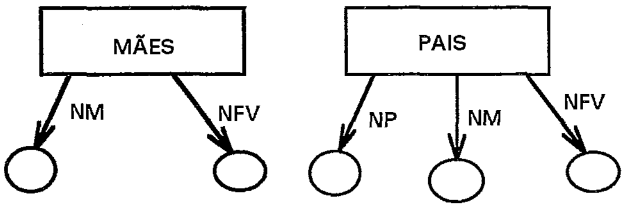
(iii) Para qualquer DE ψ em $\mathcal{R}$ tal que $A[f_1, ..., f_n]$ e $B[g_1, ..., g_n]$ ocorrem em ψ, se $\mathcal{R} \models A([fk_1, ..., fk_m] \rightarrow f_j)$, onde $fk_1, ..., fk_m$ e f_j são membros distintos de $\{f_1, ..., f_n\}$, então $\mathcal{R} \models B([gk_1, ..., gk_m] \rightarrow g_j)$ e $\mathcal{R} \models <A([fk_1, ..., fk_m] \rightarrow f_j), B([gk_1, ..., gk_m] \rightarrow g_j)>$.♦

Em um esquema restrito, para cada DE ψ em $\mathcal{R}$ tal que $A[f_1, ..., f_n]$ e $B[g_1, ..., g_n]$ estão em ψ, se nós definirmos as classes hipotéticas A' e B' tais que ***atributos***$(A')=\{f_1, ..., f_n\}$, ***atributos***$(B')=\{g_1, ..., g_n\}$, e o conjunto de DFs sobre A' é a projeção das DFs sobre A em $\{f_1,...,f_n\}$, e o conjunto de DFs sobre B' é a projeção das DFs sobre B em $\{g_1, ..., g_n\}$, então:

(i) $\{fk_1, ..., fk_m\} \in$ ***S_chaves***(A') onde $\{fk_1, ..., fk_m\} \subseteq \{f_1, ..., f_n\}$sss $\{gk_1, ..., gk_m\} \in$ ***S_chaves***(B'), e

(ii) Se $\{fk_1, ..., fk_m\} \in$ ***S_chaves***(A') onde $\{fk_1, ..., fk_m\} \subseteq \{f_1, ..., f_n\}$ então $\mathcal{R} \models <A'([fk_1, ..., fk_m] \rightarrow f_j), B'([gk_1, ..., gk_m] \rightarrow g_j)>$, para $1 \leq j \leq n$.

Por exemplo, o esquema na Figura 4.1 não é um esquema restrito já que ***PROGENITORES****(NM→OCN)* não é logicamente implicado por $\mathcal{R}$.



{NM} ∈ ***chaves****(MÃES)*

{NP,NM}∈ ***chaves****(PAIS)*

MÃES[NM, NFV]⊂***PROGENITORES****[NM, NFV*

Abreviações: NM= Nome da Mãe, NP= Nome do Pai, NFV= Nome do Filho mais Velho

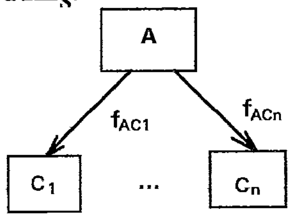
Figura 4.1: Exemplo de um esquema que não é restrito

Teorema 4.1: Para qualquer esquema restrito S, existe uma transformação T, tal que T preserva a informação e a semântica das atualizações, e o esquema transformado resultante $S'=T(S)$ está em forma normal decomposta.♦

O algorítimo de decomposição apresentado na Seção 4.1.2 mostra isso.

Na metodologia proposta, o processo de normalização é executado em pequenos passos usando a primitiva de transformação especificada na Figura 4.2. Nas especificações das primitivas de transformações omitiremos a semântica de atualização das restrições para as quais somente uma alternativa existe. A prova de que ***Decomponha*** preserva a informação e a semântica das atualizações se encontra no Apêndice B.

PRE$_S$:



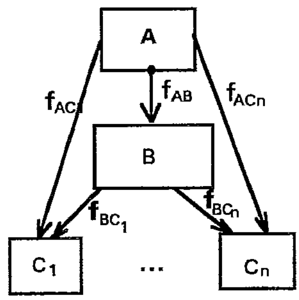
PRE$_{rest}$:

$\{f\text{AC}_1,\ldots,f\text{AC}_n\} \subseteq \textbf{\textit{atributos}}(A)$

$\{f\text{AC}_1,\ldots,f\text{AC}_n\} \not\in \textbf{\textit{chaves}}(A)$

$\downarrow$ $\textbf{\textit{Decomponha}}(A,\ \{f\text{AC}_1,\ldots,f\text{AC}_n\},\ \boldsymbol{B}, f\text{AB}\ ,\{f\text{BC}_1\ ,\ \ldots\ ,f\text{BC}_n\ \})$

POS$_S$:



POS$_{rest}$:

(i) adiciona restrição:

- $f\text{AB}: A \rightarrow \boldsymbol{B}$ com semântica de atualização :
  - $\boldsymbol{SI}\ (A,\ (f\text{AB}: A \rightarrow \boldsymbol{B})) =$ **propaga;**
  - $\boldsymbol{SD}\ (\boldsymbol{B},\ (f\text{AB}: A \rightarrow \boldsymbol{B})) =$ **propaga;**
- $f\text{AB}: A \bullet\!\!\rightarrow \boldsymbol{B}$ com semântica de atualização :
  - $\boldsymbol{SI}(\boldsymbol{B},\ (f\text{AB}: A \bullet\!\!\rightarrow \boldsymbol{B})$ =**propaga;**
  - $\boldsymbol{SD}\ (A,\ (f\text{AB}: A \bullet\!\!\rightarrow \boldsymbol{B})$=**propaga;**
- $f\text{AC}_i \equiv f\text{AB} \circ f\text{BC}_i$; $\textbf{\textit{restrições}}(f\text{BC}_i) = \textbf{\textit{restrições}}(f\text{AC}_i)$, $1\leq i \leq n$,

(ii) Substitua toda FD $\gamma = A(f\text{ACi}_1, \ldots, f\text{ACi}_m \rightarrow f\text{AC}_j)$ onde $i_k \in [1, n]$ e $j \in [1, n]$, pela FD $\boldsymbol{B}(f\text{BCi}_1\ , \ldots, f\text{BCi}_m \rightarrow f\text{BC}_j)$. ($\gamma$ torna-se redundante)

(ii) Substitua toda DFU ψ tal que $A(f\text{ACi}_1, \ldots, f\text{ACi}_m \rightarrow f\text{AC}_j)$ é um membro de ψ, $i_k \in [1, n]$ e $j \in [1, n]$, pela DFU ψ' onde ψ' é obtida de ψ trocando-se $A(f\text{ACi}_1, \ldots, f\text{ACi}_m \rightarrow f\text{AC}_j)$ por $\boldsymbol{B}(f\text{BCi}_1\ , \ldots, f\text{BCi}_m \rightarrow f\text{BC}_j)$.

(iv) Substitua toda DE ψ tal que $A[f\text{ACi}_1\ , \ldots, f\text{ACi}_m]$ ocorre em ψ, $i_k \in [1, n]$, pela DE ψ' onde ψ' é obtida de ψ trocando-se $A(f\text{ACi}_1, \ldots, f\text{ACi}_m \rightarrow f\text{AC}_j)$ por $\boldsymbol{B}(f\text{BCi}_1\ , \ldots, f\text{BCi}_m \rightarrow f\text{BC}_j)$.

$\boldsymbol{\sigma}$**:** Para qualquer instância $\mathcal{B}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{B})(A) = \mathcal{B}(A)$.

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

$\boldsymbol{\phi}$**:** Para qualquer instância $\mathcal{B}$ do esquema original, temos:

(i) $\phi(\mathcal{B})(A) = \mathcal{B}(A)$.

(ii) $\phi(\mathcal{B})(\boldsymbol{B}) = \{\ \boldsymbol{b} \mid (\exists\ \boldsymbol{a})\ (\boldsymbol{a} \in \mathcal{B}(A))\ (\text{para } 1\leq i\leq n\ (\ \phi(\ \mathcal{B})(f_{\text{BCi}})\ (\boldsymbol{b}) = \mathcal{B}(f_{\text{ACi}})(\boldsymbol{a})))\}$.

(iii) $\forall \boldsymbol{a} \in \phi(\mathcal{B})(A)\ \phi(\ \mathcal{B})(f_{\text{AB}})\ (\boldsymbol{a}) = \boldsymbol{b}$ onde $\boldsymbol{b} \in \phi(\mathcal{B})(\boldsymbol{B})$ e

(para $1\leq i\leq n\ (\phi(\mathcal{B})(f_{\text{BCi}})\ (\boldsymbol{b}) = \mathcal{B}(f_{\text{ACi}})(\boldsymbol{a}))$),

(iv) Todas as outras classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema transformado.

$\boldsymbol{\tau}$**:** (i) τ("remova $\boldsymbol{a}$ de A") ="remova $\boldsymbol{a}$ de A"

(ii) τ("adicione $\boldsymbol{a}$ em A") = "adicione $\boldsymbol{a}$ a A"

(iii) τ("adicione $\boldsymbol{a}(f_1 = v_1, \ldots, f_m = v_m)$ em A ") =

"adicione $\boldsymbol{a}(f_1 = v_1, \ldots, f_m = v_m, f\text{AB} = \boldsymbol{b}(f\text{BC}_1 = vi_1, \ldots, f\text{BC}_n = vi_n)$ em A", onde para qualquer $1\leq j\leq m$ e $1\leq k\leq n$, se $f\text{AC}_k \equiv f_j$ então $vi_k = v_j$

(iv) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 4.2: A primitiva de transformação ***DECOMPONHA***

4.1.2 O ALGORÍTIMO DE DECOMPOSIÇÃO

Nessa seção, após a apresentação de alguns conceitos preliminares, descrevemos um algorítimo para transformar um esquema restrito em outro equivalente que esteja na forma normal decomposta. Aquí, consideramos que S=($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$) seja um esquema onde $\mathcal{R}$ é uma cobertura mínima.

Definição 4.5: Seja C uma classe de S. C está em *Forma Normal Decomposta* (FND) sss:

(i) Se $C([f_1, ...,f_n] \rightarrow g)$ é uma DF em $\mathcal{R}$ então $\{f_1, ...,f_n\} \in$ *S_chaves*(C).

(ii) Se $C([f_1, ...,f_n] \rightarrow g)$ é um membro de uma DFU em $\mathcal{R}$, então $\{f_1, ...,f_n\} \in$ *S_chaves*(C).

(iii) Se ψ é uma DE em $\mathcal{R}$ tal que $C[f_1, ...,f_n]$ ocorre em ψ, então $\{f_1, ...,f_n\} \in$ *S_chaves*(C).

Teorema 4.2: Seja S=($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$) um esquema restrito. Se todas as classes em $\mathcal{C}$ estiverem em FND então S está em forma normal decomposta.♦

Prova:

Suponha que todas as classes em $\mathcal{C}$ estejam em FND. Então, as condições (i) e (ii) da Definição 4.5 são satisfeitas.

Para mostrar que a condição (iii) da Definição 4.5 também é satisfeita, vamos supor que exista uma DE ψ em $\mathcal{R}$, tal que $A[f_1, ...,f_n]$ e $B[g_1, ...,g_n]$ ocorram em ψ.

(a) Como A está em FND, pela Definição 4.5 temos que $\{f_1, ...,f_n\} \in$ *S_chaves*(A). Assim, a condição (iii-a) é satisfeita.

(b) Suponha que $\{fk_1, ...,fk_m\} \in$ *S_chaves*(A) e que $\{fk_1, ...,fk_m\} \subseteq \{f_1, ...,f_n\}$. Então $A([fk_1, ..., fk_m] \rightarrow f_i)$ para $1 \leq i \leq n$. Pela definição de esquema restrito, $B([gk_1, ...,gk_m] \rightarrow g_i$) para $1 \leq i \leq n$. Como B está em FND, pela Definição 4.5 temos que $\{g_1, ...,g_n\} \in$ *S_chaves*(B). De acordo com os axiomas de Armstrong [Ullman83] temos que, para qualquer $g \in$ *atributos*(B), $B([gk_1, ...,gk_m] \rightarrow g)$. Conseqüentemente, $\{gk_1, ...,gk_m\} \in$ *S_chaves*(B). Analogamente, podemos provar que: se $\{gk_1, ...,gk_m\} \in$ *S_chaves*(B) então $\{fk_1, ...,fk_m\} \in$ *S_chaves*(A)

(c) Suponha que $\{fk_1, ...,fk_m\} \in$ *S_chaves*(A) onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ...,f_n\}$. Então, $\mathcal{R} \models A(fk_1, ..., fk_m \rightarrow f_j)$, para $1 \leq j \leq n$. Pela definição de esquema restrito $\mathcal{R} \models \langle A([fk_1, ...fk_m] \rightarrow f_j), B([gk_1, ...,gk_m] \rightarrow g_j) \rangle$, para $1 \leq j \leq n$.

A partir de (a), (b) e (c) podemos concluir que a condição (iii) da Definição 4.5 também é satisfeita e S está em forma normal decomposta.♦

Pelo Teorema 4.2, o problema de transformar um esquema restrito em um outro equivalente que esteja em forma normal decomposta, pode ser reduzido ao problema de transformar um esquema restrito em um outro equivalente em que todas as classes estejam em FND. Para determinar o conjunto de transformações requeridas para a normalização de uma classe, usamos o grafo de decomposição da classe definido a seguir.

Definição 4.6: Um *grafo de decomposição* de uma classe C em S é um grafo acíclico direcionado (V, A), onde V é um conjunto de classes, $C \in V$, e A é um conjunto de arcos direcionados. Para cada classe F em V, $atributos^*(F)$[1] $\subseteq atributos^*(C)$, e o conjunto de DFs em F é a projeção das DFs de C em $atributos^*(F)$. Se o arco a de E para F, representado por $a=\langle E,F \rangle$, está em A então $atributos^*(F) \subset atributos^*(E)$. ♦

Definição 4.7: Seja $D_C=(V, A)$ o grafo de decomposição da classe C e seja E uma classe em V. O conjunto de classes referenciadas por E em D_C, denotado $referências(E)$, é o menor conjunto tal que:

(i) Se $\langle E,G \rangle$ está em A então $G \in referências(E)$, e

(ii) Se $G \in referências(E)$ e $H \in referências(G)$ então $H \in referências(E)$. ♦

Definição 4.8: Seja $D_C=(V, A)$ um grafo de decomposição da classe C em S. *A transformação definida pelo grafo* D_C é a transformação obtida ao se percorrer o grafo de decomposição D_C da maneira indicada no Algorítimo 4.1.

Algorítimo 4.1:

Entrada:

$S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ - Um esquema

C - Uma classe em S

$D_C = (V, A)$ - Um grafo de decomposição de C

Saída:

T - A transformação definida pelo grafo D_C .

começo

- faça $D := \{C\}$;
- faça ***Visitados*** $:= \emptyset$;
- i :=0;
- se $A=\emptyset$ então T é a transformação identidade

  senão

[1] $atributos^*(F)$ é o conjunto dos nomes dos atributos de F.

- <u>enquanto</u> $D \neq \emptyset$ <u>faça</u>

  - selecione classe E em D;

  - <u>se</u> $E \notin$ ***Visitados*** <u>então</u>

    - <u>para cada</u> arco $<E, F>$ em A onde $atributos^*(F)=\{f_1,...,f_n\}$ <u>faça</u>

      - i:=i+1;

      - T_i = ***Decomponha***$(E, \{f_1,...,f_n\}, F, f_{EF}, \{f_1,...,f_n\})$;

      - adicione F a D;

    - adicione E a ***Visitados***;

  - remova E de D;

  <u>fim</u> <u>enquanto</u>;

- $T = T_i \circ ... \circ T_1$

<u>fim</u> <u>se</u>;

<u>fim</u> algorítimo.

<u>Definição 4.9</u>: Suponha que $D_C=(V, A)$ seja o grafo de decomposição da classe C em S. Seja T a transformação definida pelo grafo D_C, e seja $S' = T(S)$. D_C *define uma decomposição normalizada de* C sss, para qualquer classe F em S', se $F \in V$ então F está em FND.

Para os exemplos 4.1 e 4.2, suponha que C seja uma classe de S, onde f_1, f_2, f_3, f_4 e f_5 são atributos de C e $\{C([f_1] \rightarrow f_2),\ C([f_1] \rightarrow f_4),\ C[f_1,\ f_2,\ f_3] \subset E[g_1,\ g_2,\ g_3]\}$ é o conjunto de todas as DEs e DFs em $\mathcal{R}$ sobre C.

<u>Exemplo 4.1</u>: Considere o grafo de decomposição D_C mostrado na Figura 4.3. De acordo com o Algorítimo 4.1 a transformação T definida pelo grafo D_C é dada por

$T = T_3 \circ T_2 \circ T_1$ onde:

T_1: ***Decomponha***$(C, \{f_1, f_4\}, B, f_{CB}, \{f_1, f_4\})$

T_2: ***Decomponha***$(C, \{f_1, f_2, f_3\}, A, f_{CA}, \{f_1, f_2, f_3\})$

T_3: ***Decomponha***$(A, \{f_1, f_2\}, D, f_{AD}, \{f_1, f_2\})$

Aplicando-se a transformação T ao esquema S, obtemos o esquema $S'=(\mathcal{C}', \mathcal{A}', \mathcal{R}')$ onde $\{D([f_1] \rightarrow f_2),\ B([f_1] \rightarrow f_4),\ A[f_1,\ f_2,\ f_3] \subset E[g_1,\ g_2,\ g_3]\}$ é o conjunto de todas as DEs e DFs em $\mathcal{R}'$ sobre as classes em V. Com base na Definição 4.9, observamos que todas as classes em V estão em FND. Assim, D_C define uma decomposição normalizada de C.

<u>Exemplo 4.2</u>: Considere o grafo de decomposição D_C mostrado na Figura 4.4. De acordo com o Algorítimo 4.1 a transformação T definida pelo grafo D_C é dada por

$T = T_2 \circ T_1$ onde:

T_1: ***Decomponha***$(C, \{f_1, f_2, f_3, f_4\}, B, f_{CB}, \{f_1, f_2, f_3, f_4\})$

T_2: ***Decomponha***$(B, \{f_1, f_2, f_4\}, A, f_{BA}, \{f_1, f_2, f_4\})$

Aplicando-se as transformações a cima ao esquema S, obtemos o esquema $S''=(\mathcal{C}'', \mathcal{A}'', \mathcal{R}'')$ onde $\{A([f_1] \rightarrow f_2),\ A([f_1] \rightarrow f_4), B[f_1, f_2, f_3] \subset E[g_1, g_2, g_3]\}$ é o conjunto de todas as DEs e DFs em $\mathcal{R}''$ sobre as classes em V. Com base na Definição 4.9, observamos que todas as classes em V estão em FND. Assim, D_C define uma decomposição normalizada de C.

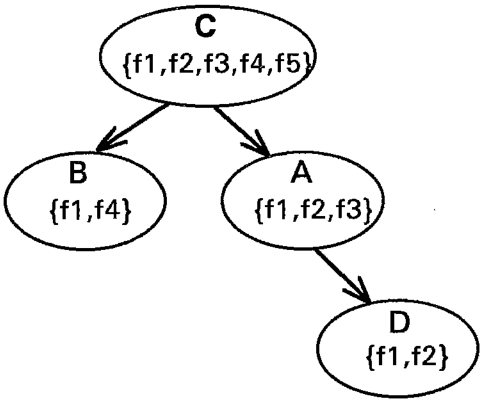


Figura 4.3: Grafo de Decomposição D_C para Exemplo 4.1

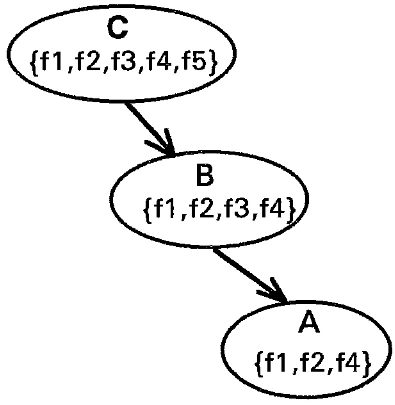


Figura 4.4: Grafo de Decomposição D_C para Exemplo 4.2

Os exemplos 4.1 e 4.2 mostram que é possível existir mais de um grafo de decomposição para uma classe C que defina uma decomposição normalizada de C. No entanto, estamos interessados naquele que define uma *decomposição normalizada ótima* de C, definida como se segue.

<u>Definição 4.10</u>: Suponha que $D_C=(V, A)$ seja um grafo de decomposição da classe C em S. Seja T a transformação definida pelo grafo D_C. *D_C define uma decomposição normalizada ótima de C* sss :

(i) D_C define uma decomposição normalizada de C;

(ii) Para quaisquer classes E e G em V, se $E \neq G$ então se $\{f_1, ...f_m\} \in$ ***chaves****(E) então $\{f_1, ...f_m\} \notin$ ***chaves****(G)

(iii) Para quaisquer classes E e G em V, se $\{f_1, ...f_m\} \in$ ***chaves****(E) e $\{f_1, ...f_m\} \subseteq$ ***atributos****(G) então $E \in$ ***referências***(G). ♦

Com base na Definição 4.10 o grafo de decomposição no Exemplo 4.1 não define uma decomposição normalizada ótima de C, enquanto que o grafo de decomposição no Exemplo 4.2 define uma decomposição normalizada ótima de C. A seguir, apresentamos um algorítimo para se obter um grafo de decomposição para uma classe C o qual define uma decomposição normalizada ótima de **C**.

<u>Algorítimo 4.2</u>:

Entrada:

S=($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$) - Um esquema restrito, onde $\mathcal{R}$ é uma cobertura mínima.

C - Uma classe em S.

Saída:

(V, A) - Um grafo de decomposição de C que define uma decomposição normalizada ótima de C.

<u>Começo</u>

/* *Criação dos vértices* (V) */

1. - faça $V := \{C\}$;
2. - <u>para todas as</u> DFs $C([f_1,...,f_n] \rightarrow g)$ em $\mathcal{R}$ <u>faça</u>
3. - crie a classe E e faça $atributos^*(E) := \{f_1,...,f_n, g\}$
4. - defina o conjunto de DFs sobre E como a projeção das DFs sobre C em $\{f_1,...,f_n, g\}$
5. - adicione E a V
6. - <u>para todas as</u> DFUs ψ em $\mathcal{R}$, onde $C([f_1,...,f_n] \rightarrow g)$ é um membro de ψ <u>faça</u>
7. - crie a classe E e faça $atributos^*(E) := \{f_1,...,f_n, g\}$
8. - defina o conjunto de DFs sobre E como a projeção das DFs sobre C em $\{f_1,...,f_n, g\}$.
9. - adicione E a V
10. - <u>para todas as</u> DEs ψ em $\mathcal{R}$, onde $C[f_1,...,f_n]$ está em ψ <u>faça</u>
11. - crie a classe E e faça $atributos^*(E) := \{f_1,...,f_n\}$
12. - defina o conjunto de DFs sobre E como a projeção das DFs sobre C em $\{f_1,...,f_n\}$.
13. - adicione E a V
14. - <u>para todas as</u> classes E e F em V onde existe $\{f_1,...,f_n\} \subseteq atributos^*(E)$ tal que $\{f_1, ...f_n\} \in chaves^*(E)$ e $\{f_1, ...f_n\} \in chaves^*(G)$ <u>faça</u>
15. - crie a classe G e faça $atributos^*(G) := atributos^*(E) \cup atributos^*(F)$
16. - defina o conjunto de DFs sobre G como a união do conjunto de DFs sobre E com o conjunto de DFs sobre F
17. - adicione G a V
18. - remova E e F de V
19. - <u>para todas as</u> classes E e F em V onde existe $\{f_1,...,f_n\} \subseteq atributos^*(E)$ tal que $\{f_1, ...f_n\} \in chaves^*(E)$ e $\{f_1, ...f_n\} \subset atributos^*(F)$ <u>faça</u>
20. - faça $atributos^*(F) := atributos^*(E) \cup atributos^*(F)$;

/* *Criação dos arcos* (**A**) */

21. - faça **A** := ∅
22. - <u>para todas as</u> classes **E** e **F** em **V** onde *atributos**(**E**)⊂*atributos**(**F**) e não existe **D** em **V** *tal que* *atributos**(**E**) ⊂ *atributos**(**D**) e *atributos**(**D**) ⊂ *atributos**(**F**) <u>faça</u>
23. - adicione < **E**, **F**> a **A**.

<u>fim</u> algorítimo.

Antes de apresentarmos a prova de que o grafo de decomposição obtido com o Algorítimo 4.2 define uma decomposição normalizada ótima da classe **C**, enunciamos um lema que é usado nessa prova e que decorre diretamente da definição de uma primitiva de transformação ***Decomponha***.

<u>Lema 4.1</u>: Seja S=($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$) um esquema em que $\mathcal{R}$ é uma cobertura mínima e seja **C** uma classe em $\mathcal{C}$. Considere que D_C=(**V**, **A**) seja um grafo de decomposição de *C* e que **T** seja uma transformação definida pelo grafo D_C. Considere também o esquema S'=($\mathcal{C}'$, $\mathcal{A}'$, $\mathcal{R}'$) onde $S'=T(S)$ e $\mathcal{R}'$ é uma cobertura mínima.

(i) Se existe uma classe **E** em **V** onde $\{f_1,..., f_n, g\} \subseteq$ *atributos**(**E**) então para qualquer classe **F** em **V** tal que **E**∈***referências***(**F**), a DF $F([f_1, ..., f_n] \rightarrow g)$ não está em $\mathcal{R}'$ e $F([f_1, ..., f_n] \rightarrow g)$ não é membro de nenhuma DFU em $\mathcal{R}'$. Isso se deve ao fato de que $\mathcal{R}'$ é uma cobertura mínima e ao fato de que se a DF, $F([f_1, ..., f_n] \rightarrow g$, ocorre, ela é redundante.

(ii) Se existe uma classe **E** em **V** onde $\{f_1,..., f_n\} \subseteq$ *atributos**(**E**) então para qualquer classe **F** em **V** tal que **E**∈***referências***(**F**), não existe nenhuma DE ψ em $\mathcal{R}'$ onde $F[f_1, ..., f_n]$ está em ψ. ♦

<u>Teorema 4.3</u>: Seja **S** um esquema restrito, seja **C** uma classe em **S** e seja D_C=(**V**, **A**) o grafo de decomposição de **C** produzido pelo Algorítimo 4.2. Então, D_C define uma decomposição normalizada ótima de **C**.

<u>Prova</u>:

Seja **T** a transformação definida por D_C e $S'=T(S)$.

(i) Para mostrarmos que D_C define uma decomposição normalizada de **C** temos que provar que, para qualquer classe **F** em **S'**, se **F** ∈ **V** então **F** está em FND.

Seja **F** uma classe em **S'**. Suponha que **F** ∈ **V**. Para mostrarmos que **F** está em FND, precisamos provar que as condições (i), (ii) e (iii) da Definição 4.5 são satisfeitas.

(a) Suponha que F não satisfaça a condição (i) da Definição 4.5. Assim, existe uma DF $F([f_1, ...f_n] \rightarrow g)$ em $\mathcal{R}'$ onde $\{f_1,...,f_n\} \notin S\_chaves(F)$. Logo, $C([f_1, ...f_n] \rightarrow g)$ está em $\mathcal{R}$, uma vez que todas as DFs de F eram DFs de C em S. Por conseguinte, nas linhas de 2 a 5, o Algorítimo 4.2 cria a classe E' em V onde $\{f_1,..., f_n, g\}=$ *atributos*$^*(E')$. Assim, $\{f_1,..., f_n\} \in S\_chaves^*(E')$. Se E' for substituida por uma outra classe E nas linhas de 10 a 17, E terá todas as chaves e atributos de E' e, assim, passaremos a chamar E' de E. Uma vez que $\{f_1,..., f_n\} \subset$ *atributos*$^*(F)$, então, segundo as linhas 18 e 19, *atributos*$^*(E) \subset$ *atributos*$^*(F)$. Logo, $E \in$ ***referências***(F) (segundo as linhas 21 e 22) e como conseqüência do Lema 4.1 $F([f_1, ...f_n] \rightarrow g) \notin \mathcal{R}'$, o que é uma contradição. Concluimos, portanto, que F satisfaz a condição (i) da Definição 4.5.

(b) Suponha que F não satisfaz a condição (ii) da Definição 4.5.
Assim, existe uma DFU ψ' em $\mathcal{R}'$ onde $F([f_1, ...f_n] \rightarrow g)$ é um membro de ψ' e $\{f_1,..., f_n\} \notin S\_chaves(F)$. Usando a mesma linha de raciocínio usada em (a), existe uma DE ψ em $\mathcal{R}$ tal que $C([f_1, ...f_n] \rightarrow g)$ é um membro de ψ e $\{f_1,..., f_n\} \notin S\_chaves(C)$. Conseqüentemente, nas linhas de 6 a 9, o Algorítimo 4.2 cria a classe E' em V onde $\{f_1,..., f_n, g\}=$ *atributos*$^*(E')$. Assim, $\{f_1,..., f_n\} \in S\_chaves^*(E')$. Se E' for substituida por uma outra classe E nas linhas de 10 a 17, E terá todas as chaves e atributos de E' e, assim, passaremos a chamar E' de E. Uma vez que $\{f_1,..., f_n\} \subset$ *atributos*$^*(F)$, então, segundo as linhas 18 e 19, *atributos*$^*(E) \subset$ *atributos*$^*(F)$. Logo, $E \in$ ***referências***(F) (segundo as linhas 21 e 22) e como conseqüência do Lema 4.1 $\psi' \notin \mathcal{R}'$, o que é uma contradição. Concluimos, portanto, que F satisfaz a condição (ii) da Definição 4.5.

(c) Suponha que F não satisfaz a condição (iii) da Definição 4.5.
Assim, existe uma DE ψ' em $\mathcal{R}'$ tal que $F[f_1, ...f_n]$ está em ψ e $\{f_1, ...f_m\} \notin S\_chaves^*(F)$. Usando a mesma linha de raciocínio usada em (a), existe uma DE ψ em $\mathcal{R}$ tal que $C([f_1, ...f_n]$ está em ψ. Conseqüentemente, nas linhas de 10 a 13, o Algorítimo 4.2 cria a classe E' em V onde $\{f_1,...f_n\}=$ *atributos*$^*(E)$ e $\{f_1,...f_n\} \in$ ***chaves***$^*(E)$. Assim, $\{f_1,..., f_n\} \in S\_chaves^*(E')$. Se E' for substituida por uma outra classe E nas linhas de 10 a 17, E terá todas as chaves e atributos de E' e, assim, passaremos a chamar E' de E. Uma vez que $\{f_1, ...f_m\} \subset$ *atributos*$^*(F)$, então, segundo as linhas 14 e 15, *atributos*$^*(E) \subset$ *atributos*$^*(F)$. Logo, $E \in$ ***referências***(F) (segundo as linhas 21 e 22) e como conseqüência do Lema 4.1 $\psi' \notin \mathcal{R}'$, o que é uma contradição. Concluimos, portanto, que F satisfaz a condição (iii) da Definição 4.5.

A partir de (a), (b) e (c) podemos concluir que F está em FND. Conseqüentemente, D_C define uma decomposição normalizada de C.

(ii) A condição (ii) da Definição 4.10 decorre diretamente das linhas 14 a 17 e a condição (iii) da Definição 4.10 decorre diretamente das linhas 18 a 22. ♦

Finalmente, apresentamos, a seguir, um algorítimo para transformar um esquema restrito em um outro equivalente que esteja em forma normal decomposta:

<u>Algorítimo 4.3</u>:

Entrada:

$S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ - Um esquema restrito, onde $\mathcal{R}$ é uma cobertura mínima.

Saída:

$S'=(\mathcal{C}', \mathcal{A}', \mathcal{R}')$ - Um esquema equivalente a S que está em forma normal decomposta.

<u>começo</u>

i:=0;

<u>para cada</u> classe C em $\mathcal{C}$ <u>faça</u>

<u>começo</u>

- i:=i+1;
- determine o grafo de decomposição $D_C=(V, A)$ de C que define uma decomposição normalizada ótima de C (Algorítimo 4.2);
- obtenha a transformação T_i definida pelo grafo D_C (Algorítimo 4.1);

<u>fim faça</u>

$S'= T_i \circ \ldots \circ T_1(S)$

<u>fim</u> algorítimo.

<u>Teorema 4.4</u>: Seja $S'=(\mathcal{C}', \mathcal{A}', \mathcal{R}')$ o esquema produzido pelo algorítimo de decomposição 4.3 quando um esquema restrito $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ é fornecido como entrada. Então o esquema S' é equivalente a S e está em forma normal decomposta. ♦

<u>Prova</u>: Como a primitiva de transformação ***Decomponha*** preserva a informação e a semântica das atualizações e é a única transformação utilizada pelo algorítimo de decomposição, podemos garantir que S' é equivalente a S. Pelo Teorema 4.3 todas as classes de S' estão em FND. Pelo Teorema 4.2 podemos concluir que S' está em forma normal decomposta.

4.2 TRANSFORMAÇÃO DE RESTRIÇÕES DE DEPENDÊNCIAS EXISTENCIAIS EM RESTRIÇÕES ESTRUTURAIS E RESTRIÇÕES DE EQUIVALÊNCIA_DE_RELACIONAMENTOS

Quando um esquema está em forma normal decomposta, podemos substituir todas as restrições de dependência existenciais por restrições estruturais e de equivalência_de_relacionamentos usando as primitivas de transformação **TDE#1**, **TDE#2**, **TDE#3**, **TDE#4**, **TDE#5**, and **TDE#6** especificadas nas figuras 4.5 a 4.9. "**TDE**" india "transfor-mação de dependência existencial". A prova de que a primitiva de transformação **TDE#1** preserva a informação e a semântica das atualizações, encontra-se no apêndice B.

PRE$_S$:

$A[f_1,...,f_n] \subset B[g_1,...,g_n]$ onde

- $\{f_1, ..., f_n\} \in$ *S_chaves*(A),
- $\{fk_1, ..., fk_m\} \in$ *S_chaves*(A), onde $\{fk_1, ..., fk_m\} \subseteq \{f_1, ..., f_n\}$, sss $\{gk_1, ..., gk_m\} \in$ *S_chaves*(B)
- Se $\{fk_1, ..., fk_m\} \in$ *S_chaves*(A) onde $\{fk_1, ..., fk_m\} \subseteq \{f_1, ..., f_n\}$ então $\mathcal{R} \models \langle A([fk_1, ..., fk_m] \rightarrow f_j), B([gk_1, ..., gk_m] \rightarrow g_j) \rangle$, para $1 \le j \le n$.

$\downarrow$ **TDE#1**$(A[f_1,...,f_n] \subset B[g_1,...,g_n])$

PÓS$_{const}$:

Substitua a restrição de dependência existencial $\psi = A[f_1,...,f_n] \subset B[g_1,...,g_n]$ pelas seguintes restrições:

- $\psi' = A \subset B$, com semântica de atualização :

  $SI(A, \psi') = SI(A, \psi)$

  $SD(B, \psi') = SD(B, \psi)$
- $f_i \equiv g_i$ para $1 \le i \le n$.

Obs.: Como todas as classes e atributos no esquema original são diretamente transformados em classes e atributos do esquema transformado, σ, ϕ e τ são mapeamentos-identidades.

Figura 4.5 - A primitiva de transformação **TDE#1**

$\mathbf{PRE_S}$:

$A[f_1,...,f_n] = B[g_1,...,g_n]$ onde

- $\{f_1, ..., f_n\} \in$ ***S_chaves***(A),
- $\{fk_1, ...fk_m\} \in$ ***S_chaves***(A), onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$, sss $\{gk_1, ...gk_m\} \in$ ***S_chaves***(B)
- Se $\{fk_1, ...fk_m\} \in$ ***S_chaves***(A) onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$ então $\mathcal{R} \models \langle A([fk_1, ...fk_m] \rightarrow f_j), B([gk_1, ...gk_m] \rightarrow g_j) \rangle$, para $1 \le j \le n$.

$\downarrow$ **TDE#2**$(A[f_1,...,f_n] = B[g_1,...,g_n])$

$\mathbf{PÓS_{const}}$:

Substitua a restrição de dependência existencial $A[f_1,...,f_n] = B[g_1,...,g_n]$ pelas seguintes restrições:

- $A = B$, com semântica de atualização :

  $SI\ (A, \psi') = SI\ (A, \psi)$

  $SI\ (B, \psi') = SI\ (B, \psi)$

  $SD\ (A, \psi') = SD\ (A, \psi)$

  $SD\ (B, \psi') = SD\ (B, \psi)$
- $f_i \equiv g_i$ para $1 \le i \le n$.

Obs.: Como todas as classes e atributos no esquema original são diretamente transformados em classes e atributos do esquema transformado, σ, ϕ e τ são mapeamentos-identidades.

Figura 4.6 - A primitiva de transformação **TDE#2**

$\mathbf{PRE_S}$:

$A[f_1,...,f_n] \mid B[g_1,...,g_n]$ onde

- $\{f_1, ..., f_n\} \in$ ***S_chaves***(A),
- $\{fk_1, ...fk_m\} \in$ ***S_chaves***(A), onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$, sss $\{gk_1, ...gk_m\} \in$ ***S_chaves***(B)
- Se $\{fk_1, ...fk_m\} \in$ ***S_chaves***(A) onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$ então $\mathcal{R} \models \langle A([fk_1, ...fk_m] \rightarrow f_j), B([gk_1, ...gk_m] \rightarrow g_j) \rangle$, para $1 \le j \le n$.

$\downarrow$ **TDE#3**$(A[f_1,...,f_n] \mid B[g_1,...,g_n])$

$\mathbf{PÓS_{const}}$:

Substitua a restrição de dependência existencial $\psi = A[f_1,...,f_n] \mid B[g_1,...,g_n]$ pelas seguintes restrições:

- $\psi' = A \mid B$, com semântica de atualização :

  $SI\ (A, \psi') = SI\ (A, \psi)$

  $SI\ (B, \psi') = SI\ (B, \psi)$
- $f_i \equiv g_i$ para $1 \le i \le n$.

Obs.: Como todas as classes e atributos no esquema original são diretamente transformados em classes e atributos do esquema transformado, σ, ϕ e τ são mapeamentos-identidades.

Figura 4.7 - A primitiva de transformação **TDE#3**

PRE$_S$:

$A[f_1,...,f_n] = B[g_1,...,g_n]$ - $C[h_1,...,h_n]$ onde

- $\{f_1, ..., f_n\} \in$ *S_chaves*(A),
- $\{fk_1, ...fk_m\} \in$ *S_chaves*(A), onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$, sss $\{gk_1, ...gk_m\} \in$ *S_chaves*(B)
- $\{fk_1, ...fk_m\} \in$ *S_chaves*(A), onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$, sss $\{hk_1, ...hk_m\} \in$ *S_chaves*(C)
- Se $\{fk_1, ...fk_m\} \in$ *S_chaves*(A) onde $\{fk_1, ...fk_m\} \subseteq \{f_1, ...f_n\}$ então

$\mathcal{R} \models <A([fk_1, ...fk_m] \rightarrow f_j), B([gk_1, ...gk_m] \rightarrow g_j, C([hk_1, ...hk_m] \rightarrow h_j)>$, para $1 \leq j \leq n$.

$\downarrow$ **TDE#4(** $A[f_1,...,f_n] = B[g_1,...,g_n]$ - $C[h_1,...,h_n]$)

PÓS$_{const}$:

Substitua a restrição de dependência existencial $\psi = A[f_1,...,f_n] = B[g_1,...,g_n]$ - $C[h_1,...,h_n]$ pelas seguintes restrições:

- $\psi' = (A = B - C)$ com semântica de atualização :

  $SI\ (A, \psi') = SI\ (A, \psi)$

  $SI\ (B, \psi') = SI\ (B, \psi)$

  $SI\ (C, \psi') = SI\ (C, \psi)$

  $SD\ (A, \psi') = SD\ (A, \psi)$

  $SD\ (B, \psi') = SI\ (B, \psi)$

  $SD\ (C, \psi') = SI\ (C, \psi)$

- $f_i = g_i$ para $1 \leq i \leq n$.

Obs.: Como todas as classes e atributos no esquema original são diretamente transformados em classes e atributos do esquema transformado, σ, ϕ e τ são mapeamentos-identidades.

Figura 4.8 - A primitiva de transformação **TDE#4**

PRE_S:

$A[f_1, ..f_n] = \bigcup_{i=1}^{m} C_i\, [f\, i_1, ..f\, i_n]$ onde

- $\{f_1, ..f_n\} \in S\_chaves(A)$,
- $\{fk_1, ...,fk_m \in S\_chaves(A)$, onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ..f_n\}$, sss $\{f\, i_{k1}, ..,f\, i_{kn}\} \in S\_chaves(C_i)$, para $1 \leq i \leq n$.
- Se $\{fk_1, ...,fk_m\} \in S\_chaves(A)$ onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ...,f_n\}$ então $\mathcal{R} \models \langle A([fk_1, ...,fk_m] \rightarrow f_j),\ C_1([f\, 1_{k1}, ..,f\, 1_{kn}] \rightarrow f\, 1_j), ... ,$
  $C_m([f\, m_{k1}, ..,f\, m_{kn}] \rightarrow f\, m_j) \rangle$, para $1 \leq j \leq n$.

$\downarrow$ $\quad$ TDE#5($A[f_1, ..f_n] = \bigcup_{i=1}^{m} C_i\, [f\, i_1, ..f\, i_n]$)

$PÓS_{const}$:

Substitua a restrição de dependência existencial $\psi = (A[f_1, ..f_n] = \bigcup_{i=1}^{m} C_i\, [f\, i_1, ..f\, i_n])$ pelas seguintes restrições:

- $\psi' = (A = \bigcup_{i=1}^{m} C_i)$, com semântica de atualização :
  $SI\,(A, \psi') = SI\,(A, \psi)$
  $SI\,(C_i, \psi') = SI\,(C_i, \psi)$, para $1 \leq i \leq n$.
  $SD\,(A, \psi') = SD\,(A, \psi)$
  $SD\,(C_i, \psi') = SD\,(C_i, \psi)$, para $1 \leq i \leq n$.
- $f_i \equiv g_i$ para $1 \leq i \leq n$.

Obs.: Como todas as classes e atributos no esquema original são diretamente transformados em classes e atributos do esquema transformado, σ, ϕ e τ são mapeamentos-identidades.

Figura 4.9 - A primitiva de transformação **TDE#5**

PRE$_s$:

$A[f_1, ..f_n] = \prod_{i=1}^{m} C_i\, [f\,i_1, ..f\,i_n]$ onde

- $\{f_1, ..f_n\} \in$ *S_chaves*(A),
- $\{fk_1, ...,fk_m \in$ *S_chaves*(A), onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ..f_n\}$, sss $\{f\,i_{k1}, ..,f\,i_{kn}\} \in$ *S_chaves*(C_i), para $1 \le i \le n$.
- Se $\{fk_1, ...,fk_m\} \in$ *S_chaves*(A) onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ...,f_n\}$ então $\mathcal{R} \models \langle A([fk_1, ...,fk_m] \rightarrow f_j), C_1([f\,1_{k1}, ..,f\,1_{kn}] \rightarrow f\,1_j), ... ,$
  $C_m([f\,m_{k1}, ..,f\,m_{kn}] \rightarrow f\,m_j) \rangle$,
  para $1 \le j \le n$.

$\downarrow$ TDE#6($A[f_1, ..f_n] = \prod_{i=1}^{m} C_i\, [f\,i_1, ..f\,i_n]$)

PÓS$_{const}$:

Substitua a restrição de dependência existencial $\psi = (A[f_1, ..f_n] = \prod_{i=1}^{m} C_i\, [f\,i_1, ..f\,i_n])$ pelas seguintes restrições:

- $\psi' = (A = \prod_{i=1}^{m} C_i)$, com semântica de atualização :
  - $SI\,(A, \psi') = SI\,(A, \psi)$
  - $SI\,(C_i, \psi') = SI\,(C_i, \psi)$, para $1 \le i \le n$.
  - $SD\,(A, \psi') = SD\,(A, \psi)$
  - $SD\,(C_i, \psi') = SD\,(C_i, \psi)$, para $1 \le i \le n$.
- $f_i \equiv g_i$ para $1 \le i \le n$.

Obs.: Como todas as classes e atributos no esquema original são diretamente transformados em classes e atributos do esquema transformado, σ, ϕ e τ são mapeamentos-identidades.

Figura 4.10 - A primitiva de transformação **TDE#6**

4.3 TRANSFORMAÇÃO DE RESTRIÇÕES FUNCIONAIS DE UNIÃO EM RESTRIÇÕES DE EQUIVALÊNCIA_DE_RELACIONAMENTOS

Quando um esquema está em forma normal decomposta, podemos substituir todas as restrições de dependências funcionais de união por restrições de equivalência_de_relacionamentos usando a primitiva de transformação **TDFU** especificada nas figura 4.11. "**TDFU**" indica "transformação de restrições de dependências funcionais de união".

PRE$_s$:

$<A([f_1, ..., f_m] \rightarrow f), B([g_1, ..., g_m] \rightarrow g) >$ onde

$\{f_1, ..., f_m\} \in$ *chaves*(A) ,

$\{g_1, ..., g_m\} \in$ *chaves*(B), e

$f_i \equiv g_i$ para $1 \leq i \leq m$.

$\downarrow$ TDFU($<A([fk_1, ..., fk_m] \rightarrow f), B([gk_1, ..., gk_m] \rightarrow g) >$)

PÓS$_{const}$:

Substitua a **DFU** $<A([f_1, ..., f_m] \rightarrow f), B([g_1, ..., g_m] \rightarrow g) >$ by $f \equiv g$.

Figura 4.11 - A primitiva de transformação **TDFU**

5. PASSO DE OTIMIZAÇÃO

O objetivo do passo de otimização é modificar o esquema para reduzir redundâncias e o o tamanho do esquema. Redundâncias podem ser eliminadas através da fusão das classes e atributos. Na metodologia proposta, definimos um conjunto de primitivas de transformação que permitem que a otimização do esquema conceitual seja realizada de forma segura e eficiente. O passo de otimização é dividido em seis sub-passos:

1) Fusão das Classes Equivalentes;
2) Fusão das Classes Embutidas;
3) Fusão dos Atributos Sinônimos;
4) Remoção dos Atributos Herdados,
5) Remoção dos Atributos Deriváveis, e
6) Remoção das Classes Deriváveis,

os quais são discutidas nas próximas seções.

5.1 FUSÃO DAS CLASSES EQUIVALENTES

A fusão de classes equivalentes pode ser feita através da aplicação da primitiva de transformação ***INTEGRE_EQUIVALENTES*** especificada na figura 5.1.

PRE$_s$: **PRE$_{rest}$:**

$\boxed{C_1}$ $\boxed{C_2}$ ... $\boxed{C_n}$ $C_1 = C_2 = \ldots = C_n$

$\downarrow$ *INTEGRE_EQUIVALENTES*$(\{C_1, C_2, \ldots, C_n\}, C)$

POS$_s$: **POS$_{rest}$:**

$\boxed{C}$

- substitua toda restrição $\psi \in$ ***restrições*** (C_i), $1 \leq i \leq n$, pela restrição ψ' tal que ψ' é obtida ψ trocando C_i por C.

ϕ: Para qualquer instância $\mathcal{D}$ do esquema original, temos:
(i) $\phi(\mathcal{D})(C) = \mathcal{D}(C_i)$, $1 \leq i \leq n$.
(ii) Todas as outras classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{D}$ do esquema transformado, temos:
(i) $\sigma(\mathcal{D})(C_i) = \mathcal{D}(C)$, para $1 \leq i \leq n$.
(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

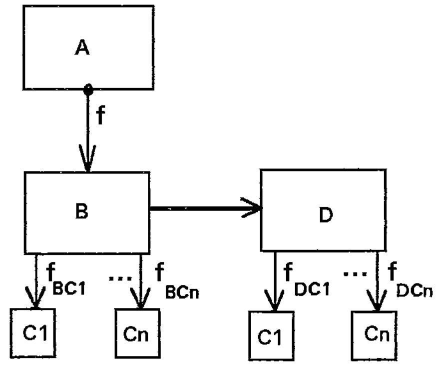
τ:
(i) τ("adicione *e* a C_i") = "adicione *e* a C", for $1 \leq i \leq n$.
(ii) τ("remova *e* in C_i") = "remova *e* de C", for $1 \leq i \leq n$.
(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figure 5.1: A primitiva de transformação ***INTEGRE_EQUIVALENTES***

5.2 FUSÃO DE CLASSES EMBUTIDAS

A fusão de classes embutidas pode ser feita através da aplicação da primitiva de transformação ***INTEGRE_EMBUTIDAS*** especificada na figura 5.2.

PRE$_S$:

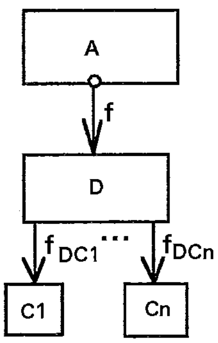


PRE$_{rest}$:

- $B \subset D$
- $f\!: A \bullet\!\!\rightarrow B$
- $f_{BC_i} \equiv f_{DC_i}$, *restrições*(f_{DC_i})|= *restrições*(f_{BC_i}), for $1 \le i \le n$.
- ***B*** é uma classe embutida e os únicos atributos e as únicas restrições não redundantes em B são estas que aparecem em PRE$_S$ and PRE$_{rest}$.

INTEGRE_EMBUTIDA (***B***, ***D***, ***f***)

POS$_S$:



POS$_{rest}$:

(i) Substitua restrição $f\!: A \rightarrow B$ pela restrição $f\!: A \rightarrow D$ com semântica de atualização:
$SI\ (A, (f\!: A \rightarrow D)) = SI\ (B, (B \subset D\ ))$
$SD\ (D, (f\!: A \rightarrow D)) = SD\ (D, (B \subset D\ ))$
$SM(f, (f\!: A \rightarrow D)) = SI\ (B, (B \subset D\ ))$

(ii) Substitua toda restrição de equivalência_de_relacionamentos ψ, tal que f_{BC_i} ocorre em ψ, por ψ' onde ψ' é obtida de ψ trocando-se f_{BC_i} por f_{DC_i}, para $1 \le i \le n$.

(iii) Remova ***B*** e todas as restrições em ***B***

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{D}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{D})(B) = \{\ b \mid b \in \mathcal{D}(D)\ \ \&\ \mathcal{D}(f^{-1})(b) \neq \emptyset\ \}$.

(ii) $\forall b \in \sigma(\mathcal{D})(B)\ \ \sigma(\mathcal{D})(f_{BC_i})(b) = \mathcal{D}(f_{DC_i})(b)$, $1 \le i \le n$;

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ: (i) ***B*** não é diretamente atualizável (***B*** é uma classe embutida).

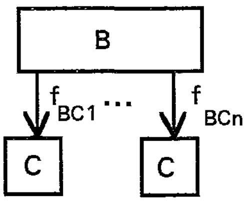
(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.2: A primitiva de transformação ***INTEGRE_EMBUTIDA***

5.3 FUSÃO DE ATRIBUTOS SINÔNIMOS

A fusão de atributos sinônimos na mesma classe pode ser feita através da aplicação da primitiva de transformação ***INTEGRE_ATRIBUTOS*** especificada na figura 5.3.

PRE_S:

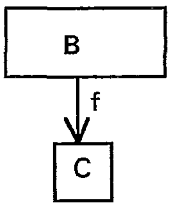


PRE_{rest}:

- $f_{BC1} \equiv f_{BC2} \equiv .... \equiv f_{BCn}$

$\downarrow$ ***INTEGRE_ATRIBUTOS*** $(\{ f_{BC1}, ... , f_{BCn} \}, f)$

$POST_S$:



$POST_{rest}$:

- Em todas as restrições nas quais f_{BCi} ocorre, substitua f_{BCi} por f, para $1 \leq i \leq n$,
- Remova atributos $f_{BC1}, ... , f_{BCn}$

ϕ: Para qualquer instância $\mathcal{D}$ do esquema original , temos:

(i) $\forall b \in \mathcal{D}(B)\ \phi(\mathcal{D})(f)(b) = \mathcal{D}(f_{BC1})(b)$

(ii) Todas as outras classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{D}$ do esquema transformado, temos:

(i) $\forall b \in \sigma(\mathcal{D})(B)\ \sigma(\mathcal{D})(f_{BCi})(b) = \mathcal{D}(f)(b)$, $1 \leq i \leq n$.

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("modifique e com ($f_{BCi} = v$)") = "modifique e com ($f = v$)", para $1 \leq i \leq n$.

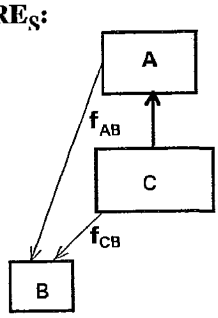
(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.3: A primitiva de transformação ***INTEGRE_ATRIBUTOS***

5.4 REMOÇÃO DE ATRIBUTOS HERDADOS

Como as entidades de uma sub-classe herdam os atributos da super-classe, um atributo da subclasse que é equivalente a um atributo da superclasse pode ser removido se ele satisfaz as pré-condições da primitiva de transformação ***REMOVA_ATRIBUTO#1*** shown in figure 5.4.

PRE$_S$:

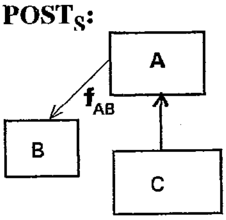


PRE$_{rest}$:

- $C \subset A$
- $f_{AB} \equiv f_{CB}$
- *restrições*$(f_{AB}) \models$ *restrições*(f_{CB})

↓ ***REMOVA_ATRIBUTO#1*** (f_{CB}, f_{AB})

POST$_S$:



POST$_{rest}$:

- Em todas as restrições nas quais f_{CB} ocorre, substitua f_{CB} por f_{AB} .
- Remova atributo f_{CB} e todas as restrições em f_{CB}.

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{S}$ do esquema transformado, temos:

(i) $\forall c \in \sigma(\mathcal{S})(C)\ \sigma(\mathcal{S})(f_{CB})(c) = \mathcal{S}(f_{AB})(c)$

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("modifique e com $(f_{CB} = v)$") = "modifique e com $(f_{AB} = v)$".

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.4: A primitiva de transformação ***REMOVA_ATRIBUTO#1***.

Em nosso enfoque, o caso em que as restrições do atributo não são implicadas logicamente pelas restrições do atributo herdado e o caso de atributos sinônimos de classes superpostas, são tratados durante a fase do projeto do esquema conceitual. As metodologias de integração de visões existentes [Batini84, Navathe86, Gottard92], sugerem que os atributos comuns de classes superpostas devem migrar para uma superclasse comum. Essa solução introduz o problema de valores nulos e dificuldades na modificação do esquema. Além disso, essa solução não é aplicável quando as classes requerem diferentes restrições para os atributos. Como mostramos no capítulo 7, esses problemas podem ser resolvidos se a integração de atributos, para esses casos especiais, é tratada somente durante o projeto do esquema lógico.

5.5 REMOÇÃO DE ATRIBUTOS DERIVÁVEIS

O objetivo desse passo de otimização é remover os atributos cujos valores são deriváveis da composição dos valores de outros atributos, de forma que as redundâncias capturadas pelas restrições de equivalência_de_relacionamentos sejam eliminadas. Atributos deriváveis podem ser removidos do esquema pela aplicação da primitiva de transformação ***REMOVA_ATRIBUTO#2*** mostrada na figura 5.5.

PRE:

- $f \in$ ***attributes***(C)
- $\psi = (f \equiv f_n \circ \dots \circ f_1)$
- ***restrições***$(f_n \circ \dots \circ f_1) \models$ ***restrições***(f)
- ***SM***$(f, \psi) =$ ***propaga***.

$\downarrow$ ***REMOVA_ATRIBUTO#2***$(f, f_n \circ \dots \circ f_1)$

POS:

(i) Em todas as restrições de equivalência_de_relacionamentos nas quais f ocorre, substitua f by $f_n \circ \dots \circ f_1$.

(ii) Remova atributo f e todas as restrições em f.

ϕ:

Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{I}$ do esquema transformado, temos:

(i) $\forall c \in \sigma(\mathcal{I})(C)\ \sigma(\mathcal{I})(f)(c) = \mathcal{I}(f_n \circ \dots \circ f_1)(c)$.

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("modifique e com $(f=v)$") = "modifique e' com $(f_n=v)$", onde $e' = f_{n-1} \circ \dots \circ f_1(e)$

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.5: A primitiva de transformação ***REMOVA_ATRIBUTO#2***

5.6 REMOÇÃO DE CLASSES DERIVÁVEIS

O objetivo desse passo de otimização é remover classes que são deriváveis de outras classes. Isso reduz o número de classes no esquema conceitual e, assim, é mais fácil verificar a existência de violações das restrições estruturais associadas com as classes deriváveis, nos casos em que essas restrições não sejam automaticamente mantidas. Existem quatro tipos de classes deriváveis, definidas a seguir. Para definições 5.1 à 5.4 suponha que $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ é um esquema e C é uma classe em $\mathcal{C}$.

<u>Definição 5.1</u>: C é *derivável por seleção de* B sss existe $f \in attributes(B)$ tal que $\mathcal{R} \models (C = B[f \neq \text{"nulo"}])$ ou $\mathcal{R} \models (C = B[f = \text{"sim"}])$[1], e todas as outras restrições em C são redundantes.

<u>Definição 5.2</u>: C é *derivável de união* sss $\mathcal{R} \models (C = \bigcup_{i=1}^{m} C_i)$ onde $C_1, \ldots, C_m \in \mathcal{C}$, e todas as outras restrições em C são redundantes.

<u>Definição 5.3</u>: C é *derivável de interseção* sss $\mathcal{R} \models (C = \bigcap_{i=1}^{m} C_i)$ onde $C_1, \ldots, C_m \in \mathcal{C}$, e todas as outras restrições em C são redundantes.

<u>Definição 5.4</u>: C é *derivável de diferença* sss $\mathcal{R} \models (C = A - B)$ onde A e $B \in \mathcal{C}$, e todas as outras restrições em C são redundantes.

A remoção de classes deriváveis é dividade em quatro sub-passos:

1) Remoção de classes deriváveis de seleção;
2) Remoção de classes deriváveis de união;
3) Remoção de classes deriváveis de intersecção, e,
4) Remoção de classes deriváveis de diferença,

que são discutidos nas seções seguintes.

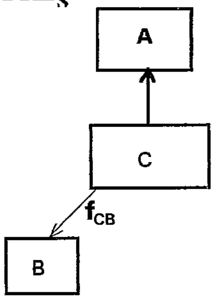
5.6.1 REMOÇÃO DE CLASSES DERIVÁVEIS DE SELEÇÃO

As primitivas de tranformação, especificadas nas figuras 5.6 e 5.7, podem ser utilizadas para transformar uma classe não derivável em uma classe derivável de seleção. A primitiva de transformação ***ADICIONE_SELEÇÃO#1***, especificada na figura 5.6, pode ser utilizada quando uma sub-classe tem um atributo que é aplicável somente para as entidades dessa subclasse. A primitiva de transformação ***ADICIONE_SELEÇÃO#2***, especificada na figura 5.7, pode ser utilizada quando uma sub-classe não tem nenhum atributo que seja aplicável somente para as entidades da subclasse. Nesse caso, adiciona-se à super-classe um novo atributo, o qual tem o

[1]Novos tipos de classes deriváveis de seleção podem ser adicionadas.

valor "sim" se a entidade é um membro da sub-classe e o valor "nulo" caso contrário.

PRE$_S$:

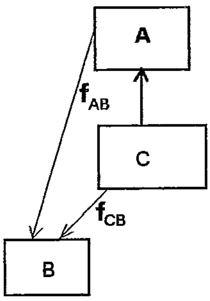


PRE$_{rest}$:

- $C \subset A$
- $f_{CB} \in$ ***attributes*** (C)
- ***Nulo*** não é permitido para f_{CB}
- Para qualquer classe D em $\mathcal{C}$ onde $D \in$ ***subclasses*** (A), $D \neq C$ e $(\neg D \mid C)$, não existe nenhum $g \in$ ***atributos*** (D) tal que $g \equiv f_{CB}$

$\downarrow$ *ADICIONE_SELEÇÃO#1*(f_{CB}, C, A, f_{AB})

POS$_S$:



POS$_{rest}$:

(i) Adicione restrições:
$f_{AB} \equiv f_{CB}$
f_{AB} tem todas as restrições de f_{CB} com exceção de que:
 Nulo é permitido para f_{AB} , e
 $\boldsymbol{SD\ (B,\ (f_{AB}:A \to B)\ )}$ =
  a) ***nulifica***, se $\boldsymbol{SD\ (B,\ (f_{CB}:C \to B))}$ = ***propaga***, ou
  b) ***rejeita***, caso contrário.

(ii) Substitua a restrição $C \subset A$ por $C = A[f_{AB} \neq$ "***nulo***"] com semântica de atualização:
 SI $(C, (C = A[f_{AB} \neq$ "***nulo***"$])) =$ ***SI*** $(C, (C \subset A))$
 SD $(A, (C = A[f_{AB} \neq$ "***nulo***"$])) =$ ***SD*** $(A, (C \subset A))$
 SD $(C, (C = A[f_{AB} \neq$ "***nulo***"$]))=$ ***nulifica***
 SM $(f_{AB}, (C = A[f_{AB} \neq$ "***nulo***"$]))=$ ***propaga***.

ϕ: Para qualquer instância $\mathcal{D}$ do esquema original, temos:

(i) $\forall a \in \phi(\mathcal{D})(A)\ \phi(\mathcal{D})(f_{AB})(a) =$
 $\mathcal{D}(f_{CB})(a)$, se $a \in \mathcal{D}(C)$
 nulo, caso contrário.

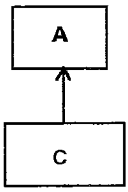
(ii) Todas as outras classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Todas as classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ: Todas as classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.6: A primitiva de tranformação *ADICIONE_SELEÇÃO#1*

PRE$_S$:

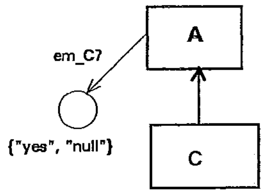


PRE$_{rest}$:

- $C \subset A$

ADICIONE_SELEÇÃO#2 (*em_C?*, *C*, *A*)

POS$_S$:



POS$_{const}$:

(i) Adiciona atributo *em_c?*:*A* $\rightarrow$\{ "sim", "nulo"\}

(ii) Substitua a restrição $C \subset A$ por $C = A[em\_c? = \text{"sim"}]$ com semântica de atualização:

$SI\ (C, (C = A[em\_c? = \text{"sim"}])) = SI\ (C, (C \subset A))$

$SD\ (A, (C = A[em\_c? = \text{"sim"}])) = SD\ (A, (C \subset A))$

$SD\ (C, (C = A[em\_c? = \text{"sim"}])) =$ ***nulifica***

$SM\ (em\_c?\ , (C = A[em\_c? = \text{"sim"}])) =$ ***propaga.***

ϕ: Para qualquer instância $\mathcal{D}$ do esquema original, temos:

(i) $\forall a \in \phi(\mathcal{D})(A)\ \phi(\mathcal{D})(em\_c?)\,(a) =$

"***sim***", se $a \in \mathcal{D}(C)$

"***nulo***", caso contrário.

(ii) Todas as outras classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

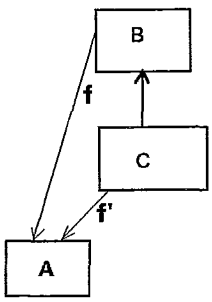
σ: Todas as classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ: Todas as classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.7: A primitiva de transformação ***ADICIONE_SELEÇÃO#2***

É possível elaborar um algorítimo para transformar um esquema, criando tantas classes deriváveis de seleção quantas forem possíveis, como é feito em [Casanova93, Markowitz90]. Porém, existem situações em que a remoção de uma classe derivável não é desejável. Assim sendo, no nosso enfoque, nós deixamos o projetista decidir (baseado em vários critérios de desempenho) se uma classe deve ser removida do esquema conceitual. Ao invés de um algorítimo, nossa metodologia fornece as primitivas de transformação ***ADICIONE_SELEÇÃO#1*** e ***ADICIONE_SELEÇÃO#2*** que permitem o projetista transformar classes não deriváveis em classes deriváveis de seleção de forma simples e segura. Classes deriváveis de seleção podem ser removidas do esquema pela aplicação das primitivas de transformação ***REMOVA_CLASSE#1*** and ***REMOVA_CLASSE#2*** mostradas nas figuras 5.8 e 5.9.

PRE$_S$:

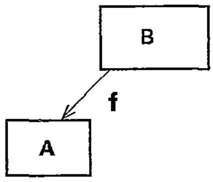


PRE$_{rest}$:

- $C = B[f \neq$"*nulo*"$]$.
- $f' \in$ *attributes*(C).
- $f' \equiv f$ e ***nulo*** não é permitido para f'.
- Qualquer outra restrição em C é redundante (*atributos*$(C) = \{f'\}$).
- $SD(C,\ C = B[f \neq$"*nulo*"$]) =$ ***nulifica.***
- $SD(B,\ C = B[f \neq$"*nulo*"$]) =$ ***propaga.***

$\downarrow$ ***REMOVA_CLASSE#1***$(C\ ,\ C = B[f \neq$"*nulo*"$])$

POS$_S$:



POS$_{const}$:

(i) Remova C e todas as restrições em C
(ii) Remova f' e todas as restrições em f'.

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{D}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{D})(C) = \{\ b \mid b \in \mathcal{D}(B)\ \ \&\ \mathcal{D}(f)(b) \neq$"***nulo***"$\}$.

(ii) $\forall c \in \sigma(\mathcal{D})(C)\ \ \sigma(\mathcal{D})(f')(c) = \mathcal{D}(f)(c)$).

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("remova *e* de C") = "modifique *e* com (f=***nulo***)".

(ii) τ("modifique *e* com (f'=v)") = "modifique *e* com (f=v)".

(iii) τ("adicione *e*(..., f'=v, ...) em C ") =

{ <u>se</u> v=***nulo*** <u>então</u> "aborte a presente transação";

{ <u>no caso de:</u>

[$SI(C,\ C = B[f \neq$"***nulo***"$]) =$ ***rejeita***] :

{se $e \in \mathcal{D}(B)$ <u>então</u> "modifique *e* com (f=v)"

<u>senão</u> "aborte a presente transação"};

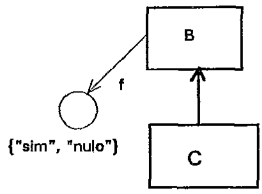
[caso-contrário]:

{ se $e \notin \mathcal{D}(B)$ "adicione *e* em B"; "modifique *e* com (f=v)" }}

(iv) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figure 5.8: A primitiva de transformação ***REMOVA_CLASSE#1***

PRE$_S$:

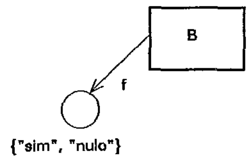


PRE$_{rest}$:

- $C = B[f = \text{"}sim\text{"}]$.
- Qualquer outra restrição em C é redundante (*atributos*$(C) = \varnothing$).
- $SD(C,\ C = B[f = \text{"}sim\text{"}]) = nulifica$.
- $SD(B,\ C = B[f = \text{"}sim\text{"}]) = propaga$.

$\downarrow$ *REMOVA_CLASSE#2*$(C\ ,\ C = B[f = \text{"}sim\text{"}]\ )$

POS$_S$:



POS$_{const}$:

- Remova C e todas as restrições em C

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{B}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{B})(C) = \{\ c \mid c \in \mathcal{B}(C)\ \ \&\ \mathcal{B}(f)(c) = \text{"}sim\text{"}\}$.

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("remova *e* de *C*") = "modifique *e* com (f="*nulo*").

(ii) τ("adicione *e* em *C* ") =

{ <u>no caso de:</u>

[$SI(C,\ C = B[f = \text{"}sim\text{"}]\ ) = rejeita$]:

{se $e \in \mathcal{B}(B)$ <u>então</u> "modifique *e* com (f="*sim*")"

<u>senão</u> "aborte a presente transação"};

[caso-contrário]:

{ se $e \notin \mathcal{B}(B)$ "adicione *e* em ***B***"; "modifique *e* com (f="*sim*")" }}

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figure 5.9: A primitiva de transformação ***REMOVA_CLASSE#2***

5.6.2 REMOÇÃO DAS CLASSE DERIVÁVEIS DE UNIÃO

Classes deriváveis de união podem ser removidas do esquema através da aplicação da primitiva de transformação ***REMOVA_CLASSE#3*** mostrada na figura 5.10.

PRE:

- $\psi = C = \bigcup_{i=1}^{m} C_i$
- $SI(C, \psi)$ = propaga para C_k, para algum $1 \leq k \leq m$.
- $SD(C, \psi)$ = ***propaga***.
- Qualquer outra restrição em C é redundante (cada atributo f de C tem a ele associado um atributo g em C_i, para $1 \leq i \leq n$, tal que $f \equiv g$ e *restrições*(g)|=*restrições*(f)).

$$\downarrow \quad REMOVA\_CLASSE\#3(\, C, \;\; C = \bigcup_{i=1}^{m} C_i \,)$$

POST:

(i) Substitua os atributos de C pelos atributos equivalentes em C_i, $1 \leq i \leq m$, para todas as restrições
de equivalência_de_atributos.

(ii) Remova C e todas as restrições em C

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{B}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{B})(C) = \bigcup_{i=1}^{m} \mathcal{B}(C_i)$.

(ii) Para qualquer atributo f de C, temos:

$\forall c \in \sigma(\mathcal{B})\ \sigma(\mathcal{B})(f)(c) = \mathcal{B}(g)(c))$, onde g é a transformação direta de algum atributo g em C_i, $1 \leq i \leq n$, tal que $f \equiv g$.

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("remova e de C") = { Se $e \in \mathcal{B}(C_i)$ então "remova e de C_i", para $1 \leq i \leq n$ }

(ii) τ("adicione e a C") = "adicione e a C_k".

(iii) Para qualquer atributo f de C, tal que $f \equiv g$ onde g é um atributo em C_i, para algum $1 \leq i \leq n$, temos:

τ("modifique e com (f=v)") = "modifique e com (g=v)".

(iv) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.10: A primitiva de transformação ***REMOVA_CLASSE#3***

5.6.3 REMOÇÃO DE CLASSES DERIVÁVEIS DE INTERSEÇÃO

Classes deriváveis de interseção podem ser removidas do esquema através da aplicação da primitiva de transformação ***REMOVA_CLASSE#4*** mostrada na figura 5.11.

PRE:

- $\psi = C = \bigcap_{i=1}^{m} C_i$
- $SD(C, \psi)$= propaga para C_k, para algum $1 \leq k \leq m$.
- $SI(C, \psi)$= ***propaga***.
- Qualquer outra restrição em C é redundante (***atributos***$(C) = \emptyset$)

$\Big\downarrow$ ***REMOVA_CLASSE#4*** C, $C = \bigcap_{i=1}^{m} C_i$)

POST:

- Remova C e todas as restrições em C

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{D}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{D})(C) = \bigcap_{i=1}^{m} \mathcal{D}(C_i)$.

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("remova e de C") = "remova e de C_k".

(ii) τ("adicione e a C") = { para todo $1 \leq i \leq n$, se $e \notin \mathcal{D}(C_i)$ então "adicione e a C_i"}.

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.11: A primitiva de transformação ***REMOVA_CLASSE#4***

5.6.4 REMOÇÃO DAS CLASSES DERIVÁVEIS DE DIFERENÇA

Classes deriváveis de diferença podem ser removidas do esquema através da aplicação da primitiva de transformação ***REMOVA_CLASSE#4*** mostrada na figura 5.12.

PRE:

- $\psi = (C = A - B)$
- $SD(C, \psi) =$ ***propaga.***
- $SI(C, \psi) =$ ***propaga.***
- Qualquer outra restrição em C é redundante (***atributos***$(C) = \emptyset$)

$\downarrow$ *REMOVA_CLASSE#5*(C, $C = A - B$)

POST:

- Remova C e todas as restrições em C

ϕ: Todas as classes e atributos no esquema transformado são transformações diretas de classes e atributos do esquema original.

σ: Para qualquer instância $\mathcal{J}$ do esquema transformado, temos:

(i) $\sigma(\mathcal{J})(C) = \mathcal{J}(A) - \mathcal{J}(B)$

(ii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

τ:

(i) τ("remova e de C") = "remova e de A".

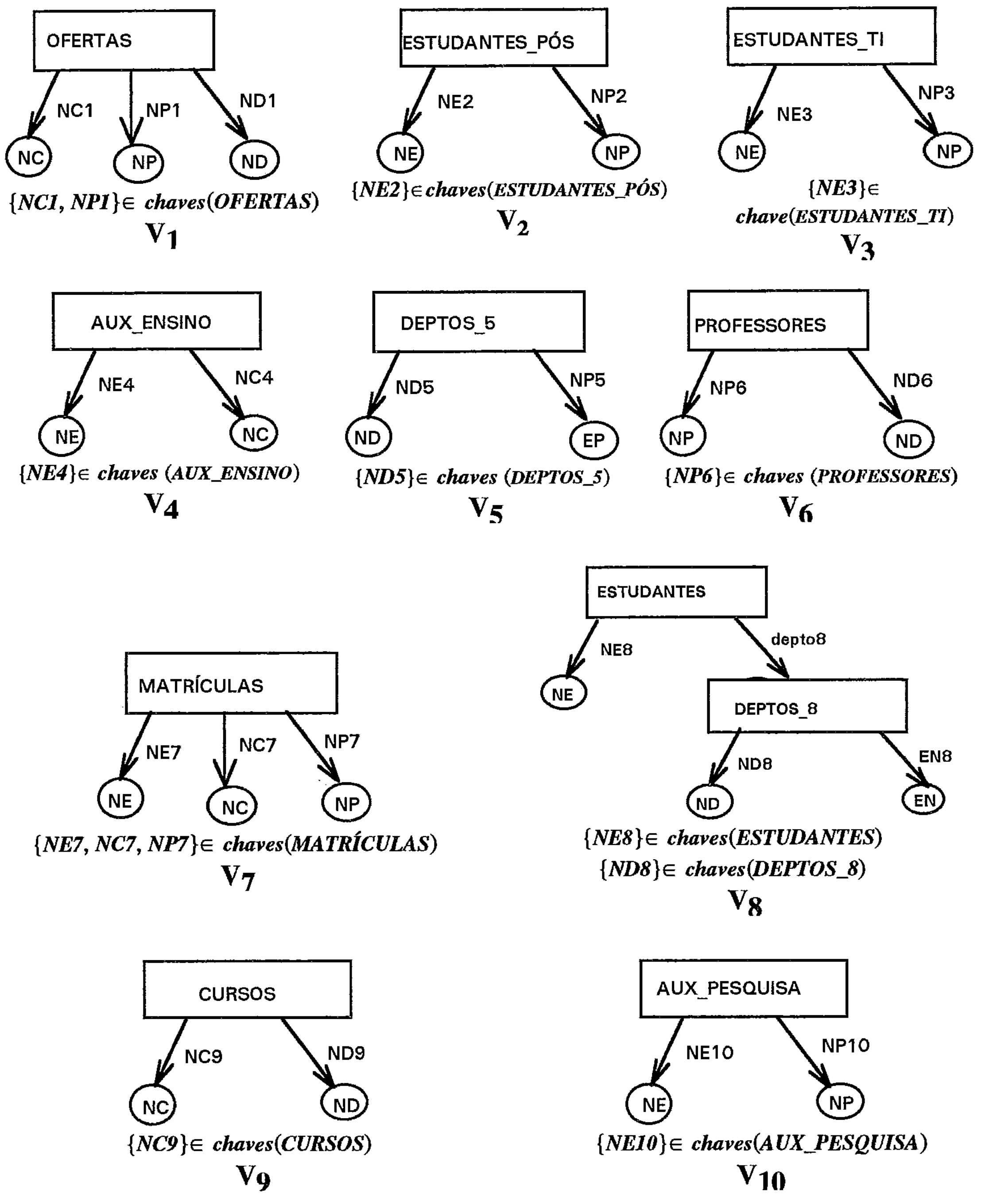
(ii) τ("adicione e a C") = { <u>no caso de</u>:
[$e \notin \mathcal{J}(B)$]: "adicione e a A";
[$e \in \mathcal{J}(B)$ e $e \in \mathcal{J}(A)$]: "remova e de B";
[$e \in \mathcal{J}(B)$ e $e \notin \mathcal{J}(A)$]: {"remova e de B"; "adicione e a A"} }.

(iii) Todas as outras classes e atributos no esquema original são transformados diretamente em classes e atributos do esquema transformado.

Figura 5.12: A primitiva de transformação ***REMOVA_CLASSE#5***

6. UM EXEMPLO

Neste capítulo, nós utilizaremos a metodologia proposta para integrarmos as visões V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9 e V_{10} na figura 6.1. Nas seções 6.1, 6.2, e 6.3 apresentamos os passos de combinação, reestruturação e otimização para a obtenção do esquema conceitual global. Na seção 6.4 nós definimos as visões V_1 , ..., V_{10} em termos do esquema conceitual global obtido.



Abreviações: **NC**= Número do Curso, **NP**= Nome do Professor, **ND**= Nome do Departamento, **NE**= Nome do Estudante, **EN**= Endereço.

Figura 6.1: As visões V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9 e V_{10}

6.1 PASSO#1: COMBINAÇÃO DAS VISÕES

Combinando as visões V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9 e V_{10} nós obtemos o esquema combinado *EC* que contém todas visões V_1, ...,V_{10} mais as restrições da figura abaixo:

Restrições Estruturais

RE1: ***ESTUDANTES_PÓS*** ⊂ ***ESTUDANTES***,
com semântica de atualização:
SI(***ESTUDANTES_PÓS***, *RE1*) = ***propaga***
SD(***ESTUDANTES***, *RE1*) = ***propaga***

RE2: ***ESTUDANTES_TI*** ⊂ ***ESTUDANTES***,
com semântica de atualização:
SI(***ESTUDANTES_TI***, *RE2*) = ***propaga***
SD(***ESTUDANTES***, *RE2*) = ***propaga***

RE3: ***AUX_ENSINO*** ⊂ ***ESTUDANTES_PÓS***,
com semântica de atualização:
SI(***AUX_ENSINO***, *RE3*) = ***rejeita***
SD(***ESTUDANTES_PÓS***, *RE3*) = ***propaga***

RE4: ***AUX_PESQUISA*** ⊂ ***ESTUDANTES_PÓS***,
com semântica de atualização:
SI(***AUX_PESQUISA***, *RE3*) = ***rejeita***
SD(***ESTUDANTES_PÓS***, *RE3*) = ***propaga***

RE5: ***DEPTOS8*** = ***DEPARTAMENTOS***,

Restrições de equiv. de relacionamentos:

ER1: ***NE2*** ≡***NE3***≡***NE4***≡***NE8***≡***NE10*** *ER2*: ***ND8***≡***ND5*** *ER3*: ***NP2***≡***NP3***≡***NP10***

Restrições referenciais:

RR1:***depto8***:***ESTUDANTES***→***DEPTOS_8*** com semântica de atualização:
SI(***ESTUDANTES***, *RR1*) = ***rejeita***, ***SD***(***DEPTOS_8***, *RR1*) = ***propaga***, ***SM***(***depto8***, *RR1*) = ***rejeita***

Restrições de dependências funcionais :

FD1:***OFERTAS***([***NC1***]→***ND1***) *FD2*:***OFERTAS***([***NP1***]→***ND1***)

Restrições de dependências existenciais

ED1:***OFERTAS***[***NC1***, ***ND1***]⊂***CURSOS***[***NC9***, ***ND9***]
com semântica de atualização:
SI(***OFERTAS***, *ED1*) = ***rejeita***
SD(***CURSOS***, *ED1*) = ***propaga***

ED2:***OFERTAS***[***NP1***,***ND1***]⊂***PROFESSORES***[***NP6***, ***ND6***]
com semântica de atualização:
SI(***OFERTAS***, *ED2*) = ***rejeita***
SD(***PROFESSORES***, *ED2*) = ***propaga***

ED3:***ESTUDANTES_PÓS***[***NP2***]⊂***PROFESSORES***[***NP6***]
com semântica de atualização:
SI(***ESTUDANTES_PÓS***, *ED3*) = ***rejeita***
SD(***PROFESSORES***, *ED3*) = ***rejeita***

ED4:***ESTUDANTES_TI***[***NP3***]⊂***PROFESSORES***[***NP6***]
com semântica de atualização:
SI(***ESTUDANTES_TI***, *ED4*) = ***rejeita***
SD(***PROFESSORES***, *ED4*) = ***rejeita***

ED5:***AUX_ENSINO***[***NC4***]⊂***CURSOS***[***NC9***]
com semântica de atualização:
SI(***AUX_ENSINO***, *ED5*) = ***rejeita***
SD(***CURSOS***, *ED5*) = ***propaga***

ED6:***DEPTOS_5***[***NP5***]⊂***PROFESSORES***[***NP6***]
com semântica de atualização:
SI(***DEPARTAMENTOS5***, *ED6*) = ***rejeita***
SD(***PROFESSORES***, *ED6*) = ***rejeita***

ED7:***PROFESSORES***[***ND6***]⊂***DEPTOS_5***[***ND5***]
com semântica de atualização:
SI(***PROFESSORES***, *E7*) = ***rejeita***
SD(***DEPARTAMENTOS***, *ED7*) = ***propaga***

ED8:***MATRÍCULAS***[***NC7***,***NP7***]⊂
OFERTAS[***NC1***,***NP1***]
com semântica de atualização:
SI(***MATRÍCULAS***, *ED8*) = ***rejeita***
SD(***OFERTAS***, *ED8*) = ***propaga***

ED9:***MATRÍCULAS***[***NE7***]⊂***ESTUDANTES***[***NE8***]
com semântica de atualização:
SI(***MATRÍCULAS***, *ED9*) = ***rejeita***
SD(***ESTUDANTES***, *ED9*) = ***propaga***

ED10:***CURSOS***[***ND9***]⊂***DEPTOS_5***[***ND5***]
com semântica de atualização:
SI(***CURSOS***, *ED10*) = ***rejeita***
SD(***DEPARTAMENTOS***, *ED10*) = ***propaga***

ED11:***AUX_PESQUISA***[***NP10***]⊂
PROFESSORES[***NP6***]
com semântica de atualização:
SI(***AUX_ENSINO***, *ED11*) = ***rejeita***
SD(***PROFESSORES***, *ED11*) = ***rejeita***

Restrições de dependências funcionais de união:

UFD1:<***CURSOS***([***NC9***]→***ND9***),***OFERTAS***([***NC1***]→***ND1***)>
UFD2:<***PROFESSORES***([***NP6***]→***ND6***),***OFERTAS***([***NP1***]→***ND1***)>

Figura 6.2: Restrições de Integridade do esquema *EC*

6.2 PASSO#2: REESTRUTURAÇÃO DAS VISÕES

A seguir discutiremos cada passo da reestrutução do esquema ***EC***.

PASSO 2.1: Normalização

Neste passo o esquema ***EC*** será transformado em um esquema equivalente que está em forma normal decomposta. Para isto, de acordo com o algoritmo 4.3, todas as classes em ***CS*** devem ser normalizadas (i.e, colocadas em FND). Para normalizar uma classe ***C***, determina-se primeiro o grafo de decomposição D_C de ***C*** usando o algorithm 4.2, e então obtém-se a transformação definida por D_C usando o algorítimo 4.1. A seguir discutiremos a normalização de cada classe do esquema ***EC***.

a) Normalização da classe ***OFERTAS***

- O grafo de decomposição da classe ***OFERTAS*** obtido com o algorítimo 4.2 é mostrado na figura 6.3(a). De acordo com o algorítimo 4.1, o grafo de decomposição na figura 6.3 (a) define a seguinte transformação:

$T = T_2 \circ T_1$, onde

T_1= ***Decomponha(OFERTAS, {NC1, ND1}, CURSOS_1, curso1 ,{NC1', ND1'})***, e

T_2= ***Decomponha(OFERTAS, {NP1, ND1}, PROF_1, professor1, {NP1', ND1''})***.

b) Normalização da classe ***ESTUDANTES_PÓS***

- O grafo de decomposição da classes ***ESTUDANTES_PÓS*** na figura 6.3 (b) define a seguinte transformação:

T_3 = ***Decomponha(ESTUDANTES_PÓS, {NP2}, PROF_2, orientador2,{NP2'})***.

c) Normalização da classe ***ESTUDANTES_TI***

- O grafo de decomposição da classes ***ESTUDANTES_TI*** na figura 6.3 (c) define a seguinte transformação:

T_4 = ***Decomponha(ESTUDANTES_TI, {NP3}, PROF_3, orientador3,{NP3'})***.

d) Normalização da classe ***AUX_ENSINO***

- O grafo de decomposição da classes ***AUX_ENSINO*** na figura 6.3 (d) define a seguinte transformação:

T_5 = ***Decomponha(AUX_ENSINO, {NC4}, CURSOS_4, curso4,{NC4'})***.

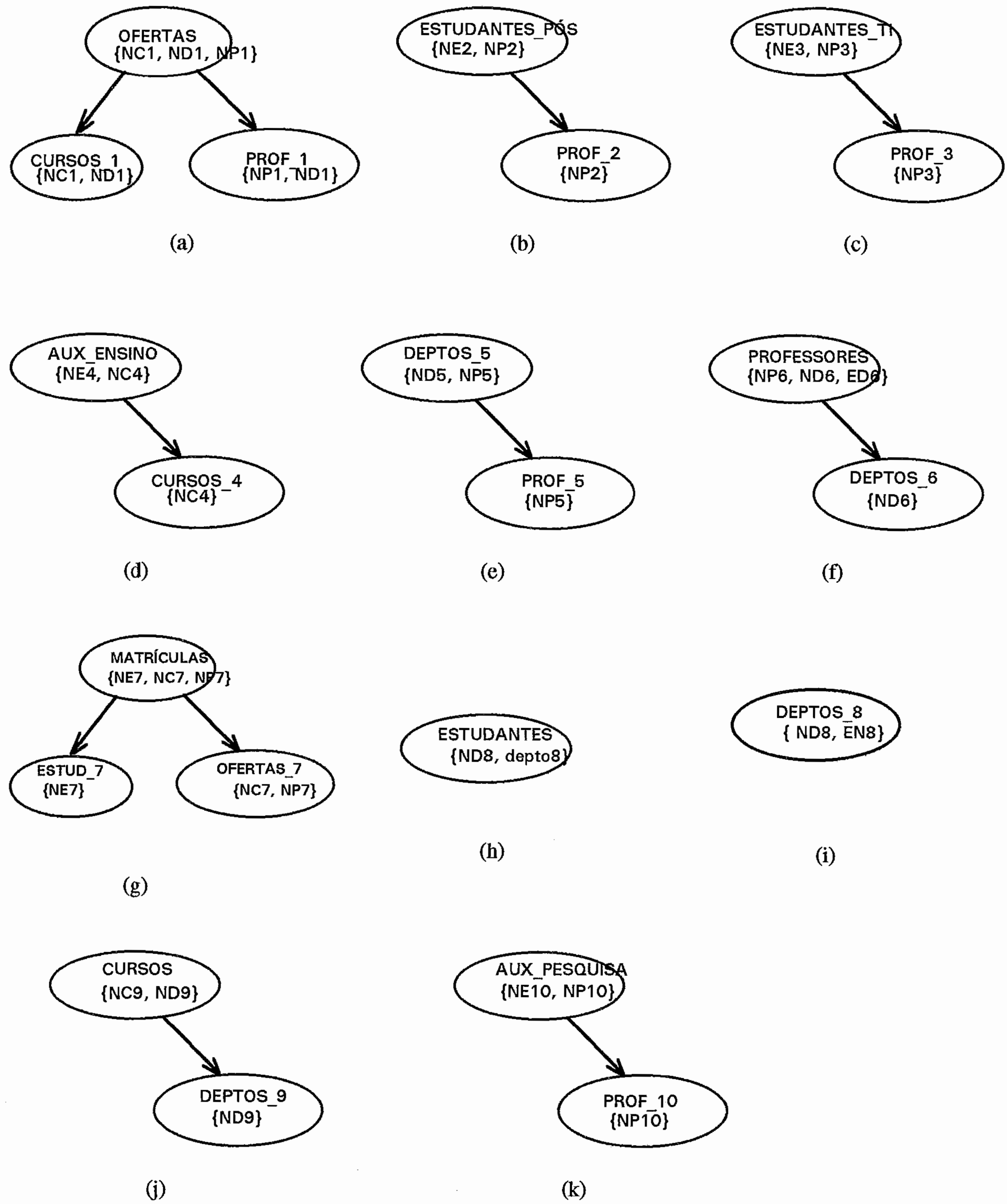


Figura 6. 3: Grafos de decomposição das classes do esquema *EC*

e) Normalização da classe ***DEPTOS_5***

- O grafo de decomposição da classes ***DEPTOS_5*** na figura 6.3 (e) define a seguinte transformação:

$T_6 =$ ***Decomponha** (DEPTOS_5,* {*NP5*}, ***PROF_5, chefe***,{*NP5'*}).

f) Normalização da classe ***PROFESSORES***

- O grafo de decomposição da classe ***PROFESSORES*** na figura 6.3 (f) define a seguinte transformação:

$T_7 =$ ***Decomponha** (PROFESSORES,* {*ND6*}, ***DEPTOS_6, depto6***,{*ND6'*}).

g) Normalização da classe ***MATRÍCULAS***

- O grafo de decomposição da classe ***MATRÍCULAS*** na figura 6.3 (g) define a seguinte transformação:

$T = T_8 \circ T_9$, onde

T_8= ***Decomponha**(MATRÍCULAS,* {*NE7*}, ***ESTUD_7, estudante7*** ,{*NE7'*}), e

T_9=***Decomponha**(MATRÍCULAS,*{*NC7,NP7*},***OFERTAS_7,oferta7***,{*NC7',NP7'*}).

h) Normalização da classe ***ESTUDANTES***

- O grafo de decomposição da classe ***ESTUDANTES*** na figura 6.3(h) indica que ***ESTUDANTES*** já se encontra em FND.

i) Normalização da classe ***DEPTOS_8***

- O grafo de decomposição da classe ***DEPTOS_8*** na figura 6.3(i) indica que ***DEPTOS_8*** já se encontra em FND.

j) Normalização da classe ***CURSOS***

- O grafo de decomposição da classe ***CURSOS*** na figura 6.3 (j) define a seguinte transformação:

$T_{10} =$ ***Decomponha** (CURSOS,* {*ND9*}, ***DEPTOS_9, depto9***,{*ND9'*}).

k) Normalização da classe ***AUX_ENSINO***

- O grafo de decomposição da classe ***AUX_ENSINO*** na figura 6.3 (k) define a seguinte transformação:

$T_{11} =$ ***Decomponha**(AUX_ENSINO,* {*NP10*}, ***PROF_10, orientador10***,{*NP10'*}).

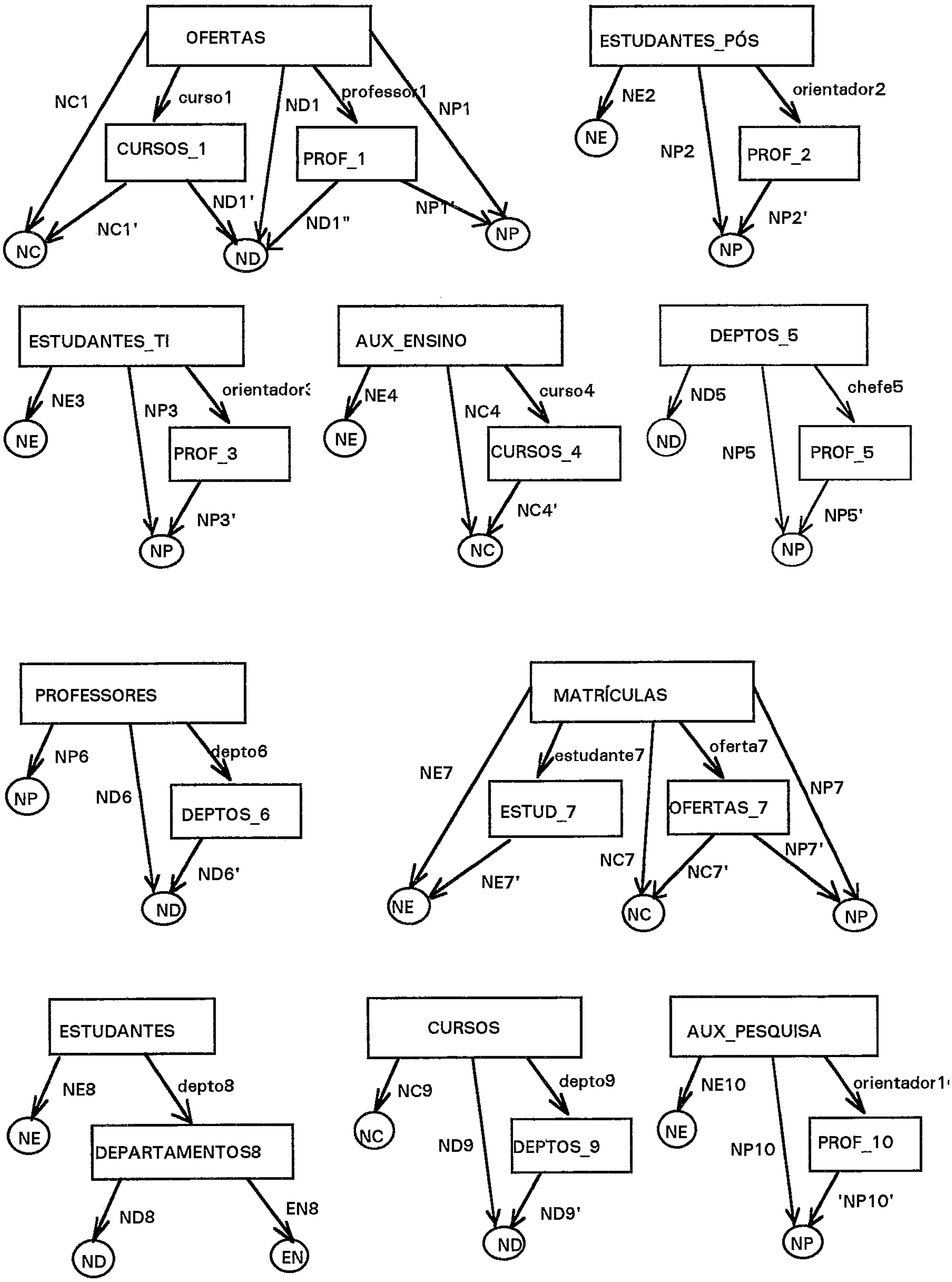


Figura 6.4: O Esquema Normalizado ***EN***

Aplicando as transformações T_1, ..., T_{11} no esquema ***EC*** temos:
Seja ***EN*** = $T_{11} \circ \ldots \circ T_1$(***EC***). A estrutura do esquema ***EN*** (esquema normalizado) é mostrada na figura 6.4. As novas restrições de ***EN*** são obtidas pelas seguintes transformações de restrições:

a) Transformações de restrições resultantes da transformação T_1

(i) Adicione as seguintes restrições:

- ***curso1:OFERTAS→CURSOS_1*** com semântica de atualização:
  ***SI (OFERTAS, (curso1:OFERTAS→CURSOS_1))* = propaga;**
  ***SD (CURSOS_1, (curso1:OFERTAS→CURSOS_1))* = propaga;**
- ***curso1:OFERTAS•→CURSOS_1*** com semântica de atualização:
  ***SI (CURSOS_1, (curso1:OFERTAS•→CURSOS_1))* = propaga;**
  ***SD (OFERTAS, (curso1:OFERTAS•→CURSOS_1)* = propaga;**
- ***NC1 ≡ curso1 ∘ NC1'; restrições(NC1')=restrições(NC1);***
- ***ND1 ≡ curso1 ∘ ND1'; restrições(ND1')=restrições(ND1);***

(ii) Substitua a FD ***OFERTAS([NC1]→ND1)>*** pela FD ***CURSOS_1([NC1']→ND1');***

(iii) Substitua a UFD ***<CURSOS([NC9]→ND9),OFERTAS([NC1]→ND1)>*** pela UFD ***<CURSOS([NC9]→ND9), CURSOS_1([NC1']→ND1') >***

(iv) Substitua a ED ***OFERTAS[NC1, ND1]⊂CURSOS[NC9, ND9]*** pela ED
ψ = ***CURSOS_1[NC1', ND1']⊂CURSOS[NC9, ND9]*** com semântica de atualização:
SI(CURSOS_1 , ψ) = rejeita
SD(CURSOS, ψ) = propaga

b) Transformações de restrições resultantes da transformação T_2

(i) Adicione as seguintes restrições:

- ***professor1:OFERTAS→PROF_1*** com semântica de atualização:
  ***SI (OFERTAS, (professor1:OFERTAS→PROF_1))* = propaga;**
  ***SD (PROF_1, (professor1:OFERTAS→PROF_1))* = propaga;**
- ***professor1:OFERTAS•→PROF_1*** com semântica de atualização:
  ***SI (PROF_1, (professor1:OFERTAS•→PROF_1))* = propaga;**
  ***SD (OFERTAS, (professor1:OFERTAS•→PROF_1)* = propaga;**
- ***NP1 ≡ professor1 ∘ NP1'; restrições(NP1')=restrições(NP1);***
- ***ND1 ≡ professor1 ∘ ND1''; restrições(ND1'')=restrições(ND1);***

(ii) Substitua a FD ***OFERTAS([NP1]→ND1)>*** pela FD ***PROF_1([NP1']→ND1'');***

(iii) Substitua a UFD ***<PROFESSORES([NP6]→ND6), OFERTAS([NP1]→ND1)>*** pela UFD ***<PROFESSORES([NP6]→ND6), PROF_1([NP1']→ND1'') >***

(iv) Substitua a ED ***OFERTAS[NP1, ND1]⊂PROFESSORES[NP6, ND6]*** pela ED
ψ = ***PROF_1[NP1', ND1'']⊂PROFESSORES[NP6, ND6]*** com semântica de atualização:
SI(PROF_1 , ψ) = rejeita
SD(PROFESSORES, ψ) = propaga

c) Transformações de restrições resultantes da transformação T_3

(i) Adicione as seguintes restrições:

- ***orientador2:ESTUDANTES_PÓS→PROF_2*** com semântica de atualização:
  SI (ESTUDANTES_PÓS, (orientador2:ESTUDANTES_PÓS→PROF_2)) = propaga;
  SD (PROF_2, (orientador2:ESTUDANTES_PÓS→PROF_2)) = propaga;
- ***orientador2:ESTUDANTES_PÓS•→PROF_2*** com semântica de atualização:
  SI (PROF_2, (orientador2:ESTUDANTES_PÓS•→PROF_2)) = propaga;
  SD (ESTUDANTES_PÓS, (orientador2:ESTUDANTES_PÓS•→PROF_2) = propaga;
- ***NP2*** ≡ ***orientador2***∘ ***NP2'***; ***restrições***(***NP2'***)=***restrições***(***NP2***);

(ii) Substitua a ED ***ESTUDANTES_PÓS***[***NP2***]⊂***PROFESSORES***[***NP6***] pela ED
ψ = ***PROF_2***[***NP2'***]⊂***PROFESSORES***[***NP6***] com semântica de atualização:
SI(PROF_2 , ψ) = ***rejeita***
SD(COURSES, ψ) = ***rejeita***

d) Transformações de restrições resultantes da transformação T_4

(i) Adicione as seguintes restrições:

- ***orientador3:ESTUDANTES_TI→PROF_3*** com semântica de atualização:
  SI (ESTUDANTES_TI, (orientador3:ESTUDANTES_TI→PROF_3)) = propaga;
  SD (PROF_3, (orientador3:ESTUDANTES_TI→PROF_3)) = propaga;
- ***orientador3:ESTUDANTES_TI•→PROF_3*** com semântica de atualização:
  SI (PROF_3, (orientador3:ESTUDANTES_TI•→PROF_3)) = propaga;
  SD (ESTUDANTES_TI, (orientador3:ESTUDANTES_TI•→PROF_3) = propaga;
- ***NP3*** ≡ ***orientador3***∘ ***NP3'***; ***restrições***(***NP3'***)=***restrições***(***NP3***);

(ii) Substitua a ED ***ESTUDANTES_TI***[***NP3***]⊂***PROFESSORES***[***NP6***] pela ED
ψ = ***PROF_3***[***NP3'***]⊂***PROFESSORES***[***NP6***] com semântica de atualização:
SI(PROF_3 , ψ) = ***rejeita***
SD(PROFESSORES, ψ) = ***rejeita***

e) Transformações de restrições resultantes da transformação T_5

(i) Adicione as seguintes restrições:

- ***curso4:AUX_ENSINO→CURSOS_4*** com semântica de atualização:
  SI (AUX_ENSINO, (curso4:AUX_ENSINO→CURSOS_4)) = propaga;
  SD (CURSOS_4, (curso4:AUX_ENSINO→CURSOS_4)) = propaga;
- ***curso4:AUX_ENSINO•→CURSOS_4*** com semântica de atualização:
  SI (CURSOS_4, (curso4:AUX_ENSINO•→CURSOS_4)) = propaga;
  SD (AUX_ENSINO, (curso4:AUX_ENSINO•→CURSOS_4) = propaga;
- ***NC4*** ≡ ***curso4***∘ ***NC4'***; ***restrições***(***NC4'***)=***restrições***(***NC4***);

(ii) Substitua a ED ***AUX_ENSINO***[***NC4***]⊂***CURSOS***[***NC9***] pela ED
ψ = ***CURSOS_4***[***NC4'***]⊂***CURSOS***[***NC9***] com semântica de atualização:
SI(CURSOS_4 , ψ) = ***rejeita***
SD(CURSOS, ψ) = ***propaga***

f) Transformações de restrições resultantes da transformação T_6

(i) Adicione as seguintes restrições:

- ***chefe:DEPTOS_5→PROF_5*** com semântica de atualização:
  SI (DEPTOS_5, (chefe:DEPTOS_5→PROF_5)) = propaga;
  SD (PROF_5, (chefe:DEPTOS_5→PROF_5)) = propaga;

- ***chefe:DEPTOS_5•→PROF_5*** com semântica de atualização:
  SI (PROF_5, (chefe:DEPTOS_5•→PROF_5)) = propaga;
  SD (DEPTOS_5, (chefe:DEPTOS_5•→PROF_5) = propaga;
- ***NP5 ≡ chefe∘ NP5'; restrições(NP5')=restrições(NP5);***

(ii) Substitua a ED ***DEPTOS_5[NP5]⊂PROFESSORES[NP6]*** pela ED
  ψ = ***PROF_5[NP5']⊂PROFESSORES[NP6]*** com semântica de atualização:
  SI(PROF_5 , ψ) = rejeita
  SD(PROFESSORES, ψ) = rejeita

(g) Transformações de restrições resultantes da transformação T_7

(i) Adicione as seguintes restrições:
- ***depto6:PROFESSORES→DEPTO_6*** com semântica de atualização:
  SI (PROFESSORES, (depto6:PROFESSORES→DEPTO_6)) = propaga;
  SD (DEPTO_6, (depto6:PROFESSORES→DEPTO_6)) = propaga;
- ***depto6:PROFESSORES•→DEPTO_6*** com semântica de atualização:
  SI (DEPTO_6, (depto6:PROFESSORES•→DEPTO_6)) = propaga;
  SD (PROFESSORES, (depto6:PROFESSORES•→DEPTO_6) = propaga;
- ***ND6 ≡ depto6∘ ND6'; restrições(ND6')=restrições(ND6);***

(ii) Substitua a ED ***PROFESSORES[ND6]⊂DEPTOS_5[ND5]*** pela ED
  ψ = ***DEPTO_6[ND6']⊂DEPTOS_5[ND5]*** com semântica de atualização:
  SI(DEPTO_6 , ψ) = rejeita
  SD(DEPTOS_5, ψ) = propaga

(h) Transformações de restrições resultantes da transformação T_8

(i) Adicione as seguintes restrições:
- ***estudante:MATRÍCULAS→ESTUD_7*** com semântica de atualização:
  SI (MATRÍCULAS, (estudante:MATRÍCULAS→ESTUD_7)) = propaga;
  SD (ESTUD_7, (estudante:MATRÍCULAS→ESTUD_7)) = propaga;
- ***estudante:MATRÍCULAS•→ESTUD_7*** com semântica de atualização:
  SI (ESTUD_7, (estudante:MATRÍCULAS•→ESTUD_7)) = propaga;
  SD (MATRÍCULAS, (estudante:MATRÍCULAS•→ESTUD_7) = propaga;
- ***NE7 ≡ estudante∘ NE7'; restrições(NE7')=restrições(NE7);***

(ii) Substitua a ED ***MATRÍCULAS[NE7]⊂ESTUDANTES[NE8]*** pela ED
  ψ = ***ESTUD_7[NE7']⊂ESTUDANTES[NE8]*** com semântica de atualização:
  SI(ESTUD_7 , ψ) = rejeita
  SD(ESTUDANTES, ψ) = propaga

i) Transformações de restrições resultantes da transformação T_9

(i) Adicione as seguintes restrições:
- ***oferta7:MATRÍCULAS→OFERTAS_7*** com semântica de atualização:
  SI (MATRÍCULAS, (oferta7:MATRÍCULAS→OFERTAS_7)) = propaga;
  SD (OFERTAS_7, (oferta7:MATRÍCULAS→OFERTAS_7)) = propaga;
- ***oferta7:MATRÍCULAS•→OFERTAS_7*** com semântica de atualização:
  SI (OFERTAS_7, (oferta7:MATRÍCULAS•→OFERTAS_7)) = propaga;
  SD (MATRÍCULAS, (oferta7:MATRÍCULAS•→OFERTAS_7) = propaga;
- ***NC7 ≡ oferta7∘ NC7'; restrições(NC7')=restrições(NC7);***
- ***NP7 ≡ oferta7∘ NP7'; restrições(NP7')=restrições(NP7);***

(ii) Substitua a ED *MATRÍCULAS*[*NC7, NP7*]⊂*OFERTAS*[*NC1, NP1*] pela ED
ψ = ***OFERTAS_7***[*NC7', NP7'*]⊂***OFERTAS***[*NC1, NP1*] com semântica de atualização:
SI*(*OFERTAS_7* , ψ) = *rejeita
SD*(*OFERTAS*, ψ) = *propaga

(j) Transformações de restrições resultantes da transformação T_{10}

(i) Adicione as seguintes restrições:
- ***depto9:CURSOS→DEPTOS_9*** com semântica de atualização:
  ***SI* (*CURSOS*, (*depto9:CURSOS→DEPTOS_9*)) = propaga;**
  ***SD* (*DEPTOS_9*, (*depto9:CURSOS→DEPTOS_9*)) = propaga;**
- ***depto9:CURSOS•→DEPTOS_9*** com semântica de atualização:
  ***SI* (*DEPTOS_9*, (*depto9:CURSOS•→DEPTOS_9*)) = propaga;**
  ***SD* (*CURSOS*, (*depto9:CURSOS•→DEPTOS_9*) = propaga;**
- ***ND9* ≡ *depto9*∘ *ND9'*; *restrições*(*ND9'*)=*restrições*(*ND9*);**

(ii) Substitua a ED ***CURSOS***[*ND9*]⊂***DEPTOS_5***[*ND5*] pela ED
ψ = ***DEPTOS_9***[*ND9'*]⊂***DEPTOS_5***[*ND5*] com semântica de atualização:
SI*(*DEPTOS_9* , ψ) = *rejeita
SD*(*DEPTOS_5*, ψ) = *propaga

k) Transformações de restrições resultantes da transformação T_{11}

(i) Adicione as seguintes restrições:
- ***orientador10:AUX_ENSINO→PROF_10*** com semântica de atualização:
  ***SI* (*AUX_ENSINO*, (*orientador10:AUX_ENSINO→PROF_10*)) = propaga;**
  ***SD* (*PROF_10*, (*orientador10:AUX_ENSINO→PROF_10*)) = propaga;**
- ***orientador10:AUX_ENSINO•→PROF_10*** com semântica de atualização:
  ***SI* (*PROF_10*, (*orientador10:AUX_ENSINO•→PROF_10*)) = propaga;**
  ***SD* (*AUX_ENSINO*, (*orientador10:AUX_ENSINO•→PROF_10*) = propaga;**
- ***NP10* ≡ *orientador10*∘ *NP10'*; *restrições*(*NP10'*)=*restrições*(*NP10*);**

(ii) Substitua a ED ***AUX_ENSINO***[*NP10*]⊂***PROFESSORES***[*NP6*] pela ED
ψ = ***PROF_10***[*NP10'*]⊂***PROFESSORES***[*NP6*] com semântica de atualização:
SI*(*PROF_10* , ψ) = *rejeita
SD*(*COURSES*, ψ) = *rejeita

PASSO2.2: Transformação das restrições de dependência existencial em restrições estruturais e restrições de equivalência_de_relacionamentos.

O esquema ***EN*** está em forma normal decomposta, e todas as restrições de dependência existenciais em ***EN*** podem ser reescritas como restrições estruturais e restrições de equivalência_de_relacionamentos. Do passo 2.1, nós temos que as restrições de dependências existenciais em ***EN*** são:

ED1:***CURSOS_1***[*NC1', ND1'*]⊂
CURSOS[*NC9, ND9*]
com semântica de atualização:
SI(***CURSOS_1***, *ED1*) = ***rejeita***
SD(***CURSOS***, *ED1*) = ***propaga***

ED2:***PROF_1***[*NP1',ND1''*]⊂
PROFESSORES[*NP6, ND6*]
com semântica de atualização:
SI(***PROF_1***, *ED2*) = ***rejeita***
SD(***PROFESSORES***, *ED2*) = ***propaga***

ED3:***PROF_2***[*NP2'*]⊂***PROFESSORES***[*NP6*]
com semântica de atualização:
SI(***PROF_2***, *ED3*) = ***rejeita***
SD(***PROFESSORES***, *ED3*) = ***rejeita***

ED4:***PROF_3***[*NP2'*]⊂***PROFESSORES***[*NP6*]
com semântica de atualização:
SI(***PROF_3***, *ED4*) = ***rejeita***
SD(***PROFESSORES***, *ED4*) = ***rejeita***

ED5:***CURSOS_4***[*NC4'*]⊂***CURSOS***[*NC9*]
com semântica de atualização:
SI(***CURSOS_4***, *ED5*) = ***rejeita***
SD(***CURSOS***, *ED5*) = ***rejeita***

ED6:***PROF_5***[*NP5'*]⊂***PROFESSORES***[*NP6*]
com semântica de atualização:
SI(***PROF_5***, *ED6*) = ***rejeita***
SD(***PROFESSORES***, *ED6*) = ***rejeita***

ED7:***DEPTOS_6***[*ND6'*]⊂***DEPTOS_5***[*ND5*]
com semântica de atualização:
SI(***DEPTO_6***[, *E7*) = ***rejeita***
SD(***DEPARTAMENTOS***, *ED7*) = ***propaga***

ED8:***OFERTAS_7***[*NC7',NP7'*]⊂
OFERTAS[*NC1,NP1*]
with update semantics:
SI(***OFERTAS_7***, *ED8*) = ***rejeita***
SD(***OFERTAS***, *ED8*) = ***propaga***

ED9:***ESTUD_7***[*NE7'*]⊂***ESTUDANTES***[*NE8*]
with update semantics:
SI(***ESTUD_7***, *ED9*) = ***rejeita***
SD(***ESTUDANTES***, *ED9*) = ***propaga***

ED10:***DEPTO_9***[*ND9'*]⊂***DEPTOS_5***[*ND5*]
with update semantics:
SI(***CURSOS***, *ED10*) = ***rejeita***
SD(***DEPARTAMENTOS***, *ED10*) = ***propaga***

ED11:***PROF_10***[*NP10'*]⊂***PROFESSORES***[*NP6*]
com semântica de atualização:
SI(***PROF_10***, *ED11*) = ***rejeita***
SD(***PROFESSORES***, *ED11*) = ***rejeita***

As restrições de dependência existenciais acima são transformadas em restrições estruturais e restrições de equivalência_de_relacionamentos através da aplicação das seguintes transformações:

T12 = **TED#1**(***CURSOS_1***[*NC1', ND1'*]⊂***CURSOS***[*NC9, ND9*])
T13 = **TED#1**(***PROF_1***[*NP1', ND1''*]⊂***PROFESSORES***[*NP6, ND6*])
T14 = **TED#1**(***PROF_2***[*NP2'*]⊂***PROFESSORES***[*NP6*])
T15 = **TED#1**(***PROF_3***[*NP2'*]⊂***PROFESSORES***[*NP6*])
T16 = **TED#1**(***CURSOS_4***[*NC4'*]⊂***CURSOS***[*NC9*])
T17 = **TED#1**(***PROF_5***[*NP5'*]⊂***PROFESSORES***[*NP6*])
T18 = **TED#1**(***DEPTO_6***[*ND6'*]⊂***DEPTOS_5***[*ND5*])
T18 = **TED#1**(***OFERTAS_7***[*NC7',NP7'*]⊂***OFERTAS***[*NC1,NP1*])
T20 = **TED#1**(***ESTUDANTES_7***[*NE7'*]⊂***ESTUDANTES***[*NE8*])
T21 = **TED#1**(***DEPTO_9***[*ND9'*]⊂***DEPTOS_9***[*ND5*])
T22 = **TED#1**(***PROF_10***[*NP10'*]⊂***PROFESSORES***[*NP6*])

Aplicando as transformações $T_{12}, \ldots, T_{22}$ no esquema *EN* temos:

Seja $EN^* = T_{22} \circ \ldots \circ T_{12}$ (*EN*). A estrutura do esquema *EN** é a mesma do esquema *EN*. As novas restrições de *EN** são obtidas pelas seguintes transformações de restrições:

a) Transformações de restrições resultantes da transformação T_{12}

- Substitua a ED ***CURSOS_1***[***NC1'***, ***ND1'***]⊂***CURSOS***[***NC9***, ***ND9***] pelas seguintes restrições
  - ***CURSOS_1⊂CURSOS*** com semântica de atualização:
    SI(CURSOS_1, CURSOS_1⊂CURSOS) = rejeita
    SD(CURSOS, CURSOS_1⊂CURSOS) = propaga
  - ***NC1' ≡ NC9***, ***ND1' ≡ ND9.***

b) Transformações de restrições resultantes da transformação T_{13}

- Substitua a ED ***PROF_1***[***NP1'***, ***ND1'***]⊂***PROFESSORES***[***NP6***, ***ND6***] pelas seguintes restrições
  - ***PROF_1⊂PROFESSORES*** com semântica de atualização:
    SI(PROF_1, PROF_1⊂PROFESSORES) = rejeita
    SD(PROFESSORES, PROF_1⊂PROFESSORES) = propaga
  - ***NP1' ≡ NP6***, ***ND1'' ≡ ND6.***

c) Transformações de restrições resultantes da transformação T_{14}

- Substitua a ED ***PROF_2***[***NP2'***]⊂***PROFESSORES***[***NP6***] pelas seguintes restrições
  - ***PROF_2⊂PROFESSORES*** com semântica de atualização:
    SI(PROF_2, PROF_2⊂PROFESSORES) = rejeita
    SD(PROFESSORES, PROF_2⊂PROFESSORES) = rejeita
  - ***NP2' ≡ NP6.***

d) Transformações de restrições resultantes da transformação T_{15}

- Substitua a ED ***PROF_3***[***NP3'***]⊂***PROFESSORES***[***NP6***] pelas seguintes restrições
  - ***PROF_3⊂PROFESSORES*** com semântica de atualização:
    SI(PROF_3, PROF_3⊂PROFESSORES) = rejeita
    SD(PROFESSORES, PROF_3⊂PROFESSORES) = rejeita
  - ***NP3' ≡ NP6.***

e) Transformações de restrições resultantes da transformação T_{16}

- Substitua a ED ***CURSOS_4***[***NC4'***]⊂***CURSOS***[***NC9***] pelas seguintes restrições
  - ***CURSOS_4⊂CURSOS*** com semântica de atualização:
    SI(CURSOS_4, CURSOS_4⊂CURSOS) = rejeita
    SD(CURSOS, CURSOS_4⊂CURSOS) = rejeita
  - ***NC4' ≡ NC9.***

f) Transformações de restrições resultantes da transformação T_{17}

- Substitua a ED ***PROF_5***[***NP5'***]⊂***PROFESSORES***[***NP6***] pelas seguintes restrições
  - ***PROF_5⊂PROFESSORES*** com semântica de atualização:
    SI(PROF_5, PROF_5⊂PROFESSORES) = rejeita
    SD(PROFESSORES, PROF_5⊂PROFESSORES) = rejeita
  - ***NP5' ≡ NP6.***

g) Transformações de restrições resultantes da transformação T_{18}

- Substitua a ED *DEPTOS_6*[*ND6'*]⊂*DEPTOS_5*[*ND5*] pelas seguintes restrições
  - ***DEPTOS_6⊂DEPTOS_5*** com semântica de atualização:
    SI(DEPTOS_6, DEPTOS_6⊂DEPTOS_5) = rejeita
    SD(DEPTOS_5, DEPTOS_6⊂DEPTOS_5) = propaga
  - *ND6'* ≡ *ND5*.

g) Transformações de restrições resultantes da transformação T_{19}

- Substitua a ED *OFERTAS_7*[*NC7', NP7'*]⊂*OFERTAS*[*NC1, NP1*] pelas seguintes restrições
  - ***OFERTAS_7⊂OFERTAS*** com semântica de atualização:
    SI(OFERTAS_7, OFERTAS_7⊂OFERTAS) = rejeita
    SD(OFERTAS, OFERTAS_7⊂OFERTAS) = propaga
  - *NC7'* ≡ *NC1*, *NP7'* ≡ *NP1*.

h) Transformações de restrições resultantes da transformação T_{20}

- Substitua a ED *ESTUD_7*[*NE7'*]⊂*ESTUDANTES*[*NE8*] pelas seguintes restrições
  - ***ESTUD_7⊂ESTUDANTES*** com semântica de atualização:
    SI(ESTUD_7, ESTUD_7⊂ESTUDANTES) = rejeita
    SD(ESTUDANTES, ESTUD_7⊂ESTUDANTES) = propaga
  - *NE7'* ≡ *NE8*.

i) Transformações de restrições resultantes da transformação T_{21}

- Substitua a ED *DEPTOS_9*[*ND9'*]⊂*DEPTOS_5*[*ND5*] pelas seguintes restrições
  - ***DEPTOS_9⊂DEPTOS_5*** com semântica de atualização:
    SI(DEPTOS_9, DEPTOS_9⊂DEPTOS_5) = rejeita
    SD(DEPTOS_5, DEPTOS_9⊂DEPTOS_5) = propaga
  - *ND9'* ≡ *ND5*.

j) Transformações de restrições resultantes da transformação T_{22}

- Substitua a ED *PROF_10*[*NP10'*]⊂*PROFESSORES*[*NP6*] pelas seguintes restrições
  - ***PROF_10⊂PROFESSORES*** com semântica de atualização:
    SI(PROF_10, PROF_10⊂PROFESSORES) = rejeita
    SD(PROFESSORES, PROF_10⊂PROFESSORES) = rejeita
  - *NP10'* ≡ *NP6*.

PASSO2.3: Transformação das restrições de dependência funcional de união em restrições de equivalência_de_relacionamentos.

As restrições de dependência funcionais de união em ***EN**** são transformadas em restrições de equivalência_de_relacionamentos através da aplicação das seguintes transformações:

T_{23} = **TDFU**(<***CURSOS***([***NC9***]→***ND9***), ***CURSOS_1***([***NC1'***]→***ND1'***) >)

T_{24} = **TDFU**(<***PROFESSORES***([***NP6***]→***ND6***), ***PROF_1***([***NP1'***]→***ND1''***) >)

Aplicando as transformações T_{23}, e T_{24} no esquema ***EN**** temos:

Seja ***ER*** = $T_{24} \circ T_{23}$ (***EN****). A estrutura do esquema ***ER*** (esquema reestruturado) é a mesma do esquema ***EN*** (figura 6.4). As novas restrições de ***EN**** são obtidas pelas seguintes transformações de restrições:

a) Transformações de restrições resultantes da transformação T_{23}
- Substitua a DFU <***CURSOS***([***NC9***]→***ND9***), ***CURSOS_1***([***NC1'***]→***ND1'***)> por ***ND9≡ND1'***.

a) Transformações de restrições resultantes da transformação T_{24}
- Substitua a DFU <***PROFESSORES***([***NP6***]→***ND6***), ***PROF_1***([***NP1'***]→***ND1''***)> por ***ND6≡ND1''***.

Está concluido o passo de reestruturação. A estrutura do esquema ***ER*** é esta da figura 6.4, e as restrições de integridade em ***ER*** estão sumarizadas abaixo.

Restrições de equivuivalência_de_relacionamentos:

*ER1: **NE2 ≡NE3≡NE4≡NE8***
*ER2: **ND8≡ND5***
*ER3: **EN8≡EN5***
*ER4: **NP2≡NP3≡NP10***
*ER5:**NC1 ≡ curso1∘ NC1'***
*ER6:**ND1 ≡ curso1∘ ND1'***
*ER5: **NP1 ≡professor1∘ NP1'**;*
*ER6:**ND1 ≡professor1∘ ND1''***
*ER7: **NP2 ≡ orientador2∘ NP2'**;*
*ER8:**NP3 ≡ orientador3∘ NP3'***
*ER9:**NC4 ≡ curso4∘ NC4'***
*ER10: **NP5 ≡ chefe∘ NP5'***
*ER11:**ND6 ≡ depto6∘ ND6'***
*ER12:**NE7 ≡ estudante∘ NE7'***
*ER13:**NC7 ≡ oferta7∘ NC7'***
*ER14:**NP7 ≡ oferta7∘ NP7'***
*ER15:**ND9 ≡ depto9∘ ND9'***
*ER16:**NP10 ≡ orientador10∘ NP10'***
*ER17:**NC1' ≡ NC9***
*ER18:**ND1' ≡ ND9**.*
*ER19:**NP1' ≡ NP6***
*ER20:**ND1' ≡ ND6***
*ER21:**NP2' ≡ NP6**.*
*ER22:**NP3' ≡ NP6***
*ER23: **NC4' ≡ NC9***
*ER24: **NP5' ≡ NP6***
*ER25: **ND6' ≡ ND5***
*ER26: **NC7' ≡ NC1***
*ER27:**NP7' ≡ NP1***
*ER28: **NE7' ≡ NE8***
*ER29: **ND9' ≡ ND5***
*ER30:**NP10' ≡ NP6***

Restrições referenciais:

RR1:***curso1*:*OFERTAS→CURSOS_1***
com semântica de atualização:
SI(OFERTAS, RR1) = propaga
SD(CURSOS_1, RR1) = propaga
SM(depto1, RR1) = propaga

RR2:***professor1*:*OFERTAS→PROF_1***
com semântica de atualização:
SI(OFERTAS, RE3) = propaga
SD(PROF_1, RE3) = propaga
SM(professor1, RE3) = propaga

RR3:***orientador2*:*ESTUDANTES_PÓS→PROF_2***
com semântica de atualização:
SI(ESTUDANTES_PÓS, RR3) = propaga
SD(PROF_2, RR3) = propaga
SM(orientador2, RR3) = propaga

RR4:***orientador3*:*ESTUDANTES_TI→PROF_3***
com semântica de atualização:
SI(ESTUDANTES_TI, RR4) = propaga
SD(PROF_3, RR4) = propaga
SM(orientador3, RR4) = propaga

RR5:***curso4*:*AUX_ENSINO→CURSOS_4***
com semântica de atualização:
SI(AUX_ENSINO, RR5) = propaga
SD(CURSOS_4, RR5) = propaga
SM(curso4, RR5) = propaga

RR6:***chefe*:*DEPTOS_5→PROF_5***
com semântica de atualização:
SI(DEPTOS_5, RR6) = propaga
SD(PROF_5, RR6) = propaga
SM(chefe, RR6) = propaga

RR7:***depto6*:*PROFESSORES→DEPTOS_6***
com semântica de atualização:
SI(PROFESSORES, RR7) = propaga
SD(DEPTOS_6, RR7) = propaga
SM(depto6, RR7) = propaga

RR8:***estudante*:*MATRÍCULAS→ESTUD_7***
com semântica de atualização:
SI(MATRÍCULAS, RR8) = propaga
SD(ESTUD_7, RR8) = propaga
SM(estudante, RR8) = propaga

RR9:***oferta7*:*MATRÍCULAS→OFERTAS_7***
com semântica de atualização:
SI(MATRÍCULAS, RR9) = propaga
SD(OFERTAS_7, RR9) = propaga
SM(oferta7, RR9) = propaga

RR10:***depto1*:*ESTUDANTES→ DEPARTAMENTOS***
com semântica de atualização:
SI(ESTUDANTES, RE3) = rejeita
SD(ESTUDANTES_PÓS, RE3) = propaga
SM(depto1, RE3) = rejeita

RR11:***depto9*:*CURSOS→DEPTOS_9***
com semântica de atualização:
SI(CURSOS, RR11) = propaga
SD(DEPTOS_9, RR11) = propaga
SM(depto9, RR11) = propaga

RR12:***orientador10*:*AUX_PESQUISA→ PROF_10*** com semântica de atualização:
SI(AUX_PESQUISA, RR12) = propaga
SD(PROF_10, RR12) = propaga
SM(orientador10, RR12) = propaga

Restrições referenciais totais

<u>RT1</u>:*curso1*:***OFERTAS•→CURSOS_1***
com semântica de atualização:
SD(OFERTAS, *RT1*) = ***propaga***
SD(CURSOS_1, *RT1*) = ***propaga***
SM(depto1, *RT1*) = ***propaga***

<u>RT2</u>:***professor1:OFERTAS•→PROF_1***
com semântica de atualização:
SD(OFERTAS, *RE3*) = ***propaga***
SD(PROF_1, *RE3*) = ***propaga***
SM(professor1, *RE3*) = ***propaga***

<u>RT3</u>:***orientador2:ESTUDANTES_PÓS•→ PROF_2*** com semântica de atualização:
SD(ESTUDANTES_PÓS, *RT3*) = ***propaga***
SD(PROF_2, *RT3*) = ***propaga***
SM(orientador2, *RT3*) = ***propaga***

<u>RT4</u>:***orientador3:ESTUDANTES_TI•→ PROF_3*** com semântica de atualização:
SD(ESTUDANTES_TI, *RT4*) = ***propaga***
SD(PROF_3, *RT4*) = ***propaga***
SM(orientador3, *RT4*) = ***propaga***

<u>RT5</u>:***curso4:AUX_ENSINO•→CURSOS_4***
com semântica de atualização:
SD(AUX_ENSINO, *RT5*) = ***propaga***
SD(CURSOS_4, *RT5*) = ***propaga***
SM(curso4, *RT5*) = ***propaga***

<u>RT6</u>:***chefe:DEPTOS_5•→PROF_5***
com semântica de atualização:
SD(DEPTOS_5, *RT6*) = ***propaga***
SD(PROF_5, *RT6*) = ***propaga***
SM(chefe, *RT6*) = ***propaga***

<u>RT7</u>:***depto6:PROFESSORES•→DEPTOS_6***
com semântica de atualização:
SD(PROFESSORES, *RT7*) = ***propaga***
SD(DEPTOS_6, *RT7*) = ***propaga***
SM(depto6, *RT7*) = ***propaga***

<u>RT8</u>:***estudante:MATRÍCULAS•→ESTUD_7***
com semântica de atualização:
SD(MATRÍCULAS, *RT8*) = ***propaga***
SD(ESTUD_7, *RT8*) = ***propaga***
SM(estudante, *RT8*) = ***propaga***

<u>RT9</u>:***oferta7:MATRÍCULAS•→OFERTAS_7***
com semântica de atualização:
SD(MATRÍCULAS, *RT9*) = ***propaga***
SD(OFERTAS_7, *RT9*) = ***propaga***
SM(oferta7, *RT9*) = ***propaga***

<u>RT11</u>:***depto9:CURSOS•→DEPTOS_9***
com semântica de atualização:
SD(CURSOS, *RT11*) = ***propaga***
SD(DEPTOS_9, *RT11*) = ***propaga***
SM(depto9, *RT11*) = ***propaga***

<u>RT12</u>:***orientador10:AUX_PESQUISA•→ PROF_10*** com semântica de atualização:
SD(AUX_PESQUISA, *RT12*) = ***propaga***
SD(PROF_10, *RT12*) = ***propaga***
SM(orientador10, *RT12*) = ***propaga***

Restrições Estruturais

RE1: ***ESTUDANTES_PÓS*** $\subset$ ***ESTUDANTES***,
com semântica de atualização:
SI(***ESTUDANTES_PÓS***, *RE1*) = ***propaga***
SD(***ESTUDANTES***, *RE1*) = ***propaga***

RE2: ***ESTUDANTES_TI*** $\subset$ ***ESTUDANTES***,
com semântica de atualização:
SI(***ESTUDANTES_TI***, *RE2*) = ***propaga***
SD(***ESTUDANTES***, *RE2*) = ***propaga***

RE3: ***AUX_ENSINO*** $\subset$ ***ESTUDANTES_PÓS***,
com semântica de atualização:
SI(***AUX_ENSINO***, *RE3*) = ***rejeita***
SD(***ESTUDANTES_PÓS***, *RE3*) = ***propaga***

RE4: ***AUX_PESQUISA*** $\subset$ ***ESTUDANTES_PÓS***,
com semântica de atualização:
SI(***AUX_PESQUISA***, *RE3*) = ***rejeita***
SD(***ESTUDANTES_PÓS***, *RE3*) = ***propaga***

RE5: ***DEPTOS8*** = ***DEPARTAMENTOS***,

RE6:***CURSOS_1***$\subset$***CURSOS***
com semântica de atualização:
SI(***CURSOS_1***, *RE6*) = ***rejeita***
SD(***CURSOS***, *RE6*) = ***propaga***

RE7:***PROF_1***$\subset$***PROFESSORES***
com semântica de atualização:
SI(***PROF_1***, *RE7*) = ***rejeita***
SD(***PROFESSORES***, *RE7*) = ***propaga***

RE8:***PROF_2***[*NP2'*]$\subset$***PROFESSORES***
com semântica de atualização:
SI(***PROF_2***, *RE8*) = ***rejeita***
SD(***PROFESSORES***, *RE8*) = ***rejeita***

RE9:***PROF_3***$\subset$***PROFESSORES***
com semântica de atualização:
SI(***PROF_3***, *RE9*) = ***rejeita***
SD(***PROFESSORES***, *RE9*) = ***rejeita***

RE10:***CURSOS_4***$\subset$***CURSOS***
com semântica de atualização:
SI(***CURSOS_4***, *RE10*) = ***rejeita***
SD(***CURSOS***, *RE11*) = ***rejeita***

RE11:***PROF_5***$\subset$***PROFESSORES***
com semântica de atualização:
SI(***PROF_5***, *RE11*) = ***rejeita***
SD(***PROFESSORES***, *RE11*) = ***rejeita***

RE12:***DEPTOS_6***$\subset$***DEPTOS_5***
com semântica de atualização:
SI(***DEPTO_6***, *RE12*) = ***rejeita***
SD(***DEPTOS_5***, *RE12*) = ***propaga***

RE13:***OFERTAS_7***$\subset$***OFERTAS***
com semântica de atualização:
SI(***OFERTAS_7***, *RE13*) = ***rejeita***
SD(***OFERTAS***, *RE13*) = ***propaga***

RE14:***ESTUD_7***$\subset$***ESTUDANTES***
com semântica de atualização:
SI(***ESTUD_7***, *RE14*) = ***rejeita***
SD(***ESTUDANTES***, *RE14*) = ***propaga***

RE15:***DEPTO_9***$\subset$***DEPTOS_5***
com semântica de atualização:
SI(***CURSOS***, *RE15*) = ***rejeita***
SD(***DEPARTAMENTOS***, *ED10*) = ***propaga***

RE16:***PROF_10***$\subset$***PROFESSORES***
com semântica de atualização:
SI(***PROF_10***, *RE16*) = ***rejeita***
SD(***PROFESSORES***, *ED11*) = ***rejeita***

6.3 PASSO#3: OTIMIZAÇÃO

A seguir discutiremos cada passo da otimização do esquema ***ER***.

PASSO3.1: INTEGRAÇÃO DAS CLASSES EQUIVALENTES

As classes ***DEPTOS_5*** e ***DEPTOS_8*** podem ser integradas em uma classe através da seguinte transformação:

T_{25} = ***MERGE_EQUIVALENT***({***DEPTOS_5***, ***DEPTOS_8***}, ***DEPARTAMENTOS***)

Seja $EO_1 = T_1(ER)$. No esquema EO_1, as classes ***DEPTOS_5*** e ***DEPTOS_8*** são removidas e a classe ***DEPARTAMENTOS*** é adicionada. A Figura 6.5 mostra a estrutura da classe ***DEPARTAMENTOS***.

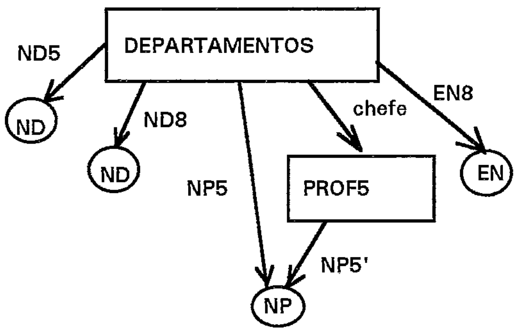


Figura 6.5: A nova classe ***DEPARTAMENTOS***

As novas restrições de EO_1 são obtidas pelas seguintes transformações de restrições:

- Substitua ***ND5:DEPTOS_5 →ND*** por ***ND5:DEPARTAMENTOS →ND***
- Substitua ***ND8:DEPTOS_8 →ND*** por ***ND8:DEPARTAMENTOS →ND***
- Substitua ***EN8:DEPTOS_8 →EN*** por ***EN8:DEPARTAMENTOS →EN***
- Substitua ***NP5:DEPTOS_5 →NP*** por ***NP5:DEPARTAMENTOS →NP***
- Substitua ***chefe:DEPTOS_5 →PROF_5*** por ***chefe:DEPARTAMENTOS →PROF_5***
- Substitua ***DEPTOS_6*** ⊂ ***DEPTOS_5*** por ***DEPTOS_6*** ⊂ ***DEPARTAMENTOS***
- Substitua ***DEPTOS_9*** ⊂ ***DEPTOS_5*** por ***DEPTOS_9*** ⊂ ***DEPARTAMENTOS***
- Substitua ***depto8:ESTUDANTES→DEPTOS_8*** por

    depto8:ESTUDANTES→DEPARTAMENTOS

PASSO3.2: INTEGRAÇÃO DAS CLASSES EMBUTIDAS

As classes embutidas do esquema EO_1, podem ser integradas através das seguintes transformações:

T_{26} = *INTEGRA_EMBUTIDA(OFERTAS_7, OFERTAS)*
T_{27} = *INTEGRA_EMBUTIDA(PROF_1, PROFESSORES)*
T_{28} = *INTEGRA_EMBUTIDA(PROF_2, PROFESSORES)*
T_{29} = *INTEGRA_EMBUTIDA(PROF_3, PROFESSORES)*
T_{30} = *INTEGRA_EMBUTIDA(PROF_5, PROFESSORES)*
T_{31} = *INTEGRA_EMBUTIDA(PROF_10, PROFESSORES)*
T_{32} = *INTEGRA_EMBUTIDA(CURSOS_4, CURSOS)*
T_{33} = *INTEGRA_EMBUTIDA(CURSOS_1, CURSOS)*
T_{34} = *INTEGRA_EMBUTIDA(DEPTOS_6, DEPARTAMENTOS)*
T_{35} = *INTEGRA_EMBUTIDA(DEPTOS_9, DEPARTAMENTOS)*
T_{36} = *INTEGRA_EMBUTIDA(ESTUD_7, ESTUDANTES)*

- Aplicando as transformações acima no esquema EO_1 temos:

Seja $EO_2 = T_{36} \circ \ldots \circ T_{26}(EO_1)$. A estrutura do esquema EO_2 é mostrada na figura 6.6. As novas restrições de EO_2 são obtidas pelas seguintes transformações de restrições:

a) Transformações de restrições resultantes da transformação T_{26}

(i) Remova a restrição ***oferta*:*MATRÍCULAS* •→*OFERTAS_7***
(ii) Substitua a restrição ***oferta*:*MATRÍCULAS* →*OFERTAS_7*** pela restrição
oferta*:*MATRÍCULAS* →*OFERTAS com semântica de atualização:
***SI(MATRÍCULAS, (oferta:MATRÍCULAS →OFERTAS)*=rejeita**
***SD(OFERTAS, (oferta:MATRÍCULAS →OFERTAS)*=propaga**
***SM(oferta, (oferta:MATRÍCULAS →OFERTAS)*=rejeita**
(iii) substitua a restrição ***NC7≡ oferta* ∘ *NC7'*** por ***NC7 ≡ oferta* ∘ *NC1***;
(iv) substitua a restrição ***NP7≡ oferta* ∘ *NP7'*** por ***NP7 ≡ oferta* ∘ *NP1***;
(v) Remova ***OFERTAS_7*** e todas as restrições em ***OFERTAS_7.***

b) Transformações de restrições resultantes da transformação T_{27}

(i) Remova a restrição ***professor1*:*OFERTAS* •→*PROF_1***
(ii) Substitua a restrição ***professor1*:*OFERTAS* →*PROF_1*** pela restrição
professor1*:*OFERTAS* →*PROFESSORES com semântica de atualização:
***SI(OFERTAS, (professor1:OFERTAS →PROFESSORES)*=rejeita**
***SD(PROFESSORES, (professor1:OFERTAS →PROFESSORES)*=propaga**
***SM(professor1, (professor1:OFERTAS →PROFESSORES)*=rejeita**
(iii) substitua a restrição ***ND1≡ professor1* ∘ *ND1"*** por ***ND1≡ professor1* ∘ *ND6***;
(iv) substitua a restrição ***NP1≡ professor1* ∘ *NP1'*** por ***NP1≡ professor1* ∘ *NP6***;
(v) Remova ***PROF_1*** e todas as restrições em ***PROF_1.***

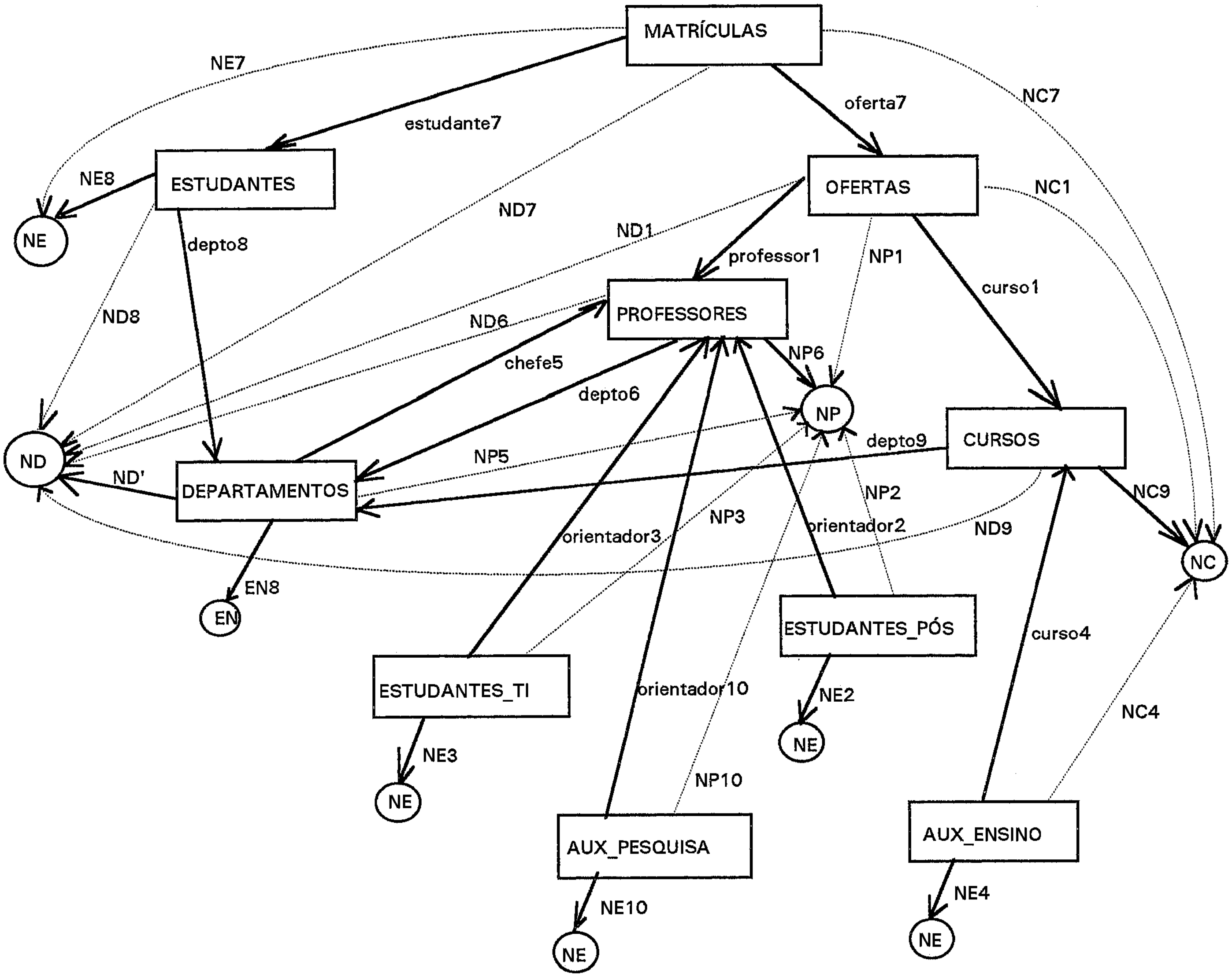


Figura 6.6: Estrutura do esquema EO_2

c) Transformações de restrições resultantes da transformação T_{28}

(i) Remova a restrição ***orientador2:ESTUDANTES_PÓS •→PROF_2***

(ii) Substitua a restrição ***orientador2:ESTUDANTES_PÓS →PROF_2*** pela restrição ***orientador2:ESTUDANTES_PÓS →PROFESSORES*** com semântica de atualização:

***SI(ESTUDANTES_PÓS, (orientador2:ESTUDANTES_PÓS →PROFESSORES)*=rejeita**

***SD(PROFESSORES, (orientador2:ESTUDANTES_PÓS →PROFESSORES)*=rejeita**

***SM(orientador2, (orientador2:ESTUDANTES_PÓS →PROFESSORES)*=rejeita**

(iv) substitua a restrição ***NP2≡ orientador2*** ∘ ***NP2'*** por ***NP2≡ orientador2*** ∘ ***NP6***;

(v) Remova ***PROF_2*** e todas as restrições em ***PROF_2.***

d) Transformações de restrições resultantes da transformação T_{29}

(i) Remova a restrição ***orientador3:ESTUDANTES_TI •→PROF_3***

(ii) Substitua a restrição ***orientador3:ESTUDANTES_TI →PROF_3*** pela restrição ***orientador3:ESTUDANTES_TI →PROFESSORES*** com semântica de atualização:

***SI(ESTUDANTES_TI, (orientador3:ESTUDANTES_TI →PROFESSORES)*=rejeita**

***SD(PROFESSORES, (orientador3:ESTUDANTES_TI →PROFESSORES)*=rejeita**

***SM(orientador3, (orientador3:ESTUDANTES_TI →PROFESSORES)*=rejeita**

(iv) substitua a restrição ***NP3≡ orientador3*** ∘ ***NP3'*** por ***NP3≡ orientador3*** ∘ ***NP6***;

(v) Remova ***PROF_3*** e todas as restrições em ***PROF_3.***

e) Transformações de restrições resultantes da transformação T_{30}

(i) Remova a restrição ***chefe:DEPARTAMENTOS •→PROF_5***

(ii) Substitua a restrição ***chefe:DEPARTAMENTOS →PROF_5*** pela restrição ***chefe:DEPARTAMENTOS →PROFESSORES*** com semântica de atualização:

***SI(DEPARTAMENTOS, (chefe:DEPARTAMENTOS →PROFESSORES)*=rejeita**

***SD(PROFESSORES, (chefe:DEPARTAMENTOS →PROFESSORES)*=rejeita**

***SM(chefe, (chefe:DEPARTAMENTOS →PROFESSORES)*=rejeita**

(iv) substitua a restrição ***NP5≡ chefe*** ∘ ***NP5'*** por ***NP5≡ chefe*** ∘ ***NP6***;

(v) Remova ***PROF_5*** e todas as restrições em ***PROF_5.***

f) Transformações de restrições resultantes da transformação T_{31}

(i) Remova a restrição ***orientador10:AUX_PESQUISA •→PROF_10***

(ii) Substitua a restrição ***orientador10:AUX_PESQUISA →PROF_10*** pela restrição ***orientador10:AUX_PESQUISA →PROFESSORES*** com semântica de atualização:

***SI(AUX_PESQUISA, (orientador10:AUX_PESQUISA →PROFESSORES)*=rejeita**

***SD(PROFESSORES, (orientador10:AUX_PESQUISA →PROFESSORES)*=rejeita**

***SM(orientador10, (orientador10:AUX_PESQUISA →PROFESSORES)*=rejeita**

(iv) substitua a restrição ***NP10≡ orientador10*** ∘ ***NP10'*** por ***NP10≡ orientador10*** ∘ ***NP6***;

(v) Remova ***PROF_10*** e todas as restrições em ***PROF_10.***

g) Transformações de restrições resultantes da transformação T_{32}

(i) Remova a restrição ***curso4:AUX_ENSINO •→CURSOS_4***

(ii) Substitua a restrição ***curso4:AUX_ENSINO →CURSOS_4*** pela restrição ***curso4:AUX_ENSINO →CURSOS*** com semântica de atualização:

***SI(AUX_ENSINO, (curso4:AUX_ENSINO →CURSOS)*=rejeita**

***SD(CURSOS, (curso4:AUX_ENSINO →CURSOS)*=rejeita**

***SM(curso4, (curso4:AUX_ENSINO →CURSOS)*=rejeita**

(iv) substitua a restrição ***NC4≡ curso4*** ∘ ***NC4'*** por ***NC4≡ curso4*** ∘ ***NC9***;

(v) Remova *CURSOS_4* e todas as restrições em *CURSOS_4.*

h) Transformações de restrições resultantes da transformação T_{33}

(i) Remova a restrição ***curso1**:OFERTAS* •→*CURSOS_1*

(ii) Substitua a restrição ***curso1**:OFERTAS* →*CURSOS_1* pela restrição
***curso1**:OFERTAS* →*CURSOS* com semântica de atualização:
SI(OFERTAS, (***curso1**:OFERTAS* →*CURSOS*)=**rejeita**
SD(CURSOS, (***curso1**:OFERTAS* →*CURSOS*)=**propaga**
*SM(**curso1***, (***curso1**:OFERTAS* →*CURSOS*)=**rejeita**

(iv) substitua a restrição *NC1≡* ***curso1*** ∘ *NC1'* por *NC1≡* ***curso1*** ∘ *NC9*;

(v) Remova *CURSOS_1* e todas as restrições em *CURSOS_1.*

i) Transformações de restrições resultantes da transformação T_{34}

(i) Remova a restrição ***depto6**:PROFESSORES* •→*DEPTOS_6*

(ii) Substitua a restrição ***depto6**:PROFESSORES* →*DEPTOS_6* pela restrição
***depto6**:PROFESSORES* →*DEPARTAMENTOS* com semântica de atualização:
SI(PROFESSORES, (***depto6**:PROFESSORES* →*DEPARTAMENTOS*)=**rejeita**
SD(DEPARTAMENTOS, (***depto6**:PROFESSORES* →*DEPARTAMENTOS*)=**propaga**
*SM(**depto6***, (***depto6**:PROFESSORES* →*DEPARTAMENTOS*)=**rejeita**

(iv) substitua a restrição *ND6≡* ***depto6*** ∘ *ND6'* por *ND6≡* ***depto6*** ∘ *ND5*;

(v) Remova *DEPTOS_6* e todas as restrições em *DEPTOS_6.*

j) Transformações de restrições resultantes da transformação T_{35}

(i) Remova a restrição ***depto9**:CURSOS* •→*DEPTOS_9*

(ii) Substitua a restrição ***depto9**:CURSOS* →*DEPTOS_9* pela restrição
***depto9**:CURSOS* →*DEPARTAMENTOS* com semântica de atualização:
SI(CURSOS, (***depto9**:CURSOS* →*DEPARTAMENTOS*)=**rejeita**
SD(DEPARTAMENTOS, (***depto9**:CURSOS* →*DEPARTAMENTOS*)=**propaga**
*SM(**depto9***, (***depto9**:CURSOS* →*DEPARTAMENTOS*)=**rejeita**

(iv) substitua a restrição *ND9≡* ***depto9*** ∘ *ND9'* por *ND9≡* ***depto9*** ∘ *ND5*;

(v) Remova *DEPTOS_9* e todas as restrições em *DEPTOS_9.*

k) Transformações de restrições resultantes da transformação T_{36}

(i) Remova a restrição ***estudante**:MATRÍCULAS* •→*ESTUD_7*

(ii) Substitua a restrição ***estudante**:MATRÍCULAS* →*ESTUD_7* pela restrição
***estudante**:MATRÍCULAS* →*ESTUDANTES* com semântica de atualização:
SI(MATRÍCULAS, (***estudante**:MATRÍCULAS* →*ESTUDANTES*)=**rejeita**
SD(ESTUDANTES, (***estudante**:MATRÍCULAS* →*ESTUDANTES*)=**propaga**
*SM(**estudante***, (***estudante**:MATRÍCULAS* →*ESTUDANTES*)=**rejeita**

(iv) substitua a restrição *ND9≡* ***estudante*** ∘ *ND9'* por *ND9≡* ***estudante*** ∘ *ND5*;

(v) Remova *ESTUD_7* e todas as restrições em *ESTUD_7.*

PASSO3.3: INTEGRAÇÃO DE ATRIBUTOS SINÔNIMOS

Os atributos *ND5* e *ND8* da classe ***DEPARTAMENTOS*** podem ser integradas em um atributo através da seguinte transformação:

T_{37} = ***MERGE_ATRIBUTOS***({*ND5*, *ND8*}, *ND'*)

Seja $EO_3 = T_{37}(EO_2)$. EO_3 é obtido pelas seguintes transformações:

- Adiciona atributo *ND':DEPARTAMENTOS →ND*
- *restrições*(*ND'*)= *restrições*(*ND5*). (*ND5* e *ND8* têm as mesmas restrições)
- Remova atributos *ND5* e *ND8* e todas as restrições em *ND5* e *ND8*

PASSO3.4: REMOÇÃO DOS ATRIBUTOS HERDADOS:

Os atributos herdados ***NP10***, ***orientador10***, ***NE2***, ***NE3***, ***NE4*** e ***NE10*** podem ser removidos de EO_3 através das seguintes transformações:

T_{38} = ***REMOVE_ATRIBUTO#1***(***orientador10***, ***orientador2***)
T_{39} = ***REMOVE_ATRIBUTO#1***(***NP10***, ***NP2***)
T_{40} = ***REMOVE_ATRIBUTO#1***(***NE2***, ***NE1***)
T_{41} = ***REMOVE_ATRIBUTO#1***(***NE3***, ***NE1***)
T_{42} = ***REMOVE_ATRIBUTO#1***(***NE4***, ***NE1***)
T_{43} = ***REMOVE_ATRIBUTO#1***(***NE10***, ***NE1***)

Seja $EO_4 = T_{38} \circ \ldots \circ T_{43}\ (EO_3)$. EO_4 é obtido pelas seguintes transformações:

- Remova atributos ***NP10***, ***orientador10***, ***NE2***, ***NE3***, ***NE4*** e ***NE10***.

PASSO3.5: REMOÇÃO DOS ATRIBUTOS DERIVÁVEIS:

Na figura 6.6 os atributos deriváveis são representados com linhas pontilhadas. Os atributos deriváveis ***NC1***, ***ND1***, ***NP1***, ***NP2***, ***NP3***, ***NC4***, ***NP5***, ***ND6***, ***NE7***, ***NC7***, ***NP7***, ***ND9*** podem ser removidos de EO_4 através das seguinte transformações:

T_{44} = ***REMOVE_ATRIBUTO#2***(***NC1***, ***curso1*** ∘ ***NC1'***)

T_{45} = ***REMOVE_ATRIBUTO#2***(***ND1***, ***curso1*** ∘ ***ND1'***)

T_{46} = ***REMOVE_ATRIBUTO#2***(***NP1***, ***professor1*** ∘ ***ND1''***)

T_{47} = ***REMOVE_ATRIBUTO#2***(***NP2***, ***orientador2*** ∘ ***NP2'***)

T_{48} = ***REMOVE_ATRIBUTO#2***(***NP3***, ***orientador3*** ∘ ***NP3'***)

T_{49} = ***REMOVE_ATRIBUTO#2***(***NC4***, ***curso4*** ∘ ***NC4'***)

T_{50} = ***REMOVE_ATRIBUTO#2***(***NP5***, ***chefe*** ∘ ***NP5'***)

T_{51} = ***REMOVE_ATRIBUTO#2***(***ND6***, ***depto6*** ∘ ***ND6'***)

T_{52} = ***REMOVE_ATRIBUTO#2***(***NE7***, ***estudante*** ∘ ***NE7'***)

T_{53} = ***REMOVE_ATRIBUTO#2***(***NC7***, ***oferta7*** ∘ ***NC7'***)

T_{54} = ***REMOVE_ATRIBUTO#2***(***NP7***, ***oferta7*** ∘ ***NP7'***)

T_{55} = ***REMOVE_ATRIBUTO#2***(***ND9***, ***depto9*** ∘ ***ND9'***)

Seja $EO_5 = T_{55} \circ \ldots \circ T_{44}\ (EO_4)$. EO_5 é obtido pelas seguintes transformações:

- Substitua as restrições ***ND1≡curso1∘ND9*** e ***ND1≡professor1∘ND9*** pela restrição ***curso1∘ND9 ≡professor1∘ND9***
- Remova atributos ***NC1***, ***ND1***, ***NP1***, ***NP2***, ***NP3***, ***NC4***, ***NP5***, ***ND6***, ***NE7***, ***NC7***, ***NP7***, ***ND9***, e ***NP10***,

PASSO3.6: REMOÇÃO DAS CLASSES DERIVÁVEIS

A classe ***AUX_ENSINO*** pode ser transformada em uma classe derivável de seleção através da aplicação da seguinte transformação:

T_{56} = ***ADICIONA_SELEÇÃO#1(curso4, AUX_ENSINO, ESTUD_PÓS, curso*)***.

Seja $EO_6 = T_{56}(EO_5)$. EO_5 é obtido pelas seguintes transformações:

(i) Adiciona attributo ***curso*:ESTUD_PÓS→CURSOS*** que tem todas as restrições de ***curso4*** com exeção das seguintes restrições:

nulo é permitido para ***curso****, e

SD (CURSOS, (curso*:ESTUD_PÓS→CURSOS) = nulifica,

(porque ***SD (CURSOS, (curso4:AUX_ENSINO→CURSOS) = propaga)***

(ii) Substitua restrição ψ =***AUX_ENSINO*** ⊂ ***ESTUD_PÓS*** por

ψ' = ***AUX_ENSINO*** ⊂ ***ESTUD_PÓS***[***curso****≠"NULO"], com semântica de atualização:

SI (AUX_ENSINO, ψ') = ***rejeita*** (porque ***SI (AUX_ENSINO,*** ψ) = ***rejeita)***

SD (ESTUD_PÓS, ψ')= ***propaga*** (porque ***SD (ESTUD_PÓS,*** ψ)= ***propaga***)

SD (AUX_ENSINO, ψ') = ***nulifica***

SM (curso*, ψ) = ***propaga***

A classe ***AUX_PESQUISA*** pode ser transformada em uma classe derivável de seleção através da aplicação da seguinte transformação:

T_{57} = ***ADICIONA_SELEÇÃO#2(é_AP?, AUX_PESQUISA, ESTUD_PÓS)***.

Seja $EO_7 = T_{57}(EO_6)$. EO_7 é obtido pelas seguintes transformações:

(i) Adiciona atributos ***é_AP?:ESTUD_PÓS*** → {"***sim***", "***nulo***"}.

(ii) Substitua restrição ψ =***AUX_PESQUISA*** ⊂ ***ESTUD_PÓS*** por

ψ' = ***AUX_PESQUISA*** ⊂ ***ESTUD_PÓS***[***é_AP?***="***sim***"], com semântica de atualização:

SI (AUX_PESQUISA, ψ') = ***rejeita*** (porque ***SI (AUX_PESQUISA,*** ψ) = ***rejeita)***

SD (ESTUD_PÓS, ψ')= ***propaga*** (porque ***SD (ESTUD_PÓS,*** ψ)= ***propaga)***

SD (AUX_PESQUISA, ψ') = ***nulifica***

SM (é_AP?, ψ') = ***propaga***

As classes deriváveis ***AUX_ENSINO*** e ***AUX_PESQUISA*** podem ser removidas através das seguintes transformações:

T_{58} = ***REMOVE_CLASSE(AUX_ENSINO)***.

T_{59} = ***REMOVE_CLASSE(AUX_PESQUISA)***.

Seja $EG^* = T_{59} \circ T_{58}(EG^*)$. A estrutura do esquema conceitual global ***EG**** é mostrada na figura 6.7 e as restrições de integridade de ***EG**** estão sumarizadas na figura 6.8.

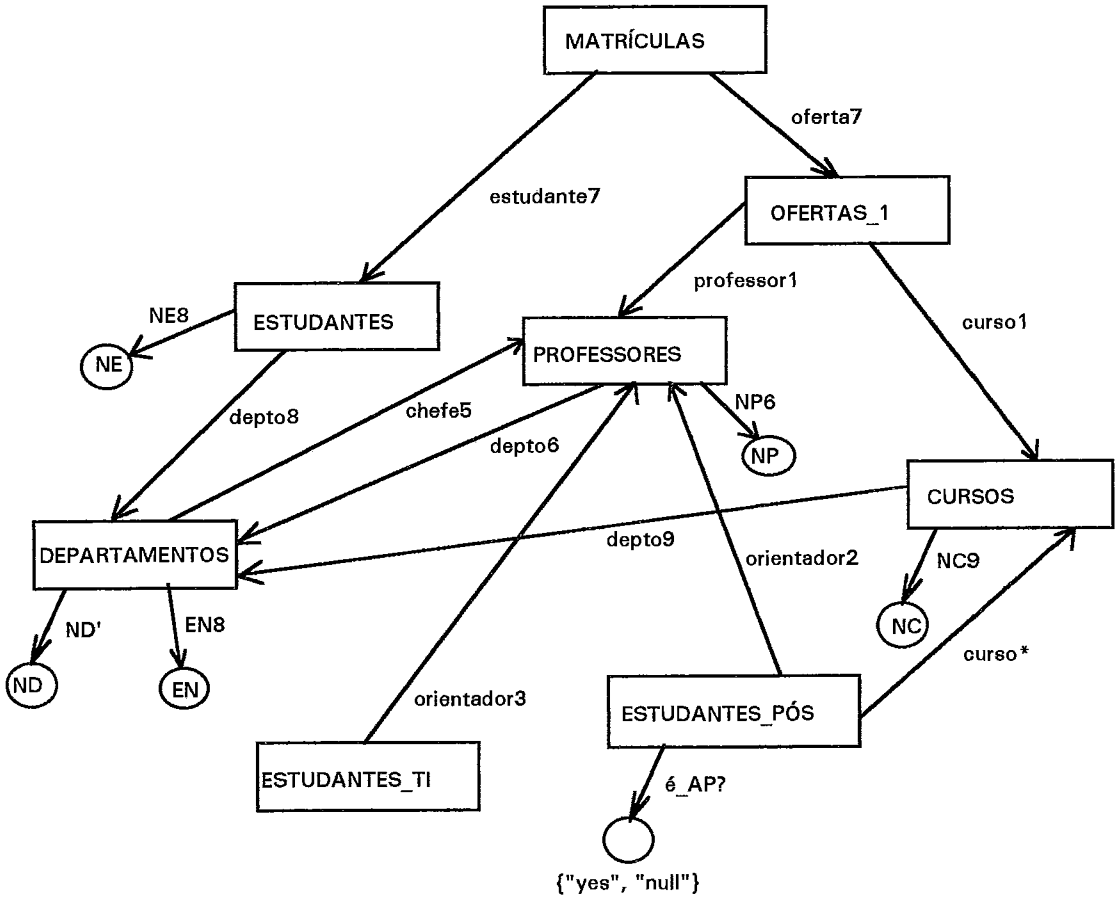


Figura 6.7: Estrutura do esquema ***EG****

Restrições de equivalência de relacionamentos:

$curso1 \circ ND9 \equiv professor1 \circ ND9$ $\qquad$ $orientador2 \equiv orientador3$

Restrições estruturais:

RE1: ***ESTUDANTES_PÓS*** $\subset$ ***ESTUDANTES***,
com semântica de atualização:
SI(ESTUDANTES_PÓS, *RE1*) = ***propaga***
SD(ESTUDANTES, *RE1*) = ***propaga***

RE2: ***ESTUDANTES_TI*** $\subset$ ***ESTUDANTES***,
com semântica de atualização:
SI(ESTUDANTES_TI, *RE2*) = ***propaga***
SD(ESTUDANTES, *RE2*) = ***propaga***

Restrições referenciais:

RR1:***curso1:OFERTAS→CURSOS_1***
com semântica de atualização:
SI(OFERTAS, *RR1*) = ***rejeita***
SD(CURSOS_1, *RR1*) = ***propaga***
SM(depto1, *RR1*) = ***rejeita***

RR2:***professor1:OFERTAS→PROF_1***
com semântica de atualização:
SI(OFERTAS, *RE3*) = ***rejeita***
SD(PROF_1, *RE3*) = ***propaga***
SM(professor1, *RE3*) = ***rejeita***

RR3:***orientador2:ESTUDANTES_PÓS→PROF_2***
com semântica de atualização:
SI(ESTUDANTES_PÓS, *RR3*) = ***rejeita***
SD(PROF_2, *RR3*) = ***rejeita***
SM(orientador2, *RR3*) = ***rejeita***

RR4:***orientador3:ESTUDANTES_TI→PROF_3***
com semântica de atualização:
SI(ESTUDANTES_TI, *RR4*) = ***rejeita***
SD(PROF_3, *RR4*) = ***rejeita***
SM(orientador3, *RR4*) = ***rejeita***

RR5:***curso4:AUX_ENSINO→CURSOS_4***
com semântica de atualização:
SI(AUX_ENSINO, *RR5*) = ***rejeita***
SD(CURSOS_4, *RR5*) = ***rejeita***
SM(curso4, *RR5*) = ***rejeita***

RR6:***chefe:DEPARTAMENTOS→PROF_5***
com semântica de atualização:
SI(DEPTOS_5, *RR6*) = ***rejeita***
SD(PROF_5, *RR6*) = ***rejeita***
SM(chefe, *RR6*) = ***rejeita***

RR7:***depto6:PROFESSORES→DEPTOS_6***
com semântica de atualização:
SI(PROFESSORES, *RR7*) = ***rejeita***
SD(DEPTOS_6, *RR7*) = ***propaga***
SM(depto6, *RR7*) = ***rejeita***

RR8:***estudante:MATRÍCULAS→ESTUD_7***
com semântica de atualização:
SI(MATRÍCULAS, *RR8*) = ***rejeita***
SD(ESTUD_7, *RR8*) = ***propaga***
SM(estudante, *RR8*) = ***rejeita***

RR9:***oferta7:MATRÍCULAS→OFERTAS_7***
com semântica de atualização:
SI(MATRÍCULAS, *RR9*) = ***rejeita***
SD(OFERTAS_7, *RR9*) = ***propaga***
SM(oferta7, *RR9*) = ***rejeita***

RR10:***depto8:ESTUDANTES→***
DEPARTAMENTOS
com semântica de atualização:
SI(ESTUDANTES, *RE3*) = ***rejeita***
SD(ESTUDANTES_PÓS, *RE3*) = ***propaga***
SM(depto1, *RE3*) = ***rejeita***

RR11:***depto9:CURSOS→DEPTOS_9***
com semântica de atualização:
SI(CURSOS, *RR11*) = ***rejeita***
SD(DEPTOS_9, *RR11*) = ***propaga***
SM(depto9, *RR11*) = ***rejeita***

RR12:***orientador10:AUX_PESQUISA→PROF_10***
com semântica de atualização:
SI(AUX_PESQUISA, *RR12*) = ***rejeita***
SD(PROF_10, *RR12*) = ***rejeita***
SM(orientador10, *RR12*) = ***propag***

Figura 6.8 - Restrições de integridade do esquema ***EG****

6.4 DEFINIÇÃO CONCEITUAL DAS VISÕES

Agora que o processo de integração das visões está concluido, precisamos definir as visões V_1 , ..., V_{10} da figura 6.1 em termos do schema conceitual global *EG** na figura 6.8. No nosso enfoque, como discutido na seção 3.3, os mapeadores de instâncias das visões e os tradutores de atualizações das visões são diretamente definidos dos mapeamentos de instâncias e dos mapementos de atualizações entre os vários esquemas intermediários gerados durante o processo de integração de visões.

No restante desta seção seja $\sigma_1, ..., \sigma_{56}$ os mapeamentos de instâncias e $\tau_1, ..., \tau_{56}$ os mapementos de atualizações associados com as transformações $T_1, ..., T_{56}$. Seja $\sigma=\sigma_1 \circ ... \circ \sigma_{56}$, $\tau=\tau_{56} \circ ... \circ \tau_1$ e $\mathcal{J} \in$ ***instâncias***(*EG**). Como $T_1, ..., T_{56}$ preservam a informação temos que $\sigma(\mathcal{J}) \in$ ***instâncias***(*EC*). De acordo com a definição 3.6, as visões V_1 , ..., V_{10} são definidas como se segue:

a) DEFINIÇÃO DE V_1

σ_{V1}:

- $\sigma_{V1}(\mathcal{J})(OFERTAS) = \sigma(\mathcal{J})(OFERTAS) = \mathcal{J}(OFERTAS)$,
- para qualquer $e \in \sigma_{V1}(\mathcal{J})(OFERTAS)$ temos:
  - $\sigma_{V1}(\mathcal{J})\ (NC1)(e) = \sigma(\mathcal{J})(NC1)(e) = \mathcal{J}(curso1 \circ NC9)(e)$
  - $\sigma_{V1}(\mathcal{J})\ (ND1)(e) = \sigma(\mathcal{J})(ND1)(e) = \mathcal{J}(curso1 \circ depto9 \circ ND')(e)$
  - $\sigma_{V1}(\mathcal{J})\ (NP1)(e) = \sigma(\mathcal{J})(NP1)(e) = \mathcal{J}(professor \circ NP6)(e)$

τ_{V1}:

- τ_{V1}("remova **o** de *OFERTAS*") = τ("remova **o** de *OFERTAS*") = "remova **o** de *OFERTAS*"
- τ_{V1}("adicione **o** a *OFERTAS*") = τ("adicione **o** a *OFERTAS*") = "adicione **o** a *OFERTAS*"
- τ_{V1}("adicione **o**(*NC1=v1, ND1=v2, NP1=v3*) a *OFERTAS*") =
  τ("adicione **o**(*NC1=v1, ND1=v2, NP1=v3*) a *OFERTAS*") =
  "adicione **o**(*curso1=c(NC9=v1, depto9=d(ND'=v2)), prof1=p(NP6=v3, depto6=d(ND'=v2))*) a *OFERTAS*"

Seja $o \in \sigma_{V1}(\mathcal{J})(OFERTAS)$, onde $c=\mathcal{J}(curso1)(o)$, $p=\mathcal{J}(professor1)(o)$, $d=\mathcal{J}(curso1 \circ depto9)(o)$

- τ_{V1}("modifique **o** com (*NC1=v*) ") = τ("modifique **o** com (*NC1=v*) ") =
  "modifique **c** com (*NC9=v*) "
- τ_{V1}("modifique **o** com (*ND1=v*) ") = τ("modifique **o** com (*ND1=v*) ") =
  "modifique **d** com (*ND'=v*) "
- τ_{V1}("modifique **o** com (*NP1=v*) ") = τ("modifique **o** com (*NP1=v*) ") =
  "modifique **p** com (*NP6=v*) "

b) DEFINIÇÃO DE V_2

σ_{V2}:

- $\sigma_{V2}(\mathcal{B})(ESTUDANTES\_PÓS) = \sigma(\mathcal{B})(ESTUDANTES\_PÓS) = \mathcal{B}(ESTUDANTES\_PÓS)$,
- para qualquer $e \in \sigma_{V2}(\mathcal{B})(ESTUDANTES\_PÓS)$ temos:
  - $\sigma_{V2}(\mathcal{B})\ (NE2)(e) = \mathcal{B}(NE8)(e)$
  - $\sigma_{V2}(\mathcal{B})\ (NP2)(e) = \mathcal{B}(orientador2 \circ NP6)(e)$

τ_{V2}:

- τ_{v2}("remova e de *ESTUDANTES_PÓS*") = "remova e de *ESTUDANTES_PÓS*"
- τ_{v2}("adicione e a *ESTUDANTES_PÓS*") = "adicione e a *ESTUDANTES_PÓS*"
- τ_{v2}("adicione e(*NE2=v1, NP2=v2*) a *ESTUDANTES_PÓS*") =
  "adicione e(*NE2=v1*, *orientador2=p(NP6=v2)* a *ESTUDANTES_PÓS*")

Seja $e \in \sigma_{V2}(\mathcal{B})(ESTUDANTES\_PÓS)$, onde $p = orientador2(e)$

- τ_{v2}("modifique e com (*NE2=v*) ") = "modifique e com (*NE8=v*) "
- τ_{v2}("modifique e com (*NP2=v*) ") = "modifique p com (*NP6=v*) "

c) DEFINIÇÃO DE V_3

σ_{V3}:

- $\sigma_{V3}(\mathcal{B})(ESTUDANTES\_TI) = \mathcal{B}(ESTUDANTES\_TI)$,
- para qualquer $e \in \sigma_{V3}(\mathcal{B})(ESTUDANTES\_TI)$ temos:
  - $\sigma_{V3}(\mathcal{B})\ (NE3)(e) = \mathcal{B}(NE8)(e)$
  - $\sigma_{V3}(\mathcal{B})\ (NP3)(e) = \mathcal{B}(orientador3 \circ NP6)(e)$

τ_{V3}:

- τ_{v3}("remova e de *ESTUDANTES_TI*") = "remova e de *ESTUDANTES_TI*"
- τ_{v3}("adicione e a *ESTUDANTES_TI*") = "adicione e a *ESTUDANTES_TI*"
- τ_{v3}("adicione e(*NE3=v1, NP3=v2*) a *ESTUDANTES_TI*") =
  "adicione e(*NE3=v1*, *orientador3=p(NP6=v2)* a *ESTUDANTES_TI*")

Seja $e \in \sigma_{V3}(\mathcal{B})(ESTUDANTES\_TI)$, onde $p = \mathcal{B}(orientador3)(e)$

- τ_{v3}("modifique e com (*NE3=v*) ") = "modifique e com (*NE8=v*) "
- τ_{v3}("modifique e com (*NP3=v*) ") = "modifique p com (*NP6=v*) "

d) DEFINIÇÃO DE V_4

σ_{V4}:

- $\sigma_{V4}(\mathcal{B})(AUX\_ENSINO) = \{\ e \mid e \in \mathcal{B}(ESTUDANTES\_PÓS)$ e $\mathcal{B}(curso4)(e) \neq$ nulo$\}$,
- para qualquer $e \in \sigma_{V4}(\mathcal{B})(AUX\_ENSINO)$ temos:
  - $\sigma_{V4}(\mathcal{B})\ (NE4)(e) = \mathcal{B}(NE8)(e)$
  - $\sigma_{V4}(\mathcal{B})\ (NC4)(e) = \mathcal{B}(curso4 \circ NC9)(e)$

τ_{V4}:

- τ_{v4}("remova e de *AUX_ENSINO*") = "modifique e com (*curso4 =nulo*)"
- τ_{v4}("adicione e(*NE4=v1, NC4=v2*) a *AUX_ENSINO*") =
  { Se $e \notin \mathcal{B}(ESTUDANTES\_PÓS)$ então "aborta a transação"
  senão "modifique e com (*curso4 =c(NC9=v2)*)"}

Seja $e \in \sigma_{V4}(\mathcal{B})(AUX\_ENSINO)$, onde $c = \mathcal{B}(curso4)(e)$

- τ_{v4}("modifique e com (*NE4=v*) ") = "modifique e com (*NE8=v*) "
- τ_{v4}("modifique e com (*NP4=v*) ") = "modifique c com (*NC9=v*) "

e) DEFINIÇÃO DE V_5

σ_{V5}:

- $\rho_{V5}(\mathcal{B})(DEPTOS\_5) = \mathcal{B}(DEPARTAMENTOS)$,
- para qualquer $e \in \sigma_{V5}(\mathcal{B})(DEPTOS\_5)$ temos:
  - $\sigma_{V5}(\mathcal{B})\ (ND')(e) = \mathcal{B}(ND')(e)$
  - $\sigma_{V5}(\mathcal{B})\ (NP5)(e) = \mathcal{B}(chefe5 \circ NP6)(e)$

τ_{V5}:

- τ_{V5}("remova **e** de ***DEPTOS_5***") = "remova **e** de ***DEPARTAMENTOS***"
- τ_{V5}("adicione **e** a ***DEPTOS_5***") = "adicione **e** a ***DEPARTAMENTOS***"
- τ_{V5}("adicione **e**(***NE5=v1, NP5=v2***) a ***DEPTOS_5***") =
  "adicione **e**(***NE5=v1, chefe5=p(NP6=v2)***) a ***DEPARTAMENTOS***")

Seja $e \in \sigma_{V5}(\mathcal{B})(DEPARTAMENTOS)$, onde $p = \mathcal{B}(chefe5)(e)$

- τ_{V5}("modifique **e** com (***NE5=v***) ") = "modifique **e** com (***NE8=v***) "
- τ_{V5}("modifique **e** com (***NP5=v***) ") = "modifique **p** com (***NP6=v***) "

f) DEFINIÇÃO DE V_6

σ_{V6}:

- $\sigma_{V6}(\mathcal{B})(PROFESSORES) = \mathcal{B}(PROFESSORES)$,
- para qualquer $e \in \sigma_{V6}(\mathcal{B})(PROFESSORES)$ temos:
  - $\sigma_{V6}(\mathcal{B})\ (NP6)(e) = \mathcal{B}(NP6)(e)$
  - $\sigma_{V6}(\mathcal{B})\ (ND6)(e) = \mathcal{B}(depto6 \circ ND')(e)$

τ_{V6}:

- τ_{V6}("remova **e** de ***PROFESSORES***") = "remova **e** de ***PROFESSORES***"
- τ_{V6}("adicione **e** a ***PROFESSORES***") = "adicione **e** a ***PROFESSORES***"
- τ_{V6}("adicione **e**(***NP6=v1, ND6=v2***) a ***PROFESSORES***") =
  "adicione **e**(***NP6=v1, depto6=p(ND6=v2)***) a ***PROFESSORES***")

Seja $e \in \sigma_{V6}(\mathcal{B})(PROFESSORES)$, onde $d = \mathcal{B}(depto6)(e)$

- τ_{V6}("modifique **e** com (***NP6=v***) ") = "modifique **e** com (***NP6=v***) "
- τ_{V6}("modifique **e** com (***ND6=v***) ") = "modifique **d** com (***ND'=v***) "

g) DEFINIÇÃO DE V_7

σ_{V7}:

- $\sigma_{V7}(\mathcal{B})(MATRÍCULAS) = \mathcal{B}(MATRÍCULAS)$,
- para qualquer $e \in \sigma_{V7}(\mathcal{B})(MATRÍCULAS)$ temos:
  - $\sigma_{V7}(\mathcal{B})\ (NE7)(e) = \mathcal{B}(estudante7 \circ NE8)(e)$
  - $\sigma_{V7}(\mathcal{B})\ (NC7)(e) = \mathcal{B}(oferta7 \circ curso1 \circ NC9)(e)$
  - $\sigma_{V7}(\mathcal{B})\ (NP7)(e) = \mathcal{B}(oferta7 \circ professor1 \circ NP6)(e)$

τ_{V7}:

- τ_{V7}("remova **m** de ***MATRÍCULAS***") = "remova **m** de ***MATRÍCULAS***"
- τ_{V7}("adicione **m** a ***MATRÍCULAS***") = "adicione **m** a ***MATRÍCULAS***"
- τ_{V7}("adicione **m**(***NE7=v1, NC7=v2, NP7=v3***) a ***MATRÍCULAS***") =
  "adicione **m**(***estudante7=e(NE8=v1), oferta7=o(curso1=c(NC9=v2), professor1=d(NP6=v3))***) a ***MATRÍCULAS***"

Seja $m \in \sigma_{V7}(\mathcal{B})(MATRÍCULAS)$, onde $e = \mathcal{B}(estudante7)(m)$, $p = \mathcal{B}(oferta7 \circ professor1)(m)$, e

$c = \mathcal{B}(oferta7 \circ curso1)(m)$

- τ_{V7}("modifique **m** com $(NE7=v)$ ") = "modifique **e** com $(NE8=v)$ "
- τ_{V7}("modifique **m** com $(NP7=v)$ ") = "modifique p com $(NP6=v)$ "
- τ_{V7}("modifique **m** com $(NC7=v)$ ") = "modifique **c** com $(NC9=v)$ "

h) DEFINIÇÃO DE V_8

σ_{V8}:

- $\sigma_{V8}(\mathcal{B})(ESTUDANTES) = \mathcal{B}(ESTUDANTES)$,
- para qualquer $e \in \sigma_{V8}(\mathcal{B})(ESTUDANTES)$ temos:
  - $\sigma_{V8}(\mathcal{B})\ (NE8)(e) = \mathcal{B}(NE8)(e)$
  - $\sigma_{V8}(\mathcal{B})\ (depto8)(e) = \mathcal{B}(depto8)(e)$
- $\sigma_{V8}(\mathcal{B})(DEPTOS\_8) = \mathcal{B}(DEPARTAMENTOS)$,
- para qualquer $e \in \sigma_{V8}(\mathcal{B})(DEPTOS\_8)$ temos:
  - $\sigma_{V8}(\mathcal{B})\ (ND8)(e) = \mathcal{B}(ND8)(e)$
  - $\sigma_{V8}(\mathcal{B})\ (EN8)(e) = \mathcal{B}(EN8)(e)$

τ_{V8}:

- τ_{V8}("remova **e** de *ESTUDANTES*") = "remova **e** de *ESTUDANTES*"
- τ_{V8}("adicione **e** a *ESTUDANTES*") = "adicione **e** a *ESTUDANTES*"
- τ_{V8}("adicione **e**$(NE8=v1,\ depto8=v2)$ a *ESTUDANTES*") =
  "adicione **e**$(NE8=v1,\ depto8=v2)$ a *ESTUDANTES*")

Seja $e \in \sigma_{V8}(\mathcal{B})(ESTUDANTES)$

- τ_{V8}("modifique **e** com $(NE8=v)$ ") = "modifique **e** com $(NE8=v)$ "
- τ_{V8}("modifique **e** com $(depto8=v)$ ") = "modifique **e** com $(depto8=v)$ "
- τ_{V8}("remova **d** de *DEPTOS_8*") = "remova **d** de *DEPARTAMENTOS*"
- τ_{V8}("adicione **d** a *DEPTOS_8*") = "adicione **d** a *DEPARTAMENTOS*"
- τ_{V8}("adicione **d**$(ND8=v1,\ EN8=v2)$ a *DEPTOS_8*") =
  "adicione **d**$(ND8=v1,\ EN8=v2)$ a *DEPARTAMENTOS*"

Seja $d \in \sigma_{V8}(\mathcal{B})(DEPTOS\_8)$

- τ_{V8}("modifique **d** com $(EN8=v)$ ") = "modifique **d** com $(EN8=v)$ "
- τ_{V8}("modifique **d** com $(ND8=v)$ ") = "modifique **d** com $(ND8=v)$ "

i) DEFINIÇÃO DE V_9

σ_{V9}:

- $\sigma_{V9}(\mathcal{B})(CURSOS) = \mathcal{B}(CURSOS)$,
- para qualquer $e \in \sigma_{V9}(\mathcal{B})(CURSOS)$ temos:
  - $\sigma_{V9}(\mathcal{B})\ (NC9)(e) = \mathcal{B}(NC9)(e)$
  - $\sigma_{V9}(\mathcal{B})\ (ND9)(e) = \mathcal{B}(depto9 \circ ND')(e)$

τ_{V9}:

- τ_{V9}("remova **e** de *CURSOS*") = "remova **e** de *CURSOS*"
- τ_{V9}("adicione **e** a *CURSOS*") = "adicione **e** a *CURSOS*"
- τ_{V9}("adicione **e**$(NP9=v1,\ ND9=v2)$ a *CURSOS*") =
  "adicione **e**$(NP9=v1,\ depto9=p(ND9=v2)$ a *CURSOS*")

Seja $e \in \sigma_{V9}(\mathcal{B})(CURSOS)$, onde $d = \mathcal{B}(depto9)(e)$

- τ_{V9}("modifique **e** com $(NP9=v)$ ") = "modifique **e** com $(NP9=v)$ "
- τ_{V9}("modifique **e** com $(ND9=v)$ ") = "modifique **d** com $(ND'=v)$ "

j) DEFINIÇÃO DE V_{10}

σ_{V10}:

- $\sigma_{V10}(\mathcal{B})(\mathit{AUX\_PESQUISA}) = \{\, e \mid e \in \mathcal{B}(\mathit{ESTUDANTES\_PÓS})$ e $\mathcal{B}(\mathit{é\_AP?})(e) = \mathit{sim} \,\}$,
- para qualquer $e \in \sigma_{V10}(\mathcal{B})(\mathit{AUX\_PESQUISA})$ temos:
  - $\sigma_{V10}(\mathcal{B})\ (\mathit{NE10})(e) = \mathcal{B}(\mathit{NE8})(e)$
  - $\sigma_{V10}(\mathcal{B})\ (\mathit{NP10})(e) = \mathcal{B}(\mathit{orientador2} \circ \mathit{NE1})(e)$

τ_{V10}:

- τ_{V10}("remova e de *AUX_PESQUISA*") = "modifique e com (*é_AP?=nulo*)"
- τ_{V10}("adicione e(*NE10=v1*, *NP10=v2*) a *AUX_PESQUISA*") =

  Se $e \notin \mathcal{B}(\mathit{ESTUDANTES\_PÓS})$ então "aborta a transação"

  senão "modifique e com (*é_AP?=sim*)"

Seja $e \in \sigma_{V10}(\mathcal{B})(\mathit{AUX\_PESQUISA})$, onde $p = \mathcal{B}(\mathit{orientador10})(e)$

- τ_{V10}("modifique e com (*NE10=v*) ") = "modifique e com (*NE8=v*) "
- τ_{V10}("modifique e com (*NP10=v*) ") = "modifique **p** com (*NP6=v*) "

Como as transformações $\mathbf{T}_1, ..., \mathbf{T}_{56}$ preservam a informação e a semântica das atualizações, podemos garantir a validade das traduções das operações de atualização das visões definidas acima. Como exemplo, mostraremos a seguir que a tradução da operação de atualização u="remova o de *OFERTAS*" da visão $\mathbf{V}_1$, definida por:

(6.1) $\tau_{v1}(u)$ = "remova o de *OFERTAS*"

é válida.

Seja u_τ = "remova o de *OFERTAS*". Da definição (3.7), $\tau_{v1}(u)$ é uma tradução válida se para todo $\mathcal{B} \in$ *instâncias*(*EG**), $u(\sigma(\mathcal{B})) = \sigma(u_\tau(\mathcal{B}))$. Seja u_τ = "remova o de *OFERTAS*". Nós temos que considerar todos os diferentes casos em que u_τ tem diferentes tipos de "efeitos colaterais". Como a restrição referencial ψ=*oferta:MATRÍCULAS*→*OFERTAS* é a única restrição de *EG** que pode ser violada por u_τ, nós temos somente que considerar os casos onde ψ não seja violada por u_τ e os casos em que ψ seja violada por u_τ. Lembre-se que *EC* e *EG** têm equivalentes capacidade de informação, porque $\mathbf{T}_1, ..., \mathbf{T}_{56}$ preservam a informação.

<u>Caso 1</u>: Considere $\mathcal{B}_G \in$ *instâncias*(*EG**) tal que $\mathcal{B}_G(oferta7^{-1})(o) = \varnothing$.
Seja $\mathcal{B}_C = \sigma(\mathcal{B}_G)$. Logo, não existe nenhum m em $\mathcal{B}_C$(*MATRÍCULAS*) tal que $\mathcal{B}_C(NC7)(m) = \mathcal{B}_C(NC1)(o)$ e $\mathcal{B}_C(NP7)(m) = \mathcal{B}_C(NP1)(o)$.

- Cálculo de $u_\tau(\mathcal{B}_G)$

Seja $\mathcal{B}_G{}^1 = efeito(u_\tau, \mathcal{B}_G)$. $\mathcal{B}_G{}^1$ é idêntica a $\mathcal{B}_G$ exceto que:

(6.1) $\mathcal{B}_G{}^1(OFERTAS) = \mathcal{B}_G(OFERTAS) - \{o\}$

Como *relevantes*$(u_\tau) = \{\psi\}$, nós temos:

$$co\text{-}efeito^*(u_\tau, \mathcal{B}_G{}^1) = co\text{-}efeito(u_\tau, \mathcal{B}_G{}^1, \psi).$$

Como a remoção de o em *OFERTAS* não viola ψ, *EC*="continue" é o "efeito colateral" requerido pela semântica de atualização de u com respeito a ψ. Logo,

$$co\text{-}efeito(u_\tau, \mathcal{B}_G{}^1, \psi) = \mathcal{B}_G{}^1$$

Logo,

$$co\text{-}efeito^*(u_\tau, \mathcal{B}_G{}^1) = \mathcal{B}_G{}^1$$

Portanto,

$$u_\tau(\mathcal{B}_G) = co\text{-}efeito^*(u_\tau, efeito(u, \mathcal{B})) = \mathcal{B}_G{}^1.$$

De (6.1) nós temos que $\mathcal{B}_G{}^1$ é idêntica a $\mathcal{B}$ exceto que:

(6.2) $\mathcal{B}_G{}^1(OFERTAS) = \mathcal{B}_G(OFERTAS) - \{o\}$

- Cálculo de $u(\mathcal{B}_C)$

Seja $\mathcal{B}_C{}^1$= *efeito*$(u, \mathcal{B}_C)$. $\mathcal{B}_C{}^1$ é idêntica a $\mathcal{B}_C$ exceto que:

(6.3) $\mathcal{B}_C{}^1(\textit{OFERTAS})=\mathcal{B}_C(\textit{OFERTAS})$ - $\{o\}$

Como *relevantes*$(u)=\{\psi= \textit{MATRÍCULAS}[NC7,NP7]\subset \textit{OFERTAS}[NC1,NP1]\}$, temos:

$$\textit{co-efeito*}(u,\ \mathcal{B}_C{}^1) = \textit{co-efeito}(u,\ \mathcal{B}_C{}^1, \psi).$$

Como a deleção de o em ***OFERTAS*** não viola ψ, ***EC***="continue" é o "efeito colateral" requerido pela semântica de atualização de u com respeito a ψ . Logo,

$$\textit{co-efeito}(u,\ \mathcal{B}_C{}^1, \psi) = \mathcal{B}_C{}^1$$

Logo,

$$\textit{co-efeito*}(u,\ \mathcal{B}_C{}^1) = \mathcal{B}_C{}^1$$

Portanto,

$$u(\mathcal{B}_C) = \textit{co-efeito*}(u, \textit{efeito}(u, \mathcal{B})) = \mathcal{B}_C{}^1.$$

De (6.3) nós temos que $\mathcal{B}_C{}^1$ é idêntica a $\mathcal{B}_C$ exceto que:

(6.4) $\mathcal{B}_C{}^1(\textit{OFERTAS})=\mathcal{B}_C(\textit{OFERTAS})$ - $\{o\}$

Como $\sigma(\mathcal{B}_G{}^1)(\textit{OFERTAS})=(\mathcal{B}_G{}^1)(\textit{OFERTAS})$

de (6.2) nós temos:

(6.5) $\sigma(\mathcal{B}_G{}^1)(\textit{OFERTAS}) = \mathcal{B}_G(\textit{OFERTAS})$ - $\{o\}$

Mas $\mathcal{B}_G(\textit{OFERTAS}) = \mathcal{B}_C(\textit{OFERTAS})$. Logo, de (6.5) e (6.4) nós temos:

(6.6) $\mathcal{B}_C{}^1(\textit{OFERTAS}) = \sigma(\mathcal{B}_G{}^1)(\textit{OFERTAS})$

De (6.6) nós temos $\mathcal{B}_C{}^1 = \sigma(\mathcal{B}_G{}^1)$. Portanto, $u(\sigma(\mathcal{B}_G)) = \sigma(\tau_{V1}(u)(\mathcal{B}_G))$.

<u>Caso 2</u>: Considere $\mathcal{B}_G \in$ *instâncias*(EG^*) tal que $\mathcal{B}_G(\textit{oferta7}^{-1})(o)=\{m_1, ..., m_n\}$.
Let $\mathcal{B}_C=\sigma(\mathcal{B}_G)$. Assim sendo, $\{m_1, ..., m_n\}$ é o conjunto de todas as entidades em $\mathcal{B}_C(\textit{MATRÍCULAS})$ onde $(\mathcal{B}_C(NC7)(m_i)=\mathcal{B}_C(NC1)(o)$ e $\mathcal{B}_C(NP7)(m_i)=\mathcal{B}_C(NP1)(o)$, $1\leq i\leq n)$.

- Cálculo de $u_\tau(\mathcal{B}_G)$

Seja $\mathcal{B}_G{}^1$= *efeito*$(u_\tau, \mathcal{B}_G)$. $\mathcal{B}_G{}^1$ é idêntica a $\mathcal{B}_G$ exceto que:

(6.7) $\mathcal{B}_G{}^1(\textit{OFERTAS})=\mathcal{B}_G(\textit{OFERTAS})$ - $\{o\}$

Como *relevantes*$(u_\tau)=\{\psi\}$, nós temos:

$$\textit{co-efeito*}(u_\tau,\ \mathcal{B}_G{}^1) = \textit{co-efeito}(u_\tau,\ \mathcal{B}_G{}^1, \psi).$$

Como ***EC***=$\{u_1$="remova m_1 de ***MATRÍCULAS***", ..., u_n= "remova m_n de ***MATRÍCULAS***"$\}$ é o "efeito colateral" requerido pela semântica de atualização de u com respeito a ψ (*SD*(***OFERTAS***, ψ) = ***propaga***), nós temos:

$$\textit{co-efeito}(u_\tau,\ \mathcal{B}_G{}^1, \psi) = u_n \circ ... \circ u_1(\mathcal{B}_G{}^1)$$

- Cálculo de $u_k(\mathcal{B}_G{}^k)$ onde $\mathcal{B}_G{}^k = u_{k-1}(\mathcal{B}_G{}^{k-1})$, para $2<k\leq n$.

Seja $\mathcal{B}_G{}^{k+1}$= *efeito*$(u_k, \mathcal{B}_G{}^k)$. $\mathcal{B}_G{}^{k+1}$ é idêntica a $\mathcal{B}_G{}^k$ exceto que:

(6.8) $\mathcal{B}_G{}^{k+1}(\textit{MATRÍCULAS})=\mathcal{B}_G{}^k(\textit{MATRÍCULAS})$ - $\{m_k\}$

Como $relevantes(u_k)=\varnothing$, nós temos:

$$co\text{-}efeito^*(u_k, \mathcal{B}_G^{k+1}) = \mathcal{B}_G^{k+1}.$$

Portanto,

$$u_k(\mathcal{B}_G^{k}) = co\text{-}efeito^*(u_k, efeito(u_k, \mathcal{B}_G^{k})) = \mathcal{B}_G^{k+1}.$$

(Continuando o cálculo de $u_\tau(\mathcal{B}_G)$)

$$co\text{-}efeito^*(u_\tau, \mathcal{B}_G^{1}) = u_n \circ ... \circ u_1 (\mathcal{B}_G^{1}) = \mathcal{B}_G^{n+1}.$$

Portanto,

$$u_\tau(\mathcal{B}_G) = co\text{-}efeito^*(u_\tau, efeito(u, \mathcal{B})) = \mathcal{B}_G^{n+1}.$$

De (6.7) e (6.8) nós temos que $\mathcal{B}_G^{n+1}$ é idêntica a $\mathcal{B}_G$ exceto que:

(6.9) $\mathcal{B}_G^{n+1}(OFERTAS)=\mathcal{B}_G(OFERTAS) - \{o\}$

$\mathcal{B}_G^{n+1}(MATRÍCULAS)=\mathcal{B}_G(MATRÍCULAS) - \{m_1, ..., m_n\}$

- Cálculo de $u(\mathcal{B}_C)$

Seja $\mathcal{B}_C^{1}= efeito(u, \mathcal{B}_C)$. $\mathcal{B}_C^{1}$ é idêntica a $\mathcal{B}_C$ exceto que:

(6.10) $\mathcal{B}_C^{1}(OFERTAS)=\mathcal{B}_C(OFERTAS) - \{o\}$

Como $relevantes(u)=\{\psi=MATRÍCULAS[NC7,NP7]\sqsubset OFERTAS[NC1,NP1]\}$, temos:

$$co\text{-}efeito^*(u, \mathcal{B}_C^{1}) = co\text{-}efeito(u, \mathcal{B}_C^{1}, \psi).$$

Como $EC=\{u_1=$ "remova m_1 de $MATRÍCULAS$", ..., $u_n=$ "remova m_n de $MATRÍCULAS$"$\}$ é o "co-efeito" requerido pela semântica de atualização de u com respeito a ψ ($SD(OFERTAS, \psi) = propaga$), nós temos:

$$co\text{-}efeito(u, \mathcal{B}_C^{1}, \psi) = u_n \circ ... \circ u_1 (\mathcal{B}_C^{1})$$

- Cálculo de $u_k(\mathcal{B}_C^{k})$ where $\mathcal{B}_G^{k} = u_{k-1}(\mathcal{B}_C^{k-1})$, para $2<k\leq n$.

Seja $\mathcal{B}_C^{k+1}= efeito(u_k, \mathcal{B}_C^{k})$. $\mathcal{B}_C^{k+1}$ é idêntica a $\mathcal{B}_C^{k}$ exceto que:

(6.11) $\mathcal{B}_C^{k+1}(MATRÍCULAS)=\mathcal{B}_C^{k}(MATRÍCULAS) - \{m_k\}$

Como $relevantes(u_k)=\varnothing$, nós temos:

$$co\text{-}efeito^*(u_k, \mathcal{B}_C^{k+1}) = \mathcal{B}_C^{k+1}.$$

Portanto,

$$u_k(\mathcal{B}_C^{k}) = co\text{-}efeito^*(u_k, efeito(u_k, \mathcal{B}_C^{k})) = \mathcal{B}_C^{k+1}.$$

(Continuando o cálculo de $u(\mathcal{B}_C)$)

$$co\text{-}efeito^*(u, \mathcal{B}_C^{1}) = u_n \circ ... \circ u_1 (\mathcal{B}_C^{1}) = \mathcal{B}_C^{n+1}.$$

Portanto,

$$u(\mathcal{B}_C) = co\text{-}efeito(u, efeito(u, \mathcal{B}_C)) = \mathcal{B}_C^{n+1}.$$

De (6.10) e (6.11) nós temos que $\mathcal{B}_C^{n+1}$ é idêntica a $\mathcal{B}_C$ exceto que:

(6.12) $\mathcal{B}_C^{n+1}(OFERTAS)=\mathcal{B}_C(OFERTAS) - \{o\}$

$\mathcal{B}_C^{n+1}(MATRÍCULAS)=\mathcal{B}_C(MATRÍCULAS) - \{m_1, ..., m_n\}$

Como $\sigma(\mathcal{B}_G^{n+1})(OFERTAS)=(\mathcal{B}_G^{n+1})(OFERTAS)$ e

$\boldsymbol{\sigma}(\mathscr{B}_G{}^{n+1})(\textit{MATRÍCULAS})=(\mathscr{B}_G{}^{n+1})(\textit{MATRÍCULAS})$,

de (6.9) nós temos:

(6.13) $\boldsymbol{\sigma}(\mathscr{B}_G{}^{n+1})(\textbf{\textit{OFERTAS}}) = \mathscr{B}_G(\textbf{\textit{OFERTAS}}) - \{\boldsymbol{o}\}$

$\boldsymbol{\sigma}(\mathscr{B}_G{}^{n+1})(\textit{MATRÍCULAS}) = \mathscr{B}_G(\textit{MATRÍCULAS}) - \{e_1,....,e_n\}$

Mas $\mathscr{B}_G(\textbf{\textit{OFERTAS}}) = \mathscr{B}_C(\textbf{\textit{OFERTAS}})$ e $\mathscr{B}_G(\textit{MATRÍCULAS}) = \mathscr{B}_C(\textit{MATRÍCULAS})$. Logo, de (6.12) e (6.13) temos:

(6.14) $\mathscr{B}_C{}^{n+1}(\textbf{\textit{OFERTAS}}) = \boldsymbol{\sigma}(\mathscr{B}_G{}^{n+1})(\textbf{\textit{OFERTAS}})$

$\mathscr{B}_C{}^{n+1}(\textit{MATRÍCULAS}) = \boldsymbol{\sigma}(\mathscr{B}_G{}^{n+1})(\textit{MATRÍCULAS})$

De (6.14) temos $\mathscr{B}_C{}^{n+1} = \boldsymbol{\sigma}(\mathscr{B}_G{}^{n+1})$. Portanto, $\boldsymbol{u}(\boldsymbol{\sigma}(\mathscr{B}_G)) = \boldsymbol{\sigma}(\boldsymbol{\tau}_{v1}(\boldsymbol{u})(\mathscr{B}_G))$.

Assim sendo, podemos concluir que a remoção de ***o*** na classe ***OFERTAS*** do esquema global ***EG**** alcança o mesmo estado (i.e., tem o mesmo efeito) como se o usuário tivesse executado a remoção de ***o*** na classe ***OFERTAS*** do esquema combinado ***EC***.

7. PROJETO DO ESQUEMA ORIENTADO A OBJETOS

Esse capítulo é organizado com se segue. Na Seção 7.1, introduzimos o paradigma orientado a objetos. Na Seção 7.2, descrevemos como projetar uma representação orientada a objetos para um esquema conceitual obtido com a nossa metodologia. Na seção 7.3, descrevemos como traduzir definições conceituais de visões em definições orientadas a objetos.

7.1 O PARADIGMA DA ORIENTAÇÃO A OBJETOS

O paradigma da orientação a objetos [Banerjee87, Fishman87, Maier85] é baseado no conceito de objeto. Um objeto encapsula dados e operações (chamadas de métodos em alguns modelos orientados a objetos). Todo objeto tem um tipo (chamado de classe em alguns modelos orientados a objetos) e o objeto é conhecido como uma instância do tipo. Tipos servem como moldes para as suas instâncias e a extensão de um tipo é o conjunto de suas instâncias. O tipo de um objeto especifica a interface (protocolo) e a estrutura de suas instâncias. A interface de um objeto consiste de um conjunto de mensagens que podem ser enviadas para o objeto, e é a única maneira através da qual pode-se manipular o objeto. Assim, quando uma mensagem é enviada para um objeto, a operação correspondente é executada. A estrutura de um objeto representa o estado interno do objeto, e é representada por um conjunto de valores de *variáveis de instâncias*. Variáveis de instâncias são como nomes de campos em um registro ou índices em um array; seus valores são referências para outros objetos.

No paradigma da orientação a objetos, os tipos são organizados em uma hierarquia de tipos, onde um tipo pode ser declarado como um subtipo de outros tipos. Nesse caso, o subtipo é uma especialização de seus supertipos, i.e., o subtipo herda as operações e varíaveis de instâncias definidas por seus supertipos, podendo introduzir operações adicionais e variáveis de instâncias.

7.2 PROJETO DO ESQUEMA ORIENTADO A OBJETOS

Um esquema orientado a objetos é uma tupla ***EO***=($\mathcal{H}$, $\mathcal{C}$) onde $\mathcal{H}$ é uma hierarquia de tipos de entidades, $\mathcal{C}$ é um conjunto de objetos *coleções*. Nos modelos orientados a objetos [Banerjee87, Fishman87, Maier85], entidades são representadas como objetos. A hierarquia de tipos, $\mathcal{H}$, de um esquema ***EO*** define o conjunto de tipos que servirão de molde para gerar as entidades do banco de dados. O conjunto de coleções $\mathcal{C}$ de um esquema ***EO*** são objetos que servem para colecionar as entidades em conjuntos, de forma que consultas possam ser executadas sobre esses conjuntos. Em alguns modelos orientados a objetos as entidades são agrupadas pelos seus tipos. [Maier86] discute os problemas deste enfoque.

A vantagem do encapsulamento de dados e operações, suportado pelo paradigma da orientação a objetos, é que as operações podem prevenir o mapeamento de um estado consistente em um estado inconsistente. No enfoque orientado a objetos as operações são responsáveis pela manutenção de suas restrições relevantes [Urban89, Vidal90]. Uma restrição é relevante para uma dada operação quando o efeito da operação pode fazer com que a restrição seja violada. Usando esse enfoque para manutenção das restrições de integridade, o processo de especificação de uma operação consiste de dois passos [Chung87]. O primeiro passo é selecionar todas as restrições relevantes para a operação. O segundo é definir os procedimentos para manutenção das restrições relevantes. Como as operações afetam o banco de dados de maneiras diferentes, os procedimentos de manutenção para uma dada restrição variam de operação para operação. A vantagem do uso deste enfoque para a manutenção das restrições de integridade é que uma restrição só é validada quando for preciso, e como a validação de uma restrição é feita pelos procedimentos de manutenção, essa pode ser feita de forma bem mais eficiente do que uma validação ingênua da restrição.

No nosso enfoque, o projeto de um esquema orientado a objetos ***OS***=($\mathcal{H}$, $\mathcal{C}$) para um dado esquema conceitual global ***EG*** =($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$), é dividido em dois passos:

1) ***Definição da hierarquia de tipos de entidade*** ($\mathcal{H}$), e,

2) ***Definição do conjunto de coleções*** ($\mathcal{C}$),

que são discutidos nas próximas seções.

7.2.1 DEFINIÇÃO DA HIERARQUIA DE TIPOS

Para se definir a hierarquia de tipos para um esquema conceitual **EG**, primeiro determina-se o conjunto de todos os possíveis tipos conceituais definidos pelas classes em ***EG***. Para isso, deve-se determinar todos os conjuntos de classes de **EG** nos quais a mesma entidade pode aparecer simultaneamente. Dizemos que um conjunto de classes $\{c_1, ..., c_n\}$ de **EG** tem *extensão vazia* se e somente se não puder existir uma entidade que seja membro das classes $c_1, ..., c_n$ e de mais nenhuma outra classe em **EG**. Caso contrário, o conjunto de classes $\{c_1, ..., c_n\}$ tem uma *extensão não vazia*. Por exemplo, os conjuntos de classes {***ESTUDANTES_PÓS***}, {***ESTUDANTES_PÓS, PROFESSORES***} no esquema conceitual da Figura 6.7 têm extensões vazias, uma vez que todos os estudantes de pós graduação são também estudantes. Os conjuntos de classes do esquema conceitual da Figura 6.7 com extensão não vazia são mostrados na tabela 7.1.

Tabela 7.1: Tipos de entidades para o esquema conceitual da Figura 6.7

Conjunto de classes com extensão não vazia	*Tipo*
{***ESTUDANTES***}	***ESTUDANTE***
{***ESTUDANTES, ESTUDANTES_PÓS***}	***ESTUDANTE_PÓS***
{***ESTUDANTES, ESTUDANTES_TI*** }	***ESTUDANTE_TI***
{***ESTUDANTES, ESTUDANTES_PÓS, ESTUDANTES_TI***}	***ESTUDANTE_PÓS_TI***
{***PROFESSORES***}	***PROFESSOR***
{***CURSOS***}	***CURSO***
{***DEPARTAMENTOS***}	***DEPARTAMENTO***
{***OFERTAS***}	***OFERTA***
{***MATRÍCULAS***}	***MATRÍCULA***

Para se definir os tipos de entidades para um esquema conceitual ***EG***=$(\mathcal{C}, \mathcal{A}, \mathcal{R})$, deve-se definir um tipo ***T*** para cada conjunto de classes $\{c_1, ..., c_n\}$, tal que $\{c_1, ..., c_n\} \subseteq \mathcal{C}$ e $\{c_1, ..., c_n\}$tenha extensão não vazia. O conjunto de tipos de entidades associados com cada conjunto de classes com extensão não vazia do esquema conceitual da Figura 6.7 é mostrado na tabela 7.1. Se o tipo ***T*** está associado com o conjunto de classes $\{c_1, ..., c_n\}$ então ***classes***(***T***)=$\{c_1, ..., c_n\}$. Os tipos de entidades de um esquema orientado a objetos estão organizados em uma hierarquia de tipos de entidades como se segue:

Definição 7.1: A hierarquia de tipos de um esquema orientado a objetos **EO** é um grafo acíclico direcionado $\mathcal{H}=(\mathcal{T},\mathcal{A})$, onde $\mathcal{T}$ é um conjunto de tipos de entidades e $\mathcal{A}$ é um conjunto de arcos tais que um arco *a* de T_1 para T_2, denominado de $a=<T_1,T_2>$ (T_1 é um subtipo de T_2), está em $\mathcal{A}$ sss

(i) ***classes***$(T_1)\subset$***classes***(T_2), e

(ii) não existir nenhum $T_3 \in \mathcal{T}$ tal que ***classes***$(T_1)\subset$***classes***(T_3) e ***classes***$(T_3)\subset$ ***classes***(T_2).

A Figura 7.2 mostra a hierarquia de tipos para o esquema conceitual da Figura 6.7.

Uma entidade ***e*** é uma instância do tipo ***T***, onde ***classes***(***T***)=$\{C_1, ..., C_n\}$), se e somente se ***e*** for membro das classes $C_1, ..., C_n$ e não for membro de nenhuma outra classe em ***EG***. Por exemplo, na figura 7.1, como a entidade ***Maria*** é membro somente das classes ***ESTUDANTES***, ***ESTUDANTES_PÓS*** e ***ESTUDANTES_TI***, ela é uma instância do tipo ***ESTUDANTES_PÓS_TI***. Portanto, uma entidade ***e*** é membro da classe ***C*** sss ***e*** for uma instância do tipo ***T*** tal que $C\in$***classes***(***T***). É importante notar que, se um tipo de entidade para um conjunto de classes com extensão vazia fosse definido, o conjunto de instâncias de tal tipo seria sempre vazio. O Tipo ***T*** associado com o conjunto de classes $\{C_1, ..., C_n\}$ deve satisfazer às restrições de tipo das classes C_1, ..., C_n. As restrições de tipo de uma classe ***C*** consistem do conjunto de atributos de ***C*** e de todas as restrições sobre os atributos de ***C***.

Para explorar o poder dos modelos orientados a objetos de definir tipos a partir da especialização de supertipos, deve-se adotar um enfoque top-down para a implementação dos tipos de entidades. Na implementação de um tipo de entidade, define-se um conjunto de variáveis de instâncias as quais guardarão os valores dos atributos das instâncias do tipo e um conjunto de operações que operam nos valores desses atributos. As operações básicas de um atributo ***f*** são:

1) ***f*** (***e***): retorna o valor do atributo ***f*** de uma entidade ***e***, e

2) ***f*_MODIFIQUE**(***e***, ***v***): modifica para v o valor do atributo ***f*** da entidade ***e***.

As operações que modificam o valor de um atributo devem manter todas as restrições associadas com esse atributo. Atributos sinônimos devem compartilhar a mesma variável de instância, de forma que redundâncias sejam eliminadas. Por exemplo, na implementação do tipo ***ESTUDANTES_PÓS_TI*** os atributos ***orientador2*** (da classe ***ESTUDANTES_PÓS***) e ***orientador3*** (da classe ***ESTUDANTES_TI***) compartilham a mesma variável de instância.

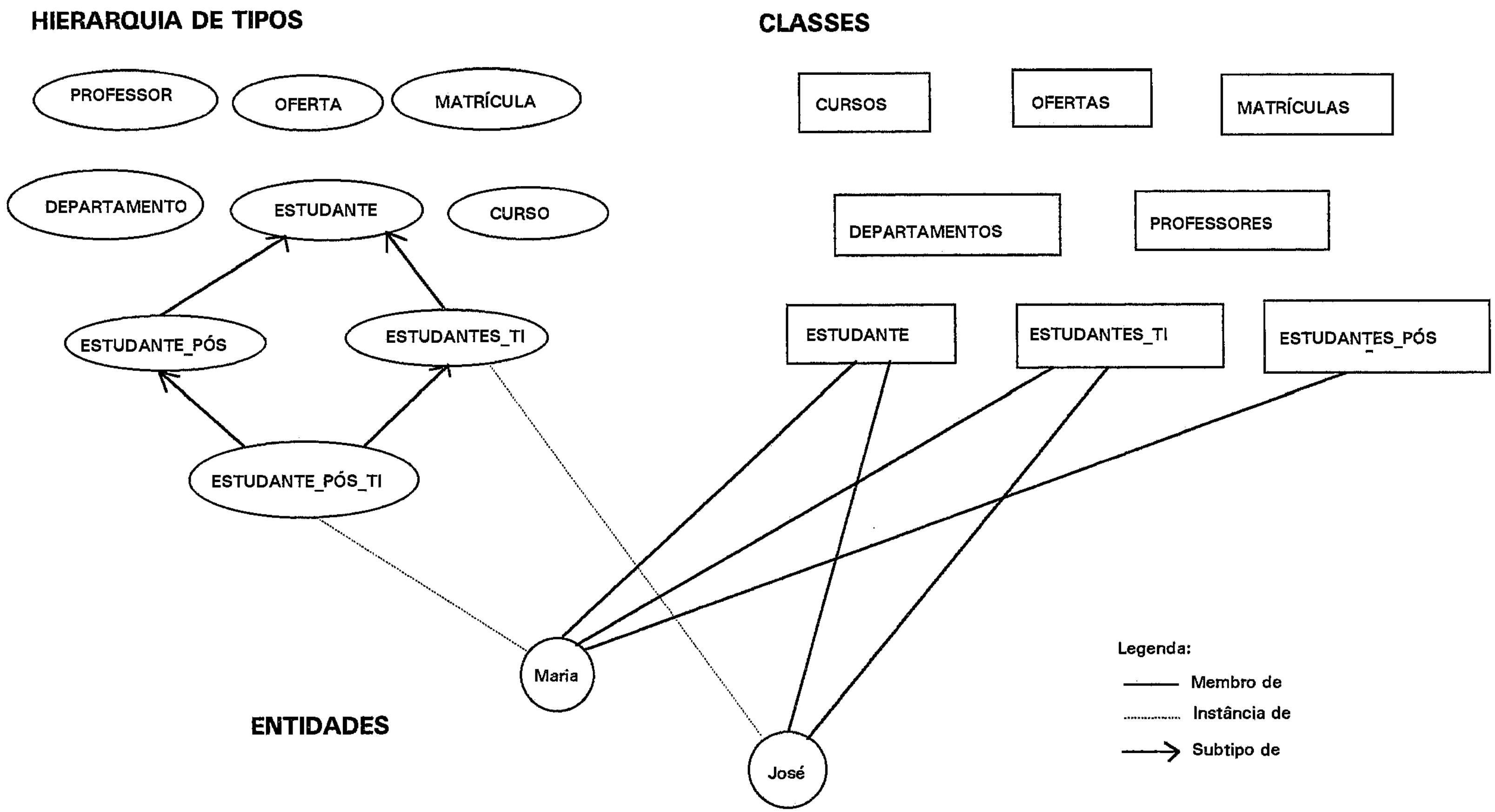


Figura 7.1: Hierarquia de tipos de entidades para o esquema conceitual da Figura 6.7

7.2.2 DEFINIÇÃO DO CONJUNTO DE COLEÇÕES

Em um esquema orientado a objetos para um dado esquema conceitual, deve-se definir um objeto coleção para cada classe do esquema conceitual. Uma coleção serve como um repositório para as entidades de sua classe associada. Os objetos coleção para o esquema conceitual da Figura 6.7 são: ***ESTUDANTES***, ***ESTUDANTES_POS***, ***ESTUDANTES_TI***, ***PROFESSORES***, ***OFERTAS***, ***MATRÍCULAS***, ***CURSOS***, e ***DEPARTAMENTOS***.

As operações básicas de um objeto coleção **C** são:

1) C_**ADICIONE**(***e***): adiciona ***e*** em ***C*** (***e*** é uma entidade ou uma especificação_ de_entidade).

2) C_**REMOVA**(***e***): remove ***e*** de ***C*** (***e*** é uma entidade).

3) C_**SELECIONE**(***e***): seleciona ***e*** em ***C*** (***e*** é uma especificação_de_entidade).

As operações de atualização C_**ADICINE** e C_**REMOVA** devem manter todas as restrições da classe conceitual associada a ***C***. Como o tipo de um entidade é definido pelo conjunto de classes às quais esta entidade pertence, o tipo de uma entidade pode mudar quando a entidade for adicionada ou removida de uma classe. Portanto, as operações C_**ADICINE** e C_**REMOVE** de uma coleção devem mudar o tipo da entidade apropriadamente.

7.3 DEFINIÇÃO ORIENTADA A OBJETOS DAS VISÕES

Após a definição de uma representação orientada a objetos para um esquema, todas as visões associadas com o esquema conceitual devem ser definidas no esquema orientado a objetos obtido. No enfoque desse trabalho, como discutido na Seção 3.3, cada visão V é definida conceitualmente pela tripla (S_V, σ_V, τ_V) onde S_V é o esquema da visão, σ_V é o mapeador de instâncias da visão e τ_V é o tradutor de atualizações da visão. Nessa seção, descrevemos como traduzir definições conceituais de visão em definições orientadas a objetos (vide Figura 7.3).

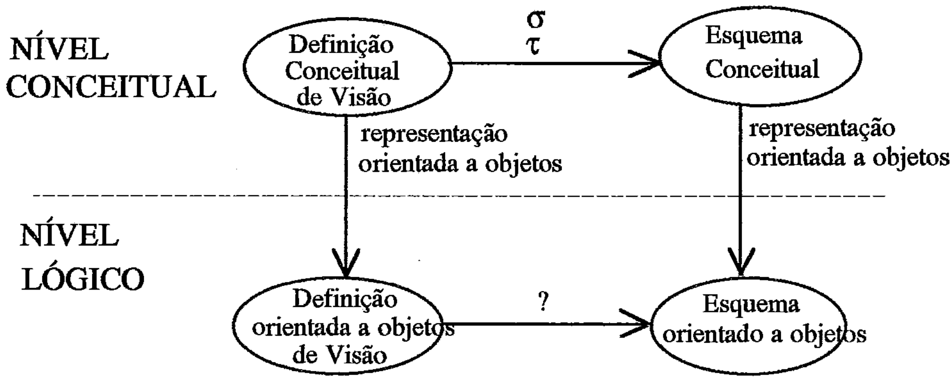


Figura 7.2 - Mapeamento do nível conceitual para o nível lógico

No restante dessa seção, suponha que o esquema conceitual ***EG*** seja representado por um esquema orientado a objetos ***EO***, e que (S_V, σ_V, τ_V) é a definição conceitual da visão V sobre ***EG***. A definição orientada a objetos da classe C de S_V sobre ***EO***, é uma tripla (Q_C, $\mathbf{P}_C$, $\mathbf{E}_C$), onde:

(i) Q_C é uma ***expressão de consulta*** que define o conjunto de entidades associadas a C (extensão de C). A consulta é definida a partir de uma ou mais coleções de ***EO***, denominadas coleções de base de C. As coleções de base para C são as que representam as classes de base de C (vide Seção 2.3). A expressão de consulta pode ser definida diretamente a partir do mapeamento de instância σ_V. Por exemplo, considere a classe *AUX_ENSINO* da visão *V4* (Seção 6.2) cuja extensão é definida conceitualmente por:

Para qualquer $\mathcal{B} \in$ ***instâncias***(***EG****),

- $\sigma_{V4}(\mathcal{B})(AUX\_ENSINO) = \{ e \mid e \in \mathcal{B}(ESTUDANTES\_PÓS)$ e $\mathcal{B}(curso^*)(e) \neq$ nulo$\}$.

A coleção ***ESTUDANTES_PÓS*** é a coleção de base para a classe ***AUX_ENSINO***, e a extensão de ***AUX_ENSINO*** é calculada pela seguinte expressão de consulta:

<u>Selecione de</u> ***ESTUDANTES_PÓS*** <u>onde</u> ***curso****≠nulo.

(ii) E_C é a ***lista de operações de entidades*** que podem ser aplicadas às entidades de C. Essa lista inclui todas as operações de atributo de C. As operações básicas para o atributo f de C são:

1) $f(e)$: retorna o valor do atributo f de uma entidade e, e

2) **f_MODIFIQUE**(e, v): modifica para v o valor do atributo f da entidade e.

Essas operações são traduzidas em operações sobre os atributos de base como especificado por σ_V and τ_V. Por exemplo, considere o atributo *NC4* (nome do curso) da classe *AUX_ENSINO* da visão *V4*, o qual é definido conceitualmente por (vide Seção 6.2):

- Para qualquer instância $\mathcal{B} \in$ *instâncias*(*EG**) e para qualquer $e \in \sigma_{V4}(\mathcal{B})(AUX\_ENSINO)$ temos:
  - $\sigma_{V4}(\mathcal{B})\ (NC4)(e) = \mathcal{B}(curso^* \circ NC9)(e)$
  - τ_{V4}("modifique e com ($NC4=v$) ") = "modifique c com ($NC9=v$) " , onde $c = \mathcal{B}(curso^*)(e)$

Assim sendo, as operações **NC4** e **NC4_MODIFIQUE** podem ser implementadas em pseudo-código como se segue:

> **NC4** (*e*) {<u>retorna</u> *NC9* (*curso** (e)) };
>
> **NC4_MODIFIQUE** (*e*, *v*) {NC9_MODIFIQUE(*curso**(*e*), *v*)};

(iii) P_C é a ***lista de operações de afiliação de conjunto*** que podem ser aplicadas para C. As operações básicas de afiliação de conjunto para C são:

1) **C_ADICIONE**(e): adiciona e a C (e é uma entidade ou especificação_de_entidade).

2) **C_REMOVA**(e): remove e de C (e é uma entidade).

3) **C_SELECIONE**(e): seleciona e em C (e é uma especificação_de_entidade).

Essas operações são traduzidas para operações sobre as coleções de base de C como especificado por τ_V. Por exemplo, considere a classe *AUX_ENSINO* da visão *V4*, onde a tradução da operação de atualização "Remova e de *AUX_ENSINO* " é definida conceitualmente por (vide Seção 6.2):

- τ_{V4}("remova e de *AUX_ENSINO*") = "modifique e com (*curso** =*nulo*)"

Assim sendo, a operação **AUX_ENSINO_REMOVA** pode ser implementada em pseudo código como se segue:

> **AUX_ENSINO_REMOVA** (*e*) { **curso*_MODIFIQUE**(*e*, ***nulo***) }

A definição orientada a objetos da visão V sobre ***EO*** consiste das definições orientadas a objetos de todas as classes em S_V. Por exemplo, suponha que $\{C_1, ..., C_n\}$ seja o conjunto de classes em S_V. Para se obter a definição orientada a objetos de V sobre ***EO***, deve-se determinar a definição orientada a objetos de C_i sobre ***EO***, para $1 \leq i \leq n$.

8. CONCLUSÕES

Nessa tese, descrevemos uma metodologia para integração de visões onde a semântica dos relacionamentos entre as visões é preservada durante o processo de integração de visões. Para isso (vide figura 8.1), primeiro, define-se o esquema combinado o qual contém todas as visões originais e um novo conjunto de restrições que expressam a maneira como os dados das diferentes visões estão relacionados. Em seguida aplica-se, ao esquema combinado, uma série de transformações que preservam a informação e a semântica das atualizações, de forma a obter um esquema global "otimizado" cujo relacionamento com as visões reflete, exatamente, os relacionamentos entre as visões da forma como estes foram definidos pelo esquema combinado. Após o témino do processo de integração, cada visão deve ser definida em termos do esquema conceitual obtido. No enfoque proposto nesse trabalho, além de alcançarmos a integração das visões no esquema global obtido, também obtemos os mapeadores de instâncias e os tradutores de atualizações das visões. No Capítulo 7, mostramos como projetar uma representação orientada a objetos para um esquema conceitual obtido com a metodologia proposta, e como traduzir as definições conceituais das visões em definições orientadas a objetos.

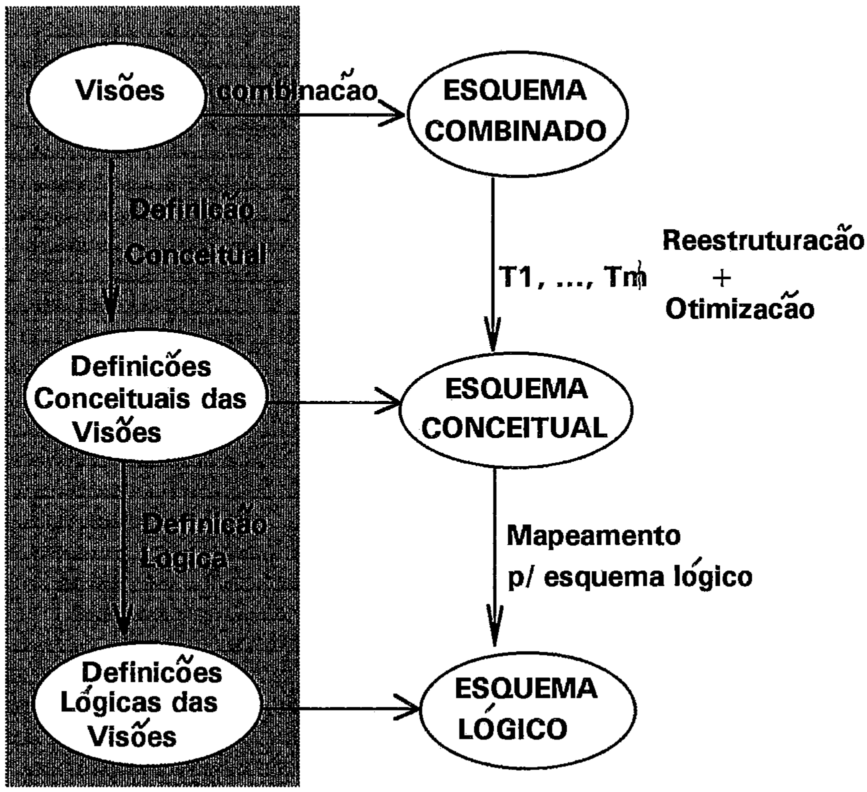


Figura 8.1: Passos da metodologia proposta

Propomos um conjunto de primitivas de transformações que permitem que a integração de esquemas seja realizada de forma segura e eficiente. Nas primitivas de transformação propostas, o relacionamento entre o esquema original e o esquema transformado é formalmente especificado pelo mapeamento de instâncias e pelo mapeamento de atualizações. Como o esquema combinado contém todas as visões originais, os mapeadores de instâncias e os tradutores de atualizações das visões são diretamente definidos pelos mapeamentos de instâncias e pelos mapeamentos de atualizações entre os vários esquemas intermediários gerados durante o processo de integração. Definimos, formalmente, as condições para a validação das traduções de atualizações em visões na presença de interferência entre visões e mostramos que, como as tranformações de esquemas na nossa metodologia preservam a informação e a semântica das atualizações, podemos provar formalmente a validade das traduções das atualizações nas visões definidas a partir dos mapeamentos de atualizações.

Discutimos a necessidade de uma metodologia para integração de visões que apoie tanto o enfoque "top down" quanto o enfoque "bottom up" durante o projeto do esquema conceitual, e mostramos como a metodologia proposta resolve as limitações existentes em outras metodologias no que se refere ao apoio do enfoque bottom up.

Discutimos, também, a falta de rigor de outras metodologias de integração de visões, proposta na literatura, para tratar o problema de reestruturação de esquemas, e mostramos como formas gerais de dependências existenciais e as semânticas de atualizações a ela associadas podem ser usadas para dar uma fundamentação teórica sólida à reestruturação de esquemas. Propomos uma *forma normal decomposta* para esquemas a qual especifica as propriedades que um esquema deve ter para que o problema de "incompatibilidade de fusão" capturado por determinadas restrições de dependências existenciais não ocorra, e apresentamos um algorítimo para transformar determinados tipos de esquemas em esquemas equivalentes que estão em forma normal decomposta.

Discutimos o problema de outros enfoques, já disponíveis na literatura, para formalmente provar a correteza das traduções de atualizações em visões na presença de interferência com visões, e mostramos que esse problema pode ser resolvido se a especificação das traduções das atualizações de visões for tratada durante o processo de integração das visões. O esquema global obtido com a metodologia proposta nessa tese tem três propriedades que tornam o processo de tradução de atualizações em visões seguro e adequado:

(i) ***A semântica necessária para eliminar ambiguidades das traduções de atualizações em visões é obtida durante o processo de integração.*** No nosso enfoque, a semântica necessária para eliminar ambiguidades é obtida durante o passo de combinação das visões e é explicitamente especificada pela semântica de atualização das restrições do esquema combinado.

(ii) ***A especificação formal dos relacionamentos entre as visões e o esquema global é gerado durante o processo de integração de visões.*** No nosso enfoque, o mapeador de instâncias e o tradutor de atualizações das visões são, diretamente, definidos a partir dos mapeamentos de instâncias e de atualizações entre os vários esquemas intermediários gerados durante o proceso de integração.

(iii) ***Os relacionamentos entre as visões são capturados pelas restrições do esquema global.*** No nosso enfoque, usando transformações que preservam a informação e a semântica das atualizações, podemos garantir que os relacionamentos entre as visões e o esquema global refletem, exatamente, os relacionamentos entre as visões da forma como foram definidos pelo esquema combinado. Isto significa que as restrições do esquema combinado, juntamente com suas respectivas semânticas de atualizações, são preservadas no esquema global obtido.

Outro benefício importante do uso de primitivas de transformação formalmente justificadas, como as propostas nessa tese, é que permite aumentar a "inteligência" das ferramentas de apoio à integração das visões. Acreditamos, assim como [Bouzeghoub85, Dogac89, Gottard92, Hayne90, Sheth88, Souza86] entre outros, que ferramentas interativas de projeto são necessárias para apoiar o projeto do esquema conceitual. O suporte para transformação de esquemas é um aspecto muito importante para tais ferramentas e oferece grande potencial para automação. Uma ferramenta, com este objetivo, deve conter um catálogo das transformações de interesse onde, para cada transformação, várias informações devem ser mantidas. No caso das transformações abordadas nesse trabalho, essas informações devem incluir: pré-condições, pós-condições, mapeamentos de instâncias e mapeamentos de atualizações. Durante o projeto do esquema conceitual, o usuário poderá escolher aplicar uma dada transformação no esquema, ou o sistema poderá automaticamente escolher as transformações que devem ser aplicadas no esquema. O uso de transformações formalmente justificadas, tem os seguintes benefícios para as ferramentas de apoio à integração de visões:

(i) Todas as pré-condições de uma primitiva de transformação podem ser automaticamente verificadas todas as vezes que a primitiva de transformação for aplicada. Se a ferramenta não tiver informação suficiente para verificar algumas das pré-condições, o usuário poderá ser induzido de forma inteligente a fornecer as informações adicionais necessárias.

(ii) Quando mais de uma transformação puder ser aplicada e não existir informação suficiente para determinar qual é a mais apropriada, a ferramenta poderá perguntar ao usuário por informações adicionais que resolvam a ambigüidade.

(iii) Os mapeadores de instâncias e os tradutores de atualizações das visões são gerados automaticamente durante o processo de integração, como discutido na Seção 3.3.

(iv) A sequência de passos de transformação ocorridos durante a integração pode ser automaticamente gravada e servir como história da integração. Essa informação é necessária durante o processo de integração (mantém uma história de todos os fatos assumidos até o momento) e, posteriormente, poderá ser utilizada como uma fonte de informação sobre o esquema integrado e para dar suporte à evolução do esquema.

Trabalhos futuros devem ser direcionados para a definição de novos tipos de restrições entre visões e/ou novas opções de semânticas de atualizações, e consequentemente para a definição de novas primitivas de transformação requeridas para a integração de visões onde essas restrições estão presentes.

REFERÊNCIAS BIBLIOGRÁFICAS

[Almarode91] Almarode J., e Anderson, T.L., "Gemstone Visual Schema Designer: A Tool for Object-Oriented Database Design," *Object-Oriented Databases: Analysis, Design & Constructions (DS-4)*, North-Holland, 1991.

[Bancilhon81] bancilhon F., Spyratos N., "Update Semantics and Relational View," *ACM Transactions on Database Systems*, vol. 6, No.4, dez. 1981.

[Banejee87] Banejee, j. et al., "Data Model Issues for Object-Oriented Applications," *ACM Transactions on Database Systems*, jan 1987.

[Batini86] C. Batini, M. Lenzerini, e S. B. Navathe, "A comparative analysis of methodologies for database schema integration," *ACM Comput. Surveys*, Vol. 18, pp. 323-364, 1986.

[Batini84] Batini C., Lenzerini M., "A Methodology for Data Schema Integration in the Entity Relationship Model," *IEEE Trans. on Software Eng.*, SE-10(6):650-664, nov. 1984.

[Bouzeghoub85] Bouzeghoub, M., et al., "SECSI: An expert System for Database Design," *Proceedings of 11th International Conference on Very Large Databases*, Stockolm, 1985.

[Casanova93] Casanova, M. A. et al., "On the Design and Maintenance of Optimized Relational Representaions of Entity-Relationship Schemas," *Data & Knowledge Engineering*, pp. 1-20, 1993.

[Casanova83] Casanova, M. A. e Vidal, V. M. P., "Towards a Sound View Integration Methodology", *Second ACM SIGACT-SIGMOD Syposium on Principles of Database Systems*, Atlanta -Georgia, março 1983.

[Carlson79] Carlson C. R., Arora A. K., "The Updatability of Relational Views based on Functional Dependencies," *Third Int'l Computer Software and Applications Conf.*, (IEEE Computer Society), Chicago Ill., nov.1979.

[Chen76] Chen, P.P.S, "The Entity-Relationship Model: Towards a Unified View of Data," *ACM Transactions on Database Systems*, vol.1, no.1, março 1976.

[Chung87] Chung, K.L., e al.,et, "Process Management and Assertion Enforcement for a Semantic Data Model," *Proceedings of the Sixth International Conference on Entity-Relationship Approach*, pp. 469-485, Nov. 1987.

[Codd 72] E. F. Codd, "Further Normalization of Data Base Relational Model", Data Base Systems, edited by Randell Rustin, Prentice Hall,1972.

[Cosmadakis84] Cosmadakis S. S. Papadimitriou C. H., "Updates of Relational Views," *J. ACM*, Vol.31, No.4, out. 1984.

[Davidson81] Davidson J. J., Kaplan S. J., "Natural Language Acess to Databases: Interpretation of Update Requests," *Proc. Seventh Int'l Joint Conf. Artificial Intelligence*, Vancouver, B. C., Canada, agosto 1981.

[Dayal82] Dayal U., Bernestein A., "On the correct Translation of Update Operations on Relational Views," *ACM Transactions on Database Systems*, vol. 7, No.3, set. 1982.

[Dogac89] Dogac, A., et al., "A Generalized Expert System for Database Design," *IEEE Trans. on Software Engineering*, Vol. 15, No. 4, abril 1989.

[Eick91] Eick, C. F., "A Methodology for the Design and Transformation of Conceptual Schemas," *Proceedings of 17th International Conference on Very Large Databases*, Barcelona, setembro 1991.

[Elmasri84] R. Elmasri, e S. B. Navathe, "Object integration in database design," *Proceedings of the 8th International Conference on Very Large Data Bases*, set. 1984.

[Elmasri84] R. Elmasri, e G. Wiederhold, "Gordas: A Formal, high-level query language for the entity-relationship model," *Proceedings of the 2nd International Conference on Entity-Relationship Approach*, Washinton, D.C., out. 1981.

[Fishman87] Fishman, D. H et al., "Iris: An Object-Oriented Database Management System," *ACM Transactions on Office Information Systems*, vol. 5, no.1, pp. 48-69, jan.1987.

[Furtado86] Furtado A. L., Casanova M. A., "Updating Relational Views," *in query Processing in Database Systems*, W. Kim, D. S. Batory, eds., Springer Verlag, New York, 1985.

[Gottard92] Gotthard, W., Lockerman, P.C., e Neufeld, A., "System-Guided View Integration for Object-Oriented Databases," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 4, no. 1, pp. 1- 22, fev. 1992.

[Hammer81] Hammer, M. e McLeod, D., "Database Description with SDM: A Semantic Database Model," *ACM TODS*, vol. 6, no.3, pp. 351-387, set. 1981.

[Hayne90] S. Hayne, S. Ram, "Multi-User View Integration System (MUVIS): An Expert Sytem for View Integration", *IEEE Conf. on Data & Knowledge Eng.*, pp. 402-409, fev. 1990.

[Hegner84] Hegner S. J., "Canonical View Update Support Througn Boolean Algebras of Components," *ACM SIGACT-SIGMOD Syposium on Principles of Database Systems*, abril 1984.

[Heiler88] Heiler, S. e Zdonik, S., "Views Data Abstraction, and Inheritance in the FUGUE Data Model," *Springer Lecture Notes in Computer Science*, no. 334, pp. 225-241, 1988.

[Keller86] Keller, A.M., "The Role of Semantics in Translating View Updates," *IEEE Computer*, vol. 19, no. 1, pp. 63-73, jan. 1986.

[Larson89] Larson, J. A., Navathe, S. B., Elmashi, R., "A Theory of Attribute Equivalence in Databases with Application to Schema Integration," *IEEE Transactions on Software Engineering*, vol. 15, no. 4, abril 1989.

[Leung90] Leung, M. H. Wong, "An Expert-System Shell Using Structured Knowledge - An Object-Oriented Approach," *IEEE Computer*, 1990.

[Maier85] Maier, D., Otis, A., e Purdy, A., "Object-Oriented Database Development at Servio Logic," *Database Engineering*, vol.18, no. 4, dez. 1985.

[Mannino84] M. V. Mannino, e W. Effelsberg, "Matching Techniques in Global Schema Design", *Proceedings Conference on Data Engineering*, pp. 418-425, 1984

[Mannino84] M. V. Mannino, e W. Effelsberg, "Matching Techniques in Global Schema Design", *Proceedings Conference on Data Engineering*, pp. 418-425, 1984.

[Markowitz91] Markowitz, V. M., "Safe Referential Integrity Structures in Relational Databases," *Proceedings of 17th International Conference on Very Large Databases*, 1991.

[Markowitz90] Markowitz, V. M., "Referential Integrity Revisited: An Object-Oriented Perspective," *Proceedings of 16th International Conference on Very Large Databases*, Australia, 1990.

[Markowitz89] Markowitz, V. M., e Shoshani, A., "On the Correctness of representing extended entity-relationship structures in the relational model," *Proceedings of 1989 SIGMOD Conference*, pp. 430-439, 1989.

[Medeiros85] Medeiros, C.M.B. e Tompa, F. Wm., "Understanding the implications of view update policies," *Proceedings of 11th International Conference on Very Large Databases*, Stockholm, 1985.

[Miller93] Miller, R. J., Ioannidis, Y. E., Ramakrishman, R., "The Use of Information Capacity in Schema Integration and Translation," *Proceedings of 19th International Conference on Very Large Databases*, Ireland, 1993.

[Mylopoulos80] Mylopoulos, J., Bernstein, P.A., e Wong, H.K.T., "A Language Facility for Designing Database-Intensive Applications," *ACM Transactions on Database Systems*, vol.5, no.2, pp.185-207, junho 1980

[Navathe84] S. B. Navathe, T. Sashidhar, e R. Elmasri, "Relationship Matching in Schema Integration," *Proceedings of the 10th International Conference on Very Large Data Bases*, pp. , 1984.

[Navathe86] Navathe, S. B., ElMasri, R., e Larson, J., "Integrating user views in database design," *IEEE Computer*, vol. 19, no. 1, pp. 50-62, jan. 1986.

[Rundensteiner92] Rundensteiner, E. A., "A Methodology for Supporting Multiple Views in Object-Oriented Databases," *Proceedings of 19th International Conference on Very Large Databases*, 1992.

[Shipman81] Shipman, D.W., "The Functional Data Model and the Data Language DAPLEX," *ACM TODS*, vol. 6, no. 1, pp. 140-173, 1981.

[Sheth88] A. P. Sheth, e James A. Larson, "A tool for integrating conceptual schemas and user views," *IEEE Conf. on Data & Knowledge Engineering*, pp. 176-183, Los Angeles, 1988.

[Smith79] J. M. Smith, e D. C. P. Smith, "Database Abstractions: Aggregation and Generalization," *ACM Trans. on Database Systems*, Vol. 2, No. 1, pp. 105-133, 1979.

[Sousa86] Sousa, J. de, "SIS - A Schema Integration System," *Proceedings BNCOD5 conference*, 1986.

[Tucherman83] Tucherman, L. et al., "A Pragmatic Approach to Structured Database Design," *Proceedings of 9th International Conference on Very Large Databases*, out 1983.

[Ullman79] Ullman, J.D., in *Principles of Database Systems*, Computer Science Press, 1979.

[Urban89] Urban, S. D., e Delcambre, L.M.L., "Constraint Analysis for Specifying Perpectives of Class Objects," *Proceedings of the International Conference on Data Engineering*, pp. 10-17, Los Angeles, fev. 1989.

[Veloso83] Veloso, P.A.S. e Furtado, A. L., "View Constructs for the Specification and Design of External Schemas," In *Entity-Relationship Approach to Software Engineering*, ed. R.T. Yeh, pp. 627-649, Noth-Holland, 1983.

[Vidal90] Vidal, V. M. P. e Winslett M., "Specifying Updatable Views in Object-Oriented Models", *Quinto Simpósio Brasileiro de Banco de Dados*, Rio de Janeiro, abril 1990.

[Vidal89] Vidal, V. M. P., "An Object-Oriented Knowledge Base for Database Applications", *Annual Review of Computing*, University of Illinois at Urbana-Champaign, set. 1989.

[Vidal84] Vidal, V. M. P., "Projeto de Banco de Dados através de Integração de Visões", *Revista Brasileira de Computação,* Rio de Janeiro, V. 4, nº 1, 1984/1985, pp. 3- 21.

APÊNDICE A

NOTAÇÃO
SI(A, ψ) : A semântica para inserções na classe A com relação à restrição ψ
SD(A, ψ) : A semântica para deleções na classe A com relação à restrição ψ
SM(f, ψ): A semântica para modificações do valor do relacionamento f com relação à restrição ψ

Nessa seção, seja S=($\mathcal{C}$, $\mathcal{A}$, $\mathcal{R}$), um esquema; seja $\boldsymbol{u}$, uma atualização em $\boldsymbol{S}$ e seja ψ, uma restrição em $\mathcal{R}$. Suponha que $\mathcal{B}$ seja a instância de S quando o efeito-colateral de $\boldsymbol{u}$ com relação a ψ for executado.

A.1 SEMÂNTICA DE ATUALIZAÇÃO PARA MANUTENÇÃO DAS RESTRIÇÕES ESTRUTURAIS

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL $\psi = A \subset B$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione $\boldsymbol{a}$ a A ". As alternativas de semânticas de atualização para $\boldsymbol{u}$ com relação a ψ são: a) *SI*(A, ψ) = ***rejeita***, determina que: (i) se $a \in \mathcal{B}(B)$, então *EC*="continue"; (ii) senão, *EC*="rollback". b) *SI*(A, ψ) = ***propaga***, determina que: (i) Se $a \in \mathcal{B}(B)$, então *EC*="continue"; (ii) senão, *EC*={"adicione $\boldsymbol{a}$ a $\boldsymbol{B}$"}.
Remoção em ***B***	Suponha que u="remova $\boldsymbol{b}$ de $\boldsymbol{B}$ ". As alternativas de semânticas de atualização para $\boldsymbol{u}$ com relação a ψ são: a) ***SD***($\boldsymbol{B}$, ψ) = ***rejeita***, determina que: (i) se $b \notin \mathcal{B}(A)$, então *EC*="continue"; (ii) senão, *EC*="rollback". b) ***SD***($\boldsymbol{B}$, ψ) = ***propaga***, determina que: (i) se $b \notin \mathcal{B}(A)$, então *EC*="continue"; (ii) senão, *EC*= {"remova $\boldsymbol{b}$ de A"}.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL $\psi=(A=B)$

Operação de atualização	Alternativas de semânticas de atualização para manter ψ
Adição em A	Suponha que u="adicione a a A". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(A, \psi)$ = ***propaga***, determina que: EC={"adicione a a B"}.
Adição em B	Suponha que u="adicione b a B". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(B, \psi)$ = ***propaga***, determina que: EC={"adicione b a A"}.
Remoção em A	Suponha que u="remova a de A". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(B, \psi)$ = ***propaga***, determina que: EC= {"remova a de B"}.
Remoção em B	Suponha que u="remova b de B". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(B, \psi)$ = ***propaga***, determina que: EC= {"remova b de A"}.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL $\psi=(A \mid B)$

Operação de atualização	Alternativas de semânticas de atualização para manter ψ
Adição em A	Suponha que u="adicione a a A". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $a \notin \mathscr{D}(B)$, então EC="continue"; (ii) senão, EC="rollback". b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se $a \notin \mathscr{D}(B)$, então EC="continue"; (ii) senão, EC={"remova a de B"}.
Adição em B	Suponha que u="adicione b a B". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(B, \psi)$ = ***rejeita***, determina que: (i) se $b \notin \mathscr{D}(A)$, então EC="continue"; (ii) senão, EC="rollback". b) $SI(B, \psi)$ = ***propaga***, determina que: (i) se $b \notin \mathscr{D}(A)$, então EC="continue"; (ii) senão, EC={"remova b de A"}.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL

$\psi=(A=B[\neq nulo])$

Nós assumimos que existe $f' \in$ ***attributes***(A) tal que $f' \equiv f$ e f' não possa assumir o valor ***nulo***.

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione ***a*** a A". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $a \in \mathcal{B}(B)$, então EC= {"continue"}; (ii) senão, EC="rollback". b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se $a \in \mathcal{B}(B)$, então EC= {"continue"}; (ii) senão, EC= {"adicione ***a*** a ***B***"}.
Adição em B	Suponha que u="adicione ***b*** a ***B*** ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(B, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(f)(a)$=***nulo***, então EC="continue"; (ii) senão, EC={"adicione ***b*** a A"}.
Remoção em B	Suponha que u="remova ***b*** de ***B***". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SD(B, \psi)$ = ***rejeita***, determina que: (i) se $b \notin \mathcal{B}(A)$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(B, \psi)$ = ***propaga***, determina que: (i) se $b \notin \mathcal{B}(A)$, então EC="continue"; (ii) senão, EC= {"remova ***b*** de A"}.
Remoção em A	Suponha que u="remova ***a*** de A". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(A, \psi)$ = ***nulifica***, determina que: EC= {"modifique ***a*** com (f=***nulo***)"}.
Modificação de f	Suponha que u="modifique ***b*** com (f=ν)". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SM(f, \psi)$ = ***propaga***, determina que: (i) se $b \notin \mathcal{B}(A)$ e $\mathcal{B}(f)(b) \neq$***nulo***, então EC= {"adicione ***b*** a A"}; (ii) se $b \in \mathcal{B}(A)$ e $\mathcal{B}(f)(b)$=***nulo***, então EC= {"remova ***b*** de A"}. (iii) senão, EC="continue".

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL
$\psi=(A=B[f=yes])$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione a a A". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $a \in \mathcal{B}(B)$, então EC= {"modifique a com (f=yes)"}; (ii) senão, EC="rollback". b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se $a \in \mathcal{B}(B)$, então EC= {"modifique a com (f=yes)"}; (ii) senão, EC= {"adicione a a B", "modifique a com (f=yes)"}.
Adição em B	Suponha que u="adicione b a B ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(B, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(f)(a)=yes$, então EC={"adicione b a A"}; (ii) senão, EC="continue".
Remoção em B	Suponha que u="remova b de B ". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SD(B, \psi)$ = ***rejeita***, determina que: (i) se $b \notin \mathcal{B}(A)$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(B, \psi)$ = ***propaga***, determina que: (i) se $b \notin \mathcal{B}(A)$, então EC="continue"; (ii) senão, EC= {"remova b de A"}.
Remoção em A	Suponha que u="remova a de A". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(A, \psi)$ = ***nulofies***, determina que: EC= {"modifique a com (f=*nulo*)"}.
Modificação de f	Suponha que u="modifique b com (f=v)". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SM(f, \psi)$ = ***propaga***, determina que: (i) se $b \notin \mathcal{B}(A)$ e $\mathcal{B}(f)(b)=yes$, então EC= {"adicione b a A"}; (ii) se $b \in \mathcal{B}(A)$ e $\mathcal{B}(f)(b) \neq yes$, então EC= {"remova b de A"}; (iii) senão, EC="continue".

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL $\psi=(A=B\text{-}C)$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione a a A". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(A, \psi) = propaga$, determina que: (i) se $a \notin \mathcal{B}(C)$, então EC={"adicione a a B"}; (ii) se $a \in \mathcal{B}(C)$ e $a \in \mathcal{B}(B)$, então EC={"remova a de C"}; (iii) se $a \in \mathcal{B}(C)$ e $a \notin \mathcal{B}(B)$, então EC={"remova a de B", "adicione a a B"}.
Adição em B	Suponha que u="adicione b a B ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(B, \psi) = propaga$, determina que: (i) se $b \notin \mathcal{B}(C)$, então EC={"adicione b a A"}; (ii) senão, EC={"continue"}.
Adição em C	Suponha que u="adicione c a C ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(C, \psi) = propaga$, determina que: (i) se $c \notin \mathcal{B}(A)$, então EC={"continue"}; (ii) senão, EC={"remova c de A"}.
Remoção em A	Suponha que u="remova a de A ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(B, \psi) = propaga$, determina que: EC= {"remova a de C"}.
Remoção em B	Suponha que u="remova b de B ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(B, \psi) = propaga$, determina que: (i) se $b \notin \mathcal{B}(A)$, então EC={"continue"}; (ii) senão, EC={"remova b de A"}.
Remoção em C	Suponha que u="remova c de C ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(C, \psi) = propaga$, determina que: (i) se $c \notin \mathcal{B}(A)$, então EC={"continue"}; (ii) senão, EC={"adicione c a A"}.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL $\psi=(C=C_1\cup\ldots\cup C_n)$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em C	Suponha que u="adicione c a C ". A única alternativa de semântica de atualização para u com relação a ψ é: $SI(A, \psi)$ = propaga para C_i, para algum $1\leq i \leq n$, determina que: EC={"adicione c a C_i"}.
Adição em C_i	Suponha que u="adicione c a C_i", $1\leq i \leq n$. A única alternativa de semântica de atualização para u com relação a ψ é: $SI(B, \psi)$ = ***propaga***, determina que: (i) se $c \in \mathcal{B}(C)$, então EC={"continue"}; (ii) senão, EC={"adicione c a C"}.
Remoção em C	Suponha que u="remova c de C ". Seja $F=\{C_{i1}, \ldots, C_{im}\}$ onde $1\leq i_k \leq n$ e onde $C_{ik}\in F$ sss $c \in \mathcal{B}(C_{ik})$, para $1\leq k\leq m$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(C, \psi)$ = ***propaga***, determina que: EC= {"remova c de C_{i1}", ..., "remova c de C_{im}"}.
Remoção em C_i	Suponha que u="remova c de C_i ". Seja $F=\{C_{i1}, \ldots, C_{im}\}$ onde $1\leq i_k\leq n$ e onde $C_{ik}\in F$ sss $c \in \mathcal{B}(C_{ik})$, para $1\leq k \leq m$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(C, \psi)$ = ***propaga***, determina que: (i) se $F= \emptyset$, então EC= {"remova c de C"}; (ii) senão, EC={"continue"}.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO ESTRUTURAL $\psi=(C=C_1\cap ... \cap C_n)$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em C	Suponha que u="adicione c a C". Seja $F=\{C_{i1}, ..., C_{im}\}$ onde $1 \leq i_k \leq n$ e onde $C_{ik} \in F$ sss $c \notin \mathcal{B}(C_{ik})$, para $1 \leq k \leq m$. A única alternativa de semântica de atualização para u com relação a ψ é: $SI(A, \psi) =$ ***propaga***, determina que: $EC=$ {"adicione c a C_{i1}", ..., "adicione c a C_{im}"}.
Adição em C_i	Suponha que u="adicione c a C_i", $1 \leq i \leq n$. A única alternativa de semântica de atualização para u com relação a ψ é: $SI(B, \psi) =$ ***propaga***, determina que: (i) se $c \in \mathcal{B}(C_k)$, para $1 \leq k \leq m$, então $EC=$ {"adicione c a C"}; (ii) senão, $EC=$ {"continue"}.
Remoção em C	Suponha que u="remova c de C ". A única alternativa de semântica de atualização para u com relação a ψ é: $SI(A, \psi) = C_i$, $1 \leq i \leq n$, determina que: $EC=$ {"remova c de C_i"}.
Remoção em C_i	Suponha que u="remova c de C_i ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SD(C, \psi) =$ ***propaga***, determina que: (i) se $c \in \mathcal{B}(C)$, então $EC=$ {"remova c de C"}; (ii) senão, $EC=$ {"continue"}.

A.2 SEMÂNTICA DE ATUALIZAÇÃO PARA MANUTENÇÃO DAS RESTRIÇÕES DE DEPENDÊNCIAS EXISTENCIAIS

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO EXISTENCIAL $\psi = A[g_1,...,g_n] \subset B[f_1,...,f_n]$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione a a A ". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se não existir $b \in \mathcal{B}(B)$ tal que $\mathcal{B}(f_k)(b) = \mathcal{B}(g_k)(a)$, para $1 \leq k \leq n$, então EC="rollback"; (ii) senão, EC="continue". b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se não existir $b \in \mathcal{B}(B)$ tal que $\mathcal{B}(f_k)(b) = \mathcal{B}(g_k)(a)$, para $1 \leq k \leq n$, então EC={"adicione $b(f_1 = \mathcal{B}(g_1)(a), ..., f_n = \mathcal{B}(g_n)(a))$ a B "}; (ii) senão, EC="continue".
Remoção em B	Suponha que u="remova b de B ". Seja $\mathcal{B}(A^*) = \{a \mid a \in \mathcal{B}(A)$ e $\mathcal{B}(g_k)(a) = \mathcal{B}(f_k)(b)$, para $1 \leq k \leq n\}$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SD(B, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(A^*) = \emptyset$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(B, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*) = \emptyset$, então EC="continue"; (ii) se $\mathcal{B}(A^*) = \{a_1, ..., a_n\} \neq \emptyset$, então $EC = \{u_k \mid u_k$="remova a_k de A", para $1 \leq k \leq n\}$.
Modificação de f_k	Suponha que u="modifique b com ($f_k = v$)", onde $1 \leq k \leq n$. Seja $\mathcal{B}(A^*) = \{a \mid a \in \mathcal{B}(A)$ e $\mathcal{B}(g_i)(a) = \mathcal{B}(f_i)(b)$, para $1 \leq i \leq n\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*) = \emptyset$, então EC="continue"; (ii) se $\mathcal{B}(A^*) = \{a_1, ..., a_m\} \neq \emptyset$, então $EC = \{u_i \mid u_i$="modifique a_i com ($g_k = \mathcal{B}(f_k)(b)$) ", para $1 \leq i \leq m\}$.
Modificação de g_k	Suponha que u="modifique a com ($g_k = v$)", onde $1 \leq k \leq n$. Seja $\mathcal{B}(B^*) = \{b \mid b \in \mathcal{B}(B)$ e $\mathcal{B}(g_i)(a) = \mathcal{B}(f_i)(b)$, para $1 \leq i \leq n\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(g_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(B^*) = \emptyset$, então EC="continue"; (ii) se $\mathcal{B}(B^*) = \{b_1, ..., b_m\} \neq \emptyset$, então $EC = \{u_i \mid u_i$="modifique b_i com ($f_k = \mathcal{B}(g_k)(a)$)", para $1 \leq i \leq m\}$.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A A RESTRIÇÃO EXISTENCIAL ψ = ($A[g_1,\ldots,g_n]=B[f_1,\ldots,f_n]$)

Operação de atualização	Alternativas de semânticas de atualização para manter ψ
Adição em A	Suponha que u="adicione a a A ". A única altenativa de semântica de atualização para u com relação a ψ é: a) $SI(A, \psi)$ = ***propaga***, determina que: EC={"adicione $b(f_1=\mathcal{B}(g_1)(a), \ldots, f_n=\mathcal{B}(g_n)(a))$ a B "}.
Remoção em A	Suponha que u="remova a de A ". Seja $\mathcal{B}(B^*)=\{b \mid b\in\mathcal{B}(B)$ e $\mathcal{B}(f_k)(b)=\mathcal{B}(g_k)(a)$, para $1\leq k\leq n\}$. A única altenativa de semântica de atualização para u com relação a ψ é: b) $SD(B, \psi)$ = ***propaga***, determina que: se $\mathcal{B}(B^*)=\{b_1, \ldots, b_n\}$, então $EC=\{u_k \mid u_k$="remova b_k de B", para $1\leq k\leq n\}$}.
Modificação de g_k	Suponha que u="modifique a com $(g_k=v)$", onde $1\leq k\leq n$. Seja $\mathcal{B}(B^*)=\{b \mid b\in\mathcal{B}(B)$ e $\mathcal{B}(g_i)(a)=\mathcal{B}(f_i)(b)$, para $1\leq i\leq n\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(g_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(B^*)=\emptyset$, então EC="continue"; (ii) se $\mathcal{B}(B^*)=\{b_1, \ldots, b_m\}\neq\emptyset$, então $EC=\{u_i \mid u_i$="modifique b_i com $(f_k=\mathcal{B}(g_k)(a))$", para $1\leq i\leq m\}$.

<u>Obs</u>: As alternativas de semântica de atualização para *adição em **B***, *remoção em **B*** e *modificação de f_k* são similires.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO EXISTENCIAL
$\psi = (A[g_1,...,g_n] \mid B[f_1,...,f_n])$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione a a A ". Seja $\mathcal{B}(B^*)=\{b \mid b \in \mathcal{B}(B)$ e $\mathcal{B}(g_k)(a)=\mathcal{B}(f_k)(b)$, para $1 \leq k \leq n\}$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(B^*)=\emptyset$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(B, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(B^*)=\emptyset$, então EC="continue"; (ii) se $\mathcal{B}(B^*)=\{b_1, ..., b_n\} \neq \emptyset$, então $EC=\{u_k \mid u_k$="remova b_k de B", para $1 \leq k \leq n\}\}$.
Modificação de g_k	Suponha que u="modifique a com $(g_k=v)$", onde $1 \leq k \leq n$. Seja $\mathcal{B}(B^*)=\{b \mid b \in \mathcal{B}(B)$ e $\mathcal{B}(g_k)(a)=\mathcal{B}(f_k)(b)$, para $1 \leq k \leq n\}$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SM(g_k, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(B^*)=\emptyset$, então EC="continue"; (ii) senão, EC="rollback". b) $SM(g_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(B^*)=\emptyset$, então EC="continue"; (ii) se $\mathcal{B}(B^*)=\{b_1, ..., b_n\} \neq \emptyset$, então $EC=\{u_k \mid u_k$="remova b_k de B", para $1 \leq k \leq n\}\}$.

Obs: As alternativas de semântica de atualização para *adição em* ***B*** e *modificação de* f_k são similires.

A.3 SEMÂNTICA DE ATUALIZAÇÃO PARA MANUTENÇÃO DAS RESTRIÇÕES DE ATRIBUTOS

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO REFERENCIAL $\psi = f{:}A \rightarrow B$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Remoção em B	Suponha que u="remova b de B ". Let $\mathcal{B}(A^*)=\{a \mid a \in \mathcal{B}(A)$ e $\mathcal{B}(f)(a)=b\}$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SD(B, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(A^*)=\varnothing$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(B, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*)=\varnothing$, então EC="continue"; (ii) se $\mathcal{B}(A^*)=\{a_1, ..., a_n\} \neq \varnothing$ então $EC=\{u_k \mid u_k$="remova a_k de A", para $1 \leq k \leq n\}$. c) $SD(B, \psi)$ = ***nulifica***, determina que: (i) se $\mathcal{B}(A^*)=\varnothing$, então EC="continue"; (ii) se $\mathcal{B}(A^*)=\{a_1, ..., a_n\}$, então $EC=\{u_k \mid u_k$="modifique a_k com (f=***nulo***) ", para $1 \leq k \leq n\}$.
Adição em A	Suponha que u="adicione a a A". Seja $b = \mathcal{B}(f)(a)$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se b=**nulo** ou $b \in \mathcal{B}(B)$, então EC="continue"; (ii) senão, EC="rollback". b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se b=**nulo** ou $b \in \mathcal{B}(B)$, então EC="continue"; (ii) senão, EC={"adicione b a B "}.
Modificação de f	Suponha que u="modifique a com ($f = b$)". As alternativas de semânticas de atualização para u com relação a ψ são: a) $SM(f, \psi)$ = ***rejeita***, determina que: (i) se b=**nulo** ou $b \in \mathcal{B}(B)$, então EC="continue"; (ii) senão, EC="rollback". b) $SM(f, \psi)$ = ***propaga***, determina que: (i) se b=**nulo** ou $b \in \mathcal{B}(B)$, então EC="continue"; (ii) senão, EC={"adicione b a B"}.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO DE TOTALIDADE
$\psi = f{:}A \bullet\!\!\rightarrow B$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
*Adição em **B***	Suponha que u="adicione b em B ". Let $\mathcal{B}(A^*)=\{a \mid a \in \mathcal{B}(A)$ e $\mathcal{B}(f)(a)=b\}$. A única alternativa de semânticas de atualização para u com relação a ψ é: a) $SI(B,\ \psi)$ = ***propaga***, determina que: EC={"adicione $a(f{=}b)$ em A"}.
Remoção em A	Suponha que u="remova a de A ". Seja $b = \mathcal{B}(f)(a)$. Seja $\mathcal{B}(A^*)=\{a' \mid a' \in \mathcal{B}(A),\ a' \neq a$ e $\mathcal{B}(f)(a')=b\}$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SD(A,\ \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(A^*) \neq \varnothing$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(A,\ \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*) \neq \varnothing$, então EC="continue"; (ii) senão EC="remova b de B".
Modificação de f	Suponha que u="modifique a com $(f = b')$ ". Seja $b = \mathcal{B}(f)(a)$. Seja $\mathcal{B}(A^*)=\{a' \mid a' \in \mathcal{B}(A),\ a' \neq a$ e $\mathcal{B}(f)(a')=b\}$. As alternativas de semânticas de atualização para u com relação a ψ são: a) $SD(A,\ \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(A^*) \neq \varnothing$, então EC="continue"; (ii) senão, EC="rollback". b) $SD(A,\ \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*) \neq \varnothing$, então EC="continue"; (ii) senão EC="remova b de B".

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO DE CHAVE $\psi=(\{f_1,\dots,f_n\}\in CHAVES(A))$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u="adicione a a A ". A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se não existir $a'\in \mathcal{B}(A)$ tal que $\mathcal{B}(f_k)(a')=\mathcal{B}(f_k)(a)$, para $1\le k\le n$, então EC="continue"; (ii) senão, EC="rollback".
Modificação de f_k	Suponha que u="modifique a com $(f_k=v)$", onde $1\le k\le n$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f_k, \psi)$ = ***rejeita***, determina que: (i) se não existir $a'\in \mathcal{B}(A)$ tal que $\mathcal{B}(f_k)(a')=\mathcal{B}(f_k)(a)$, para $1\le k\le n$, então EC="continue"; (ii) senão, EC="rollback".

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO DE DEPENDÊNCIA FUNCIONAL $\psi=(A([f_1 \dots f_n]\rightarrow f)$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u= "adicione a a A ". Seja $\mathcal{B}(A^*)=\{a' \mid a'\in\mathcal{B}(A)$ e $\mathcal{B}(f_k)(a')=\mathcal{B}(f_k)(a)$, para $1\le k\le n$, e $\mathcal{B}(f)(a')\neq\mathcal{B}(f)(a)\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}(A^*)=\emptyset$, então EC="continue"; (ii) senão, EC="rollback".
Modificação de f	Suponha que u="modifique a com $(f=v)$". Seja $\mathcal{B}(A^*)=\{a' \mid a'\in\mathcal{B}(A)$ e $\mathcal{B}(f_k)(a')=\mathcal{B}(f_k)(a)$, para $1\le k\le n$, e $\mathcal{B}(f)(a')\neq\mathcal{B}(f)(a)\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*)=\emptyset$, então EC="continue"; (ii) se $\mathcal{B}(A^*)=\{a_1, \dots, a_m\}$ então $EC=\{u_k \mid u_k$="modifique a_k com $(f=\mathcal{B}(f)(a))$ ", para $1\le k\le m\}$.
Modificação de f_k	Suponha que u="modifique a com $(f_k=v)$", onde $1\le k\le n$. Seja $\mathcal{B}(A^*)=\{a' \mid a'\in\mathcal{B}(A)$ e $\mathcal{B}(f_k)(a')=\mathcal{B}(f_k)(a)$, para $1\le k\le n$, e $\mathcal{B}(f)(a')\neq\mathcal{B}(f)(a)\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*)=\emptyset$, então EC="continue"; (ii) se $\mathcal{B}(A^*)=\{a_1, \dots, a_m\}$, então $EC=\{u_j \mid u_j$="modifique a_j com $(f_k=\mathcal{B}(f)(a))$ ", para $1\le j\le m\}$.

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO DE DEPENDÊNCIA FUNCIONAL DE UNIÃO

$\psi=(<A_1([f1_1,\ldots,f1_m]\rightarrow f_1),\ldots, A_n([fn_1,\ldots,fn_m]\rightarrow f_n)>)$

Operação de atualização	**Alternativas de semânticas de atualização para manter ψ**
Adição em A	Suponha que u= "adicione a a A_k ", onde $1\le k\le n$. Seja $\mathcal{D}(A_j{*})=\{a' \mid a'\in\mathcal{D}(A_j)$ e $\mathcal{D}(fj_p)(a')=\mathcal{D}(fk_p)(a)$, para $1\le p\le m$, e $\mathcal{D}(fj)(a')\neq\mathcal{D}(fk)(a)\}$, para $1\le j\le n$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{D}(A_j{*})=\emptyset$, para $1\le j\le n$, então EC="continue"; (ii) senão, EC="rollback".
Modificação de f_k	Suponha que u="modifique a com $(f_k=v)$", onde $1\le k\le n$. Seja $\mathcal{D}(A_j{*})=\{a' \mid a'\in\mathcal{D}(A_j)$ e $\mathcal{D}(fj_p)(a')=\mathcal{D}(fk_p)(a)$, para $1\le p\le m$, e $\mathcal{D}(fj)(a')\neq\mathcal{D}(fk)(a)\}$, para $1\le j\le n$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{D}(A_j{*})=\emptyset$, para $1\le j\le n$, então EC="continue"; (ii) senão, para $1\le j\le n$, se $\mathcal{D}(A_j{*})=\{a_1, \ldots, a_q\}$, então $U_j=\{u_k \mid u_k$="modifique a_k com $(f_j=\mathcal{D}(f)(a))$ ", para $1\le k\le q\}$ $EC=\bigcup_{j=1}^{n} U_j$
Modificação de fk_p	Suponha que u="modifique a com $(fk_p=v)$", onde $1\le k\le n$ e $1\le p\le m$. Seja $\mathcal{D}(A_j{*})=\{a' \mid a'\in\mathcal{D}(A_j)$ e $\mathcal{D}(fj_p)(a')=\mathcal{D}(fk_p)(a)$, para $1\le p\le m$, e $\mathcal{D}(fj)(a')\neq\mathcal{D}(fk)(a)\}$, para $1\le j\le n$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(fk_p, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{D}(A_j{*})=\emptyset$, para $1\le j\le n$, então EC="continue"; (ii) senão, para $1\le j\le n$, se $\mathcal{D}(A_j{*})=\{a_1, \ldots, a_q\}$, então $U_j=\{u_k \mid u_k$="modifique a_k com $(fj_p=\mathcal{D}(fk_p)(a))$ ", para $1\le k\le q\}$. $EC=\bigcup_{j=1}^{n} U_j$

SEMÂNTICA DE ATUALIZAÇÃO PARA MANTER A RESTRIÇÃO DE EQUIVALÊNCIA_DE_RELACIONAMENTO

$\psi = (f \equiv f_n \circ \ldots \circ f_1)$ onde $f_1{:}A \to B_1$, ..., $f_n{:}B_{n-1} \to B_n$ e $f{:}A \to B_n$ são atributos.

Adição em A	Suponha que u="adicione $\boldsymbol{a}$ a A". Seja $c=\mathcal{B}\,(f_{n-1}\circ \ldots \circ f_1)(a)$ As alternativas de semânticas de atualização para u com relação a ψ são: a) $SI(A, \psi)$ = ***rejeita***, determina que: (i) se $\mathcal{B}\,(f)(a) \neq \mathcal{B}\,(f_n\circ \ldots \circ f_1)(a)$, então EC="rollback"; (ii) senão, EC="continue". b) $SI(A, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}\,(f)(a) \neq \mathcal{B}\,(f_n\circ \ldots \circ f_1)(a)$, então EC={"modifique $c$com $(f_n= \mathcal{B}\,(f)(a)\,)$"}; (ii) senão, EC="continue".
Modificação de f	Suponha que u="modifique $\boldsymbol{a}$ com $(f{=}v\,)$". Seja $c=\mathcal{B}\,(f_{n-1}\circ \ldots \circ f_1)(a)$ A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}\,(f)(a) \neq \mathcal{B}\,(f_n\circ \ldots \circ f_1)(a)$, então EC={"modifique c com $(f_n= \mathcal{B}\,(f)(a)\,)$"}; (ii) senão, EC="continue".
Modificação de f_k	Suponha que u="modifique c com $(f_k{=}v\,)$", onde $1 \le k \le n$. Seja $\mathcal{B}(A^*)=\{a \mid a\in \mathcal{B}(A)$ e $\mathcal{B}(f_{k-1}\circ \ldots \circ f_1)(a)=c\}$. A única alternativa de semântica de atualização para u com relação a ψ é: a) $SM(f_k, \psi)$ = ***propaga***, determina que: (i) se $\mathcal{B}(A^*)=\varnothing$, então EC="continue"; (ii) se $\mathcal{B}(A^*)=\{a_1, \ldots, a_m\}\neq\varnothing$, então $EC=\{u_i \mid u_i$="modifique a_i com $(f{=}\mathcal{B}(f_n\circ \ldots \circ f_1)(a_i))$", para $1 \le i \le m\}$.

APÊNDICE B

Nesse apêndice, nós selecionamos duas primitivas de transformações para provarmos que elas preservam a informação e a semântica das atualizações. Antes da apresentação dessas provas, enunciamos duas proposições que são usadas nessas provas.

Propriedade B.1: Seja **T** = (***pre***, ***pós***, ϕ, σ, τ) uma transformação que preserva a informação e a semântica das atualizações. Suponha que nós tenhamos um esquema S_1=($\mathcal{C}_1$, $\mathcal{A}_1$, $\mathcal{R}_1$) e um esquema S_2 =($\mathcal{C}_2$, $\mathcal{A}_2$, $\mathcal{R}_2$) tais que **T**(S_1)=S_2. Se a restrição $\psi_1 \in \mathcal{R}_1$ for diretamente transformada em uma restrição $\psi_2 \in \mathcal{R}_2$ então:

(i) $\forall$ ***u*** $\in$ ***atualizações***(S_1), ψ_1 é relevante para ***u*** sss ψ_2 é relevante para τ(***u***).

(ii) $\forall \mathcal{D}_1 \in$ ***instâncias***(S_1), onde $\mathcal{D}_2 = \phi(\mathcal{D}_1)$, temos que:

a) se ***co-efeito***(***u***, $\mathcal{D}_1$, ψ_1) = "aborte a presente transação" (***EC***= "rollback"), então ***co-efeito***(τ(***u***), $\mathcal{D}_2$, ψ_2) = "aborte a presente transação";

b) se ***co-efeito***(***u***, $\mathcal{D}_1$, ψ_1) = $\mathcal{D}_1$ (***EC***= "continue"), então ***co-efeito***(τ(***u***), $\mathcal{D}_2$, ψ_2) = $\mathcal{D}_2$ (***EC***= "continue");

c) se ***co-efeito***(***u***, $\mathcal{D}_1$, ψ_1) = $u_n \circ \ldots \circ u_1(\mathcal{D}_1)$ (***EC***={u_1, ..., u_n}), então ***co-efeito***(τ(***u***), $\mathcal{D}_2$, ψ_2) = $\tau(u_n) \circ \ldots \circ \tau(u_1)(\mathcal{D}_2)$ (***EC***= {$\tau(u_1)$, ..., $\tau(u_n)$}).

Para as proposições B.1 e B.2, suponha que a restrição $\psi_1 \in \mathcal{R}_1$ seja diretamente transformada em uma restrição $\psi_2 \in \mathcal{R}_2$. Suponha que nós tenhamos $\mathcal{D}_1 \in$ ***instâncias***(S_1) tal que $\mathcal{D}_2 = \phi(\mathcal{D}_1)$.

Proposição B.1: Seja ***u*** $\in$ ***atualizações***(S_1) onde ψ_1 é relevante para ***u*** e ***co-efeito***(***u***, $\mathcal{D}_1$, ψ_1) = $u_n \circ \ldots \circ u_1(\mathcal{D}_1)$. Se $\tau(u_k)$ for uma tradução correta de u_k, com $1 \leq k \leq n$, então ***co-efeito***(***u***, $\mathcal{D}_1$, ψ_1) = σ(***co-efeito***(τ(***u***), $\mathcal{D}_2$, ψ_2)).

Prova:

Pela propriedade B.1 temos que: ***co-efeito***(τ(***u***), $\mathcal{D}_2$, ψ_2)= $\tau(u_n) \circ \ldots \circ \tau(u_1)(\mathcal{D}_2)$.

Como $\tau(u_1)$ é uma tradução correta de u_1, pela Definição 3.2, temos que:

(1) $u_1(\mathcal{D}_1) = \sigma(\tau(u_1)(\mathcal{D}_2))$.

Suponha que $u_{k-1} \circ \ldots \circ u_1(\mathcal{D}_1) = \sigma(\tau(u_{k-1}) \circ \ldots \circ \tau(u_1)(\mathcal{D}_2))$, $1 < k < n$. Uma vez que $\tau(u_k)$ é uma tradução correta de u_k, pela Definição 3.2, temos que:

(2) $u_k \circ \ldots \circ u_1(\mathcal{D}_1) = \sigma(\tau(u_k) \circ \ldots \circ \tau(u_1)(\mathcal{D}_2))$.

Assim, por (1) e (2) temos que: $u_n \circ \ldots \circ u_1(\mathcal{D}_1) = \sigma(\tau(u_n) \circ \ldots \circ \tau(u_1)(\mathcal{D}_2))$ e, conseqüentemente, ***co-efeito***(***u***, $\mathcal{D}_1$, ψ_1) = σ(***co-efeito***(τ(***u***), $\mathcal{D}_2$, ψ_2)).

Proposição B.2: Seja $\boldsymbol{u}'$ uma atualização desencadeada durante o cálculo de $\boldsymbol{u}(\mathcal{B}_1)$ tal que ψ_1 seja relevante para $\boldsymbol{u}'$. Suponha que $\mathcal{B}_1{}^*$ seja obtido aplicando-se $\boldsymbol{u}(\mathcal{B}_1)$ e ignorando-se *co-efeito*$(\boldsymbol{u}', \mathcal{B}_1', \psi_1)$: $\mathcal{B}_1{}^* = \boldsymbol{u}(\mathcal{B}_1)$ - *co-efeito*$(\boldsymbol{u}', \mathcal{B}_1', \psi_1)$; e suponha que $\mathcal{B}_2{}^*$ seja obtido aplicando-se $\boldsymbol{\tau}(\boldsymbol{u})(\mathcal{B}_2)$ e ignorando-se *co-efeito*$(\boldsymbol{\tau}(\boldsymbol{u}'), \mathcal{B}_2', \psi_2)$: $\mathcal{B}_2{}^* = \boldsymbol{\tau}(\boldsymbol{u})(\mathcal{B}_2)$ - *co-efeito*$(\boldsymbol{\tau}(\boldsymbol{u}'), \mathcal{B}_2', \psi_2)$. Assim, se $\mathcal{B}_1{}^* = \boldsymbol{\sigma}(\mathcal{B}_2{}^*)$, então $\boldsymbol{u}(\mathcal{B}_1) = \boldsymbol{\sigma}(\boldsymbol{\tau}(\boldsymbol{u})(\mathcal{B}_2))$.

Prova:

Suponha que $\mathcal{B}_1{}^*=\boldsymbol{\sigma}(\mathcal{B}_2{}^*)$. Como assumimos que a ordem das operações desencadeadas seja irrelevante, temos que:

$\boldsymbol{u}(\mathcal{B}_1)=$ *co-efeito*$(\boldsymbol{u}', \mathcal{B}_1{}^*, \psi_1)$ e $\boldsymbol{\tau}(\boldsymbol{u})(\mathcal{B}_2)=$ *co-efeito*$(\boldsymbol{\tau}(\boldsymbol{u}'), \mathcal{B}_2{}^*, \psi_2)$.

Pela Proposição B.1, temos que *co-efeito*$(\boldsymbol{u}', \mathcal{B}_1{}^*, \psi_1)= \boldsymbol{\sigma}($*co-efeito*$(\boldsymbol{\tau}(\boldsymbol{u}'), \mathcal{B}_2{}^*, \psi_2))$.

Conseqüentemente, $\boldsymbol{u}(\mathcal{B}_1)= \boldsymbol{\sigma}(\boldsymbol{\tau}(\boldsymbol{u})(\mathcal{B}_2))$.

B.1 Prova de que a primitiva de transformação *DECOMPONHA* preserva as informação.

No restante dessa seção, seja $S_1=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ um esquema, onde $A \in \mathcal{C}$ e $f_{AC_1}: A \rightarrow C_1, \ldots, f_{AC_n}: A \rightarrow C_n \in \mathcal{A}$. Seja $S_2=$*DECOMPONHA*$(A, \{f_{AC_1}, \ldots, f_{AC_n}\}, B, f_{AB}, \{f_{BC_1}, \ldots, f_{BC_n}\})(S_1)$. Suponha que $\mathcal{B}_1$ seja uma instância de S_1 e que $\mathcal{B}_2$ seja uma instância de S_2 tais que $\mathcal{B}_2=\phi(\mathcal{B}_1)$.

Propriedade B.2: Para qualquer $b \in \mathcal{B}_2(B)$ existe $a \in \mathcal{B}_2(A)$ tal que $b=\mathcal{B}_2(f_{AB})(a)$ e $\mathcal{B}_2(f_{AC_k})(a)= \mathcal{B}_2(f_{BC_k})(b)$, para $1 \leq k \leq n$.

A Propriedade B.2 é uma conseqüência direta das pós-condições $\mathcal{B}_2(f_{AC_k})(a) = \mathcal{B}_2(f_{BC_k})(\mathcal{B}_2(f_{AB})(a))$ de *DECOMPONHA*.

Propriedade B.3: Para qualquer classe $D \in \mathcal{C}$ e para qualquer atributo $f: D \rightarrow C \in \mathcal{A}$, então:

(i) $\mathcal{B}_2(D)=\mathcal{B}_1(D)$ e

(ii) para qualquer $d \in \mathcal{B}_1(D)$, $\mathcal{B}_2(f)(a)= \mathcal{B}_1(f)(a)$.

A Propriedade B.3 é satisfeita porque todas as classes e atributos de S_1 são diretamente transformados em classes e atributos de S_2.

Lema B.1: A DF $A(f_{ACi_1}, ..., f_{ACi_m} \rightarrow f_{ACj})$, onde $i_k \in [1, n]$, para $1 \leq k \leq m$ e $j \in [1, n]$, é válida em $\mathcal{D}_1$ sss a DF $B(f_{BCi_1}, ..., f_{BCi_m} \rightarrow f_{BCj})$ for válida em $\mathcal{D}_2$.

Prova:

(i) Suponha que $A(f_{ACi_1}, ..., f_{ACi_m} \rightarrow f_{ACj})$ seja válida em $\mathcal{D}_1$. Suponha, também, que existam $\boldsymbol{b}$ e $\boldsymbol{b'}$ em $\mathcal{D}_2(\boldsymbol{B})$ tais que $\mathcal{D}_2(f_{BCi_k})(\boldsymbol{b}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b'})$, para $1 \leq k \leq m$, e que $\mathcal{D}_2(f_{BC_j})(\boldsymbol{b}) \neq \mathcal{D}_2(f_{BC_j})(\boldsymbol{b'})$. Pela propriedade B.2, existem $\boldsymbol{a}$ e $\boldsymbol{a'}$ em $\mathcal{D}_2(A)$ tais que: $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b})$ e $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a'}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b'})$, para $1 \leq k \leq m$; e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) = \mathcal{D}_2(f_{BC_j})(\boldsymbol{b})$ e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a'}) = \mathcal{D}_2(f_{BC_j})(\boldsymbol{b'})$. Conseqüentemente, $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(f_{ACi_k})(\boldsymbol{a'})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_2(f_{AC_j})(\boldsymbol{a'})$. Mas, pela Propriedade B.3, $\mathcal{D}_1(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_1(f_{ACi_k})(\boldsymbol{a'})$, para $1 \leq k \leq m$, e $\mathcal{D}_1(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_1(f_{AC_j})(\boldsymbol{a'})$, uma contradição.

(ii) Suponha que $B(f_{BCi_1}, ..., f_{BCi_m} \rightarrow f_{BCj})$ seja válida em $\mathcal{D}_2$. Suponha, também, que existam $\boldsymbol{a}$ e $\boldsymbol{a'}$ em $\mathcal{D}_1(A)$ tais que $\mathcal{D}_1(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_1(f_{ACi_k})(\boldsymbol{a'})$, para $1 \leq k \leq m$, e que $\mathcal{D}_1(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_1(f_{AC_j})(\boldsymbol{a'})$. Pela propriedade B.3, $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(f_{ACi_k})(\boldsymbol{a'})$, para $1 \leq k \leq m$ e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_2(f_{AC_j})(\boldsymbol{a'})$. Pela propriedade B.1, existem $\boldsymbol{b}$ e $\boldsymbol{b'}$ em $\mathcal{D}_2(A)$ tais que: $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b})$ e $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a'}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b'})$, para $1 \leq k \leq m$; e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) = \mathcal{D}_2(f_{BC_j})(\boldsymbol{b})$ e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a'}) = \mathcal{D}_2(f_{BC_j})(\boldsymbol{b'})$. Conseqüentemente, $\mathcal{D}_2(f_{BCi_k})(\boldsymbol{b}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b'})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{BC_j})(\boldsymbol{b}) \neq \mathcal{D}_2(f_{BC_j})(\boldsymbol{b'})$, uma contradição.

Lema B.2: Seja ψ uma DFU contendo $A(f_{ACi_1}, ..., f_{ACi_m} \rightarrow f_{ACj})$, onde $i_k \in [1, n]$ e $j \in [1,n]$; e seja ψ' uma DFU formada pela troca de $A(f_{ACi_1}, ..., f_{ACi_m} \rightarrow f_{ACj})$ por $B(f_{BCi_1}, ..., f_{BCi_m} \rightarrow f_{BCj})$ em ψ. ψ é válida em $\mathcal{D}_1$ sss ψ' for válida em $\mathcal{D}_2$.

Prova:

(i) Suponha que $\psi = \langle A(f_{ACi_1}, ..., f_{ACi_m} \rightarrow f_{ACj}), D(g_1, ..., g_m \rightarrow g) \rangle$ seja válida em $\mathcal{D}_1$. Suponha, também, que existam $\boldsymbol{b} \in \mathcal{D}_2(B)$ e $\boldsymbol{d} \in \mathcal{D}_2(D)$ tais que $\mathcal{D}_2(f_{BCi_k})(\boldsymbol{b}) = \mathcal{D}_2(g_k)(\boldsymbol{d})$, para $1 \leq k \leq m$ e que $\mathcal{D}_2(f_{BC_j})(\boldsymbol{b}) \neq \mathcal{D}_2(g)(\boldsymbol{d})$. Pela Propriedade B.2, existe $\boldsymbol{a} \in \mathcal{D}_2(A)$ tal que $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) = \mathcal{D}_2(f_{BC_j})(\boldsymbol{b})$. Conseqüentemente, $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(g_k)(\boldsymbol{d})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_2(g)(\boldsymbol{d})$. Mas, pela Propriedade B.3, $\mathcal{D}_1(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_1(g_k)(\boldsymbol{d})$, para $1 \leq k \leq m$, e $\mathcal{D}_1(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_1(g)(\boldsymbol{d})$, uma contradição.

(ii) Suponha que $\langle B(f_{BCi_1}, ..., f_{BCi_m} \rightarrow f_{BCj}), D(g_1, ..., g_m \rightarrow g) \rangle$ seja válida em $\mathcal{D}_2$. Suponha, também, que existam $\boldsymbol{a} \in \mathcal{D}_1(D)$ e $\boldsymbol{d} \in \mathcal{D}_1(D)$ tais que $\mathcal{D}_1(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_1(g_k)(\boldsymbol{d})$, para $1 \leq k \leq m$, e $\mathcal{D}_1(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_1(g)(\boldsymbol{d})$. Pela Propriedade B.3, $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(g_k)(\boldsymbol{d})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) \neq \mathcal{D}_2(g)(\boldsymbol{d})$. Pela propriedade B.2, existe $\boldsymbol{b}$ em $\mathcal{D}_2(A)$ tal que $\mathcal{D}_2(f_{ACi_k})(\boldsymbol{a}) = \mathcal{D}_2(f_{BCi_k})(\boldsymbol{b})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{AC_j})(\boldsymbol{a}) = \mathcal{D}_2(f_{BC_j})(\boldsymbol{b})$. Conseqüentemente, $\mathcal{D}_2(f_{BCi_k})(\boldsymbol{b}) = \mathcal{D}_2(g_k)(\boldsymbol{d})$, para $1 \leq k \leq m$, e $\mathcal{D}_2(f_{BC_j})(\boldsymbol{b}) \neq \mathcal{D}_2(g)(\boldsymbol{d})$, o que é uma contradição.

Lemma B.3: Seja ψ uma DE contendo $A[f_{ACi_1}, \ldots, f_{ACi_m}]$, onde $i_k \in [1, n]$, e seja ψ' uma DE formada pela troca de $A[f_{ACi_1}, \ldots, f_{ACi_m}]$ por $B[f_{BCi_1}, \ldots, f_{BCi_m}]$ em ψ. ψ é válida em $\mathcal{I}_1$ sss ψ' for válida em $\mathcal{I}_2$. Temos que provar esse lema para todos os tipos de DEs contendo $A[f_{ACi_1}, \ldots, f_{ACi_m}]$.

Caso 1: $\psi = A[f_{ACi_1}, \ldots, f_{ACi_m}] \subset D[g_1, \ldots, g_m]$

(i) Suponha que $A[f_{ACi_1}, \ldots, f_{ACi_m}] \subset D[g_1, \ldots, g_m]$ seja válida em $\mathcal{I}_1$. Assim, para qualquer $a \in \mathcal{I}_1(A)$, existe $d \in \mathcal{I}_1(D)$ tal que $\mathcal{I}_1(f_{ACi_k})(a) = \mathcal{I}_1(g_k)(d)$, para $1 \leq k \leq m$. Pela Propriedade B.3, para qualquer $a \in \mathcal{I}_2(A)$, existe $d \in \mathcal{I}_2(D)$ tal que $\mathcal{I}_2(f_{ACi_k})(a) = \mathcal{I}_2(g_k)(d)$, para $1 \leq k \leq m$. Pela propriedade B.2, para qualquer $b \in \mathcal{I}_2(B)$, existe $d \in \mathcal{I}_2(D)$ tal que $\mathcal{I}_2(f_{BCi_k})(b) = \mathcal{I}_2(g_k)(d)$, para $1 \leq k \leq m$. Conseqüentemente, $B[f_{BCi_1}, \ldots, f_{BCi_m}] \subset D[g_1, \ldots, g_m]$ é válida em $\mathcal{I}_2$.

(ii) Suponha que a DE $B[f_{BCi_1}, \ldots, f_{BCi_m}] \subset D[g_1, \ldots, g_m]$ seja válida em $\mathcal{I}_2$. Assim, para qualquer $b \in \mathcal{I}_2(B)$ existe $d \in \mathcal{I}_2(D)$ tal que $\mathcal{I}_2(f_{BCi_k})(b) = \mathcal{I}_2(g_k)(d)$, para $1 \leq k \leq m$. Pela propriedade B.2, para qualquer $a \in \mathcal{I}_2(A)$ existe $d \in \mathcal{I}_2(D)$ tal que $\mathcal{I}_2(f_{ACi_k})(a) = \mathcal{I}_2(g_k)(d)$, for $1 \leq k \leq m$. Pela propriedade B.3, para qualquer $a \in \mathcal{I}_1(A)$ existe $d \in \mathcal{I}_1(D)$ tal que $\mathcal{I}_1(f_{ACi_k})(a) = \mathcal{I}_1(g_k)(d)$, para $1 \leq k \leq m$. Conseqüentemente, $A[f_{ACi_1}, \ldots, f_{ACi_m}] \subset D[g_1, \ldots, g_m]$ é válida em $\mathcal{I}_1$.

As provas para os outros tipos de DEs são muito similares e são omitidas.

Lema B.4: $\mathcal{I}_1 = \sigma(\phi(\mathcal{I}_1))$

Prova:

Como qualquer classe de S_1 é diretamente transformada em uma classe de S_2, então, para qualquer classe C de S_1, $\mathcal{I}_1(C) = \sigma(\phi(\mathcal{I}_1))(C)$ (pela Propriedade 4.3). Como qualquer atributo em S_1 é diretamente transformado em um atributo de S_2, então, para qualquer atributo $f: A \rightarrow C$ de S_1, e para qualquer $e \in \mathcal{I}_1(A)$, $\mathcal{I}_1(f)(e) = \sigma(\phi(\mathcal{I}_1))(f)(e)$ (pela Propriedade 4.3). Conseqüentemente, pela definição de equivalência de instâncias (Definição 2.2) $\mathcal{I}_1 = \sigma(\phi(\mathcal{I}_1))$.

Teorema B.1: A primitiva de transformação ***DECOMPONHA*** preserva a informação.

Prova: Pelos lemas B.1, B.2, B.3 e B.4, podemos concluir que a transformação ***DECOMPONHA*** preserva a informação.

B.2 Prova de que a primitiva de transformação *DECOMPONHA* preserva a semântica das atualizações.

Nessa seção, seja $S_1=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ um esquema, onde $A \in \mathcal{C}$ e $f_{AC_1}: A \rightarrow C_1, \ldots, f_{AC_n}: A \rightarrow C_n \in rel(\mathcal{A})$. Seja $S_2=(\mathcal{C}', \mathcal{A}', \mathcal{R}')$ tal que S_2=*DECOMPONHA*$(A, \{f_{AC_1},\ldots, f_{AC_n}\}, B, f_{AB}, \{f_{BC_1}, \ldots, f_{BC_n}\})(S_1)$. Sem perda de generalidade (vide Proposição B.2), suponha que as únicas DFs, DFUs e DEs em $\mathcal{R}$ sejam:

$\psi_1 = A(f_{ACi_1}, \ldots, f_{ACi_m} \rightarrow f_{ACj})$, onde $1 \leq j$, $i_k \leq n$.

$\psi_2 = \langle A(f_{ACi_1}, \ldots, f_{ACi_m} \rightarrow f_{ACj}), E(g_1, \ldots, g_m \rightarrow g)\rangle$ onde $1 \leq j$, $i_k \leq n$.

$\psi_3 = A[f_{ACi_1}, \ldots, f_{ACi_m}] \subset D[g_1, \ldots, g_m]$, onde $D \in \mathcal{C}$ e $1 \leq i_k \leq n$.

$\psi_4 = H[g_1, \ldots, g_m] \subset A[f_{ACi_1}, \ldots, f_{ACi_m}]$, onde $H \in \mathcal{C}$ e $1 \leq i_k \leq n$..

Conseqüentemente, as únicas DFs, DFUs e DEs em $\mathcal{R}'$ são:

$\psi'_1 = B(f_{BCi_1}, \ldots, f_{BCi_m} \rightarrow f_{BCj})$, onde $i_k \in [1, n]$ para $1 \leq k \leq m$ e $j \in [1, n]$.

$\psi'_2 = \langle B(f_{BCi_1}, \ldots, f_{BCi_m} \rightarrow f_{BCj}), E(g_1, \ldots, g_m \rightarrow g)\rangle$ onde $i_k \in [1, n]$ para $1 \leq k \leq m$ e $j \in [1,n]$.

$\psi'_3 = B[f_{BCi_1}, \ldots, f_{BCi_m}] \subset D[g_1, \ldots, g_m]$, onde $D \in \mathcal{C}$ e $i_k \in [1, n]$ para $1 \leq k \leq m$.

$\psi'_4 = H[g_1, \ldots, g_m] \subset B[f_{BCi_1}, \ldots, f_{BCi_m}]$, onde $H \in \mathcal{C}$ e $i_k \in [1, n]$ para $1 \leq k \leq m$.

Nós provamos o teorema, mostrando que a semântica de qualquer atualização u em ***atualizações***(S_1) é preservada em S_2. Para cada atualização u temos que considerar todos os estados distintos nos quais u tem tipos distintos de efeitos colaterais.

1- Seja u= "remova a de A" (do esquema S_1).

Consideremos $\mathcal{B}_1 \in$ ***instâncias***(S_1) e $\mathcal{B}_2 \in$ ***instâncias***(S_1) onde $\mathcal{B}_2=\phi(\mathcal{B}_1)$. Como ψ_4 é a única restrição relevante em S_1, nós só precisamos considerar o caso ψ_4.

Definamos:

$\mathcal{B}_1(A^*)=\{a' \mid a' \in \mathcal{B}_1(A), a' \neq a$, e $\mathcal{B}_1(f_{ACi_k})(a')=\mathcal{B}_1(f_{ACi_k})(a)$, para $1 \leq k \leq m\}$,

$\mathcal{B}_1(H^*)=\{h \mid h \in \mathcal{B}_1(H)$ e $\mathcal{B}_1(f_{ACi_k})(a)=\mathcal{B}_1(g_k)(h)$, para $1 \leq k \leq m\}$,

$b=\mathcal{B}_2(f_{AB})(a)$,

$\mathcal{B}_2(A^*)=\{a' \mid a' \in \mathcal{B}_1(A), a' \neq a$, e $\mathcal{B}_2(f_{ACi_k})(a')=\mathcal{B}_2(f_{ACi_k})(a)$, para $1 \leq k \leq m\}$ e

$\mathcal{B}_2(H^*)=\{h \mid h \in \mathcal{B}_2(H)$ e $\mathcal{B}_2(f_{BCi_k})(b)=\mathcal{B}_2(g_k)(h)$, para $1 \leq k \leq m\}$.

Pelas propriedades B.2 e B.3 temos que: $\mathcal{B}_1(A^*)=\mathcal{B}_2(A^*)$ e $\mathcal{B}_1(H^*)=\mathcal{B}_2(H^*)$.

<u>Caso A</u>. Suponha que $\mathcal{B}_1(A^*) \neq \emptyset$.

1. Cálculo de $u(\mathcal{B}_1)$

Seja $\mathcal{B}_1{}^1$= ***efeito***$(u, \mathcal{B}_1)$. $\mathcal{B}_1{}^1$ é idêntico a $\mathcal{B}_1$ exceto que:

(A.1) $\mathcal{B}_1{}^1(A)=\mathcal{B}_1{}^1(A) - \{a\}$

Como $relevantes(u)=\{\psi_4\}$, temos que:

$co\text{-}efeito^*(u, \mathcal{B}_1^1) = co\text{-}efeito(u, \mathcal{B}_1^1, \psi_4)$.

Como EC="continue" é o efeito colateral requerido pela semântica de atualização de u com relação a ψ_4 (ψ_4 não é violada por u em $\mathcal{B}_1$), temos que:

$co\text{-}efeito(u, \mathcal{B}_1^1, \psi_4) = \mathcal{B}_1^1$.

Logo,

$co\text{-}efeito^*(u, \mathcal{B}_1^1) = \mathcal{B}_1^1$

Conseqüentemente,

(A.2) $u(\mathcal{B}_1)=co\text{-}efeito^*(u, efeito(u, \mathcal{B}_1)) = \mathcal{B}_1^1$.

2. Cálculo de $\tau(u)(\mathcal{B}_2)$

Seja $u_\tau = \tau(u)$ = "remova a de A "(do esquema S_2).

Como $\mathcal{B}_1(A^*)=\mathcal{B}_2(A^*)$, temos que $\mathcal{B}_2(A^*)\neq\emptyset$. Assim $|f_{AB}^{-1}(f_{AB}(a))| >1$.

Seja $\mathcal{B}_2^1 = efeito(u_\tau, \mathcal{B}_2)$. $\mathcal{B}_2^1$ é idêntico a $\mathcal{B}_2$ exceto que:

(A.3) $\mathcal{B}_2^1(A)=\mathcal{B}_2(A) - \{a\}$

Como $relevantes(u_\tau)=\{\psi = f_{AB}{:}A \bullet\!\!\longrightarrow B\}$, temos que:

$co\text{-}efeito^*(u_\tau, \mathcal{B}_2) = co\text{-}efeito(u_\tau, \mathcal{B}_2^1, \psi)$.

Como EC="continue" é o efeito colateral requerido pela semântica de atualização de u_τ com relação a ψ (ψ não é violada por u em $\mathcal{B}_2^1$), temos que:

$co\text{-}efeito(u_\tau, \mathcal{B}_2^1, \psi) = \mathcal{B}_2^1$.

Logo,

$co\text{-}efeito^*(u, \mathcal{B}_2^1) = \mathcal{B}_2^1$

Conseqüentemente,

(A.4) $u_\tau(\mathcal{B}_2)=co\text{-}efeito^*(u_\tau, efeito(u_\tau, \mathcal{B}_2)) = \mathcal{B}_2^1$.

Como $\sigma(\mathcal{B}_2^1)(A) = \mathcal{B}_2^1(A)$, por (A.3) temos que:

(A.5) $\sigma(\mathcal{B}_2^1)(A) = \mathcal{B}_2(A) - \{a\}$,

Mas $\mathcal{B}_2(A)=\mathcal{B}_1(A)$. Assim, por (A.1) e (A.5) temos que:

(A.6) $\sigma(\mathcal{B}_2^1)(A) = \mathcal{B}_1^1(A)$

Por (A.1), (A.3) e (A.6) temos que:

(A.7) $\mathcal{B}_1^1 = \sigma(\mathcal{B}_2^1)$.

Por (A.2), (A.4) e (A.7) temos que: $u(\mathcal{B}_1)=\sigma(u_\tau(\mathcal{B}_2))$

Conseqüentemente, $u(\mathcal{B}_1) = \sigma(\tau(u)(\phi(\mathcal{B}_1)))$.

Caso B. Suponha que $\mathcal{B}_1(A^*)=\varnothing$ e que $\mathcal{B}_1(H^*)=\varnothing$.

1. Cálculo de $u(\mathcal{B}_1)$.

Seja $\mathcal{B}_1{}^1$= *efeito*$(u, \mathcal{B}_1)$. $\mathcal{B}_1{}^1$ é idêntico a $\mathcal{B}_1$ exceto que:

(B.1) $\mathcal{B}_1{}^1(A)=\mathcal{B}_1{}^1(A) - \{a\}$

Como *relevantes*$(u)=\{\psi_4\}$, temos que:

co-efeito$^*(u, \mathcal{B}_1{}^1)$ = *co-efeito*$(u, \mathcal{B}_1{}^1, \psi_4)$.

Como EC="continue" é o efeito colateral requerido pela semântica de atualização de u com relação a ψ_4 (ψ_4 não é violada por u em $\mathcal{B}_1$), temos que:

co-efeito$(u, \mathcal{B}_1{}^1, \psi_4) = \mathcal{B}_1{}^1$.

Logo,

co-efeito$^*(u, \mathcal{B}_1{}^1) = \mathcal{B}_1{}^1$

Conseqüentemente,

(B.2) $u(\mathcal{B}_1)$=*co-efeito*$^*(u,$ *efeito*$(u, \mathcal{B}_1)) = \mathcal{B}_1{}^1$.

2. Cálculo de $\tau(u)(\mathcal{B}_2)$

Seja $u_\tau = \tau(u)$ = "remova a de A" (do esquema S_2).

Como $\mathcal{B}_1(A^*)=\mathcal{B}_2(A^*)$, temos que $\mathcal{B}_2(A^*)=\varnothing$. Seja $b=f_{AB}(a)$. Assim $|f_{AB}{}^{-1}(b)| = 1$.

Seja $\mathcal{B}_2{}^1$= *efeito*$(u_\tau, \mathcal{B}_2)$. $\mathcal{B}_2{}^1$ é idêntico a $\mathcal{B}_2$ exceto que:

(B.3) $\mathcal{B}_2{}^1(A)=\mathcal{B}_2(A)-\{a\}$

Como *relevantes*$(u_\tau)=\{\psi = f_{AB}: A \bullet\!\!\rightarrow B\}$, temos que:

co-efeito$^*(u_\tau, \mathcal{B}_2{}^1)$ = *co-efeito*$(u_\tau, \mathcal{B}_2{}^1, \psi)$.

Como $EC=\{u^1$="remova b de B"$\}$ é o efeito colateral requerido pela semântica de atualização de u_τ com relação a ψ ($SD(A, \psi)$=*propaga*), temos que:

co-efeito$(u_\tau, \mathcal{B}_2{}^1, \psi) = u^1(\mathcal{B}_2{}^1)$

2.1 Cálculo de $u^1(\mathcal{B}_2)$

Seja $\mathcal{B}_2{}^2$= *efeito*$(u^1, \mathcal{B}_2{}^1)$. $\mathcal{B}_2{}^2$ é idêntico a $\mathcal{B}_2{}^1$ exceto que:

(B.4) $\mathcal{B}_2{}^2(B)=\mathcal{B}_2{}^1(B)-\{b\}$

Como *relevantes*$(u^1)=\{\psi'_4\}$, temos que:

co-efeito$^*(u^1, \mathcal{B}_2{}^2)$ = *co-efeito*$(u^1, \mathcal{B}_2{}^2, \psi'_4)$

Como EC="continue" é o efeito colateral requerido pela semântica de atualização de u^1 com relação a ψ'_4 (ψ'_4 não é violada por u^1 em $\mathcal{B}_2{}^2$), temos que:

co-efeito$(u^1, \mathcal{B}_2{}^2, \psi'_4) = \mathcal{B}_2{}^2$

Logo,

co-efeito$^*(u^1, \mathcal{B}_2{}^2) = \mathcal{B}_2{}^2$

Conseqüentemente,

$u^1(\mathcal{B}_2)$ =*co-efeito*$^*(u^1,$ *efeito*$(u^1, \mathcal{B}_2{}^1)) = \mathcal{B}_2{}^2$.

(continuação do cálculo de $\boldsymbol{u}_\tau(\mathscr{D}_2)$)

(B.5) $\boldsymbol{u}_\tau(\mathscr{D}_2) = \textit{co-efeito*}(\boldsymbol{u}_\tau,\ \mathscr{D}_2^1) = \textit{co-efeito}(\boldsymbol{u}_\tau,\ \mathscr{D}_2^1, \psi) = \boldsymbol{u}^1(\mathscr{D}_2^1) = \mathscr{D}_2^2$

Como $\boldsymbol{\sigma}(\mathscr{D}_2^2)(A) = \mathscr{D}_2^2(A)$, por (A.3) e (A.4) temos que:

(B.6) $\boldsymbol{\sigma}(\mathscr{D}_2^2)(A) = \mathscr{D}_2(A) - \{\boldsymbol{a}\}$,

Mas $\mathscr{D}_2(A) = \mathscr{D}_1(A)$. Assim, por (A.1) e (A.3), temos que:

(B.7) $\boldsymbol{\sigma}(\mathscr{D}_2^2)(A) = \mathscr{D}_1^1(A)$

Por (B.1), (B.3), (B.4) e (B.6) temos que:

(B.8) $\mathscr{D}_1^1 = \boldsymbol{\sigma}(\mathscr{D}_2^1)$.

Por (B.2), (B.5) e (B.8) temos que: $\boldsymbol{u}(\mathscr{D}_1) = \boldsymbol{\sigma}(\boldsymbol{u}_\tau(\mathscr{D}_2))$

Conseqüentemente, $\boldsymbol{u}(\mathscr{D}_1) = \boldsymbol{\sigma}(\boldsymbol{\tau}(\boldsymbol{u})(\boldsymbol{\phi}(\mathscr{D}_1)))$.

Caso C. Suponha que $\mathcal{B}_1(H^*)=\{h_1, \ldots, h_n\}$, n≥1, e que $\mathcal{B}_1(A^*)=\varnothing$. Como $\mathcal{B}_2(H^*)=\mathcal{B}_1(H^*)$ e $\mathcal{B}_2(A^*)=\mathcal{B}_1(A^*)$, temos que: $\mathcal{B}_2(H^*)=\{h_1, \ldots, h_n\}$, $\mathcal{B}_2(A^*)=\varnothing$ e $\mathcal{B}_2(f_{AB}{}^{-1})(b))=\{a\}$.
Nós temos que considerar todas as alternativas possíveis de semânticas de atualização para ψ_4.

Caso C1: Suponha que $SD(A,\psi_4)$=*rejeita*,
1. Cálculo de $u(\mathcal{B}_1)$
Seja $\mathcal{B}_1{}^1$= *efeito*$(u, \mathcal{B}_1)$. $\mathcal{B}_1{}^1$ é idêntico a $\mathcal{B}_1$ exceto que:
(C1.1) $\mathcal{B}_1{}^1(A)=\mathcal{B}_1{}^1(A) - \{a\}$
Como *relevantes*$(u)=\{\psi_4\}$, temos que:
*co-efeito**$(u, \mathcal{B}_1{}^1)$ = *co-efeito*$(u, \mathcal{B}_1{}^1, \psi_4)$.
Como EC="rollback" é o efeito colateral requerido pela semântica de atualização de u com relação a ψ_4, temos que:
co-efeito$(u, \mathcal{B}_1, \psi_4)$ = "aborte a presente transação"
Conseqüentemente,
(C1.2) $u(\mathcal{B}_1)=\mathcal{B}_1$.

2. Cálculo de $\tau(u)(\mathcal{B}_2)$.
Seja $u_\tau = \tau(u)$ = "remova a de A" (do esquema S_2).
Seja $\mathcal{B}_2{}^1$= *efeito*$(u_\tau, \mathcal{B}_2)$. $\mathcal{B}_2{}^1$ é idêntico a $\mathcal{B}_2$ exceto que:
(C1.3) $\mathcal{B}_2{}^1(A)=\mathcal{B}_2(A)-\{a\}$
Como *relevantes*$(u_\tau)=\{\psi = f_{AB}:A \bullet\!\!\rightarrow B\}$, temos que:
*co-efeito**$(u_\tau, \mathcal{B}_2{}^1)$ = *co-efeito*$(u_\tau, \mathcal{B}_2{}^1, \psi)$.
Como $EC=\{u^1$="remova b de B"$\}$ é o efeito colateral requerido pela semântica de atualização de u_τ com relação a ψ ($SD(A, \psi)$=*propaga*), temos que:
co-efeito$(u_\tau, \mathcal{B}_2{}^1, \psi) = u^1(\mathcal{B}_2{}^1)$.
2.1 Cálculo de $u^1(\mathcal{B}_2)$
Seja $\mathcal{B}_2{}^2$= *efeito*$(u^1, \mathcal{B}_2{}^1)$. $\mathcal{B}_2{}^2$ é idêntico a $\mathcal{B}_2{}^1$ exceto que:
(C1.4) $\mathcal{B}_2{}^2(B)=\mathcal{B}_2{}^1(B)-\{b\}$
Como *relevantes*$(u^1)=\{\psi'_4\}$, temos que:
*co-efeito**$(u^1, \mathcal{B}_2{}^2)$ = *co-efeito*$(u^1, \mathcal{B}_2{}^2, \psi'_4)$
Como EC="rollback" é o efeito colateral requerido pela semântica de atualização de u^1 com relação a ψ'_4 ($SD(B, \psi'_4)$ = *propaga*), temos que:
co-efeito$(u^1, \mathcal{B}_2{}^2, \psi'_4)$ = "aborte a presente transação"
Conseqüentemente,
(C1.4) $u_\tau(\mathcal{B}_2)=\mathcal{B}_2$.
mas $\mathcal{B}_1=\sigma(\mathcal{B}_2)$.
Por (C1.2) e (C1.4) temos que: $u(\mathcal{B}_1)=\sigma(\tau(u)(\mathcal{B}_2))$
Conseqüentemente, $u(\mathcal{B}_1) = \sigma(\tau(u)(\phi(\mathcal{B}_1)))$.

Caso C2: Suponha que $SD(A,\psi_4)$=*propaga*.

1. Cálculo de $u(\mathcal{B}_1)$

Seja $\mathcal{B}_1^1$= *efeito*$(u, \mathcal{B}_1)$. $\mathcal{B}_1^1$ é idêntico a $\mathcal{B}_1$ exceto que:

(C2.1) $\mathcal{B}_1^1(A)=\mathcal{B}_1^1(A) - \{a\}$

Como *relevantes*$(u)=\{\psi_4\}$, temos que:

*co-efeito**$(u, \mathcal{B}_1^1)$ = *co-efeito*$(u, \mathcal{B}_1^1, \psi_4)$.

Como $EC=\{u_1^1$= "remova h_1 de H", ..., u_n^1= "remova h_n de H"$\}$ é o efeito colateral requerido pela semântica de atualização de u com relação a ψ_4, temos que:

co-efeito$(u, \mathcal{B}_1^1, \psi_4) = u_n^1 \circ ... \circ u_1^1(\mathcal{B}_1^1)$

1.2 Cálculo de $u_k^1(\mathcal{B}_1^k)$ onde $\mathcal{B}_1^k = u_{k-1}^1(\mathcal{B}_1^{k-1})$, para $1<k\leq n$.

Seja $\mathcal{B}_1^{k+1}$= *efeito*$(u_k^1, \mathcal{B}_1^k)$. $\mathcal{B}_1^{k+1}$ é idêntico a $\mathcal{B}_1^k$ exceto que:

(C2.2) $\mathcal{B}_1^{k+1}(H)=\mathcal{B}_1^k(H) - \{h_k\}$

Como *relevantes*$(u_k^1)=\varnothing$, temos que:

*co-efeito**$(u_k^1, \mathcal{B}_1^k) = \mathcal{B}_1^{k+1}$.

Conseqüentemente,

$u_k^1(\mathcal{B}_1^k)$= *co-efeito**$(u_k^1$, *efeito*$(u_k^1, \mathcal{B}_1^k)) = \mathcal{B}_1^{k+1}$.

(Continuação do cálculo de $u(\mathcal{B}_1)$)

*co-efeito**$(u, \mathcal{B}_1)$ = *co-efeito*$(u, \mathcal{B}_1^1, \psi_4) = u_n^1 \circ ... \circ u_1^1(\mathcal{B}_1^1) = \mathcal{B}_1^{n+1}$

Por (C2.1) e (C2.2), temos que $\mathcal{B}_1^{n+1}$ é idêntico a $\mathcal{B}_1$ exceto que:

(C2.3) $\mathcal{B}_1^{n+1}(A)=\mathcal{B}_1(A) - \{a\}$

$\mathcal{B}_1^{n+1}(H)=\mathcal{B}_1(H) - \{h_1, ..., h_n\}$

2. Cálculo de $\tau(u)(\mathcal{B}_2)$

Seja $u_\tau = \tau(u)$ = "remova a de A" (do esquema S_2).

Seja $\mathcal{B}_2^1$= *efeito*$(u_\tau, \mathcal{B}_2)$. $\mathcal{B}_2^1$ é idêntico a $\mathcal{B}_2$ exceto que:

(C2.4) $\mathcal{B}_2^1(A)=\mathcal{B}_2(A)-\{a\}$

Como *relevantes*$(u_\tau)=\{\psi = f_{AB}:A \bullet\!\!\rightarrow B\}$, temos que:

*co-efeito**$(u_\tau, \mathcal{B}_2^1)$ = *co-efeito*$(u_\tau, \mathcal{B}_2^1, \psi)$.

Como $EC=\{u^1$="remova b de B"$\}$ é o efeito colateral requerido pela semântica de atualização de u_τ com relação a ψ ($SD(A, \psi)$=*propaga*), temos que:

co-efeito$(u_\tau, \mathcal{B}_2^1, \psi) = u^1(\mathcal{B}_2^1)$.

2.1 Cálculo de $\boldsymbol{u}^1(\mathcal{D}_2)$

Seja $\mathcal{D}_2^2 = \textit{efeito}(\boldsymbol{u}^1, \mathcal{D}_2^1)$. $\mathcal{D}_2^2$ é idêntico a $\mathcal{D}_2^1$ exceto que:

(C2.5) $\mathcal{D}_2^2(B) = \mathcal{D}_2^1(B) - \{\boldsymbol{b}\}$

Como $\textit{relevantes}(\boldsymbol{u}^1) = \{\psi'_4\}$, temos que:

$\textit{co-efeito*}(\boldsymbol{u}^1, \mathcal{D}_2^2) = \textit{co-efeito}(\boldsymbol{u}^1, \mathcal{D}_2^2, \psi'_4)$

Como $EC = \{\boldsymbol{u}_1^2 =$ "remova $\boldsymbol{h}_1$ de $\boldsymbol{H}$", ..., $\boldsymbol{u}_n^2 =$ "remova $\boldsymbol{h}_n$ de $\boldsymbol{H}$"$\}$ é o efeito colateral requerido pela semântica de atualização de $\boldsymbol{u}_\tau$ com relação a ψ'_4 temos que:

$\textit{co-efeito}(\boldsymbol{u}_\tau, \mathcal{D}_2, \psi'_4) = \boldsymbol{u}_n^2 \circ \ldots \circ \boldsymbol{u}_1^2(\mathcal{D}_2^2)$

2.2 Cálculo de $\boldsymbol{u}_k^2(\mathcal{D}_2^{k+1})$ onde $\mathcal{D}_2^{k+1} = \boldsymbol{u}_{k-1}^2(\mathcal{D}_2^{k-2})$, para $1<k\leq n$.

Seja $\mathcal{D}_2^{k+2} = \textit{efeito}(\boldsymbol{u}_k^2, \mathcal{D}_2^{k+1})$. $\mathcal{D}_2^{k+2}$ é idêntico a $\mathcal{D}_2^{k+1}$ exceto que:

(C2.6) $\mathcal{D}_2^{k+2}(H) = \mathcal{D}_2^{k+1}(H) - \{\boldsymbol{h}_k\}$

Como $\textit{relevantes}(\boldsymbol{u}_k^2) = \varnothing$, temos que:

$\textit{co-efeito*}(\boldsymbol{u}_k^2, \mathcal{D}_2^{k+1}) = \mathcal{D}_2^{k+2}$.

Conseqüentemente,

$\boldsymbol{u}_k^2(\mathcal{D}_2^{k+1}) = \textit{co-efeito*}(\boldsymbol{u}_k^2, \textit{efeito}(\boldsymbol{u}_k^2, \mathcal{D}_2^{k+1}) = \mathcal{D}_2^{k+2}$

(continuação do cálculo de $\boldsymbol{u}^1(\mathcal{D}_2)$)

$\textit{co-efeito*}(\boldsymbol{u}^1, \mathcal{D}_2^2) = \textit{co-efeito}(\boldsymbol{u}^1, \mathcal{D}_2^2, \psi'_4) = \boldsymbol{u}_n^2 \circ \ldots \circ \boldsymbol{u}_1^2(\mathcal{D}_2^2) = \mathcal{D}_2^{n+2}$

Conseqüentemente,

$\boldsymbol{u}^1(\mathcal{D}_2) = \textit{co-efeito*}(\boldsymbol{u}^1, \textit{efeito}(\boldsymbol{u}^1, \mathcal{D}_2^1)) = \mathcal{D}_2^{n+2}$.

(continuação do cálculo de $\boldsymbol{u}_\tau(\mathcal{D}_2)$)

$\textit{co-efeito*}(\boldsymbol{u}_\tau, \mathcal{D}_2^1) = \textit{co-efeito}(\boldsymbol{u}_\tau, \mathcal{D}_2^1, \psi) = \boldsymbol{u}^1(\mathcal{D}_2) = \mathcal{D}_2^{n+2}$

Conseqüentemente,

(C2.7) $\boldsymbol{u}_\tau(\mathcal{D}_2) = \textit{co-efeito*}(\boldsymbol{u}_\tau, \textit{efeito}(\boldsymbol{u}_\tau, \mathcal{D}_2)) = \mathcal{D}_2^{n+2}$.

Por (C2.4), (C2.5), (C2.6) e (C2.7) temos que $\mathcal{D}_2^{n+2}$ é idêntico a $\mathcal{D}_2$ exceto que:

(C2.9) $\mathcal{D}_2^{n+2}(A) = \mathcal{D}_2(A) - \{\boldsymbol{a}\}$

$\mathcal{D}_2^{n+2}(H) = \mathcal{D}_2(H) - \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}$

$\mathcal{D}_2^{n+2}(B) = \mathcal{D}_2(B) - \{\boldsymbol{b}\}$

Como $\boldsymbol{\sigma}(\mathcal{D}_2^{n+2})(A) = \mathcal{D}_2^{n+2}(A)$ e $\boldsymbol{\sigma}(\mathcal{D}_2^{n+2})(H) = \mathcal{D}_2^{n+2}(H)$, por (C2.9) temos que:

(C2.10) $\boldsymbol{\sigma}(\mathcal{D}_2^{n+2})(A) = \mathcal{D}_2(A) - \{\boldsymbol{a}\}$,

$\boldsymbol{\sigma}(\mathcal{D}_2^{n+2})(H) = \mathcal{D}_2(H) - \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}$

Mas $\mathcal{D}_2(A) = \mathcal{D}_1(A)$ e $\mathcal{D}_2(H) = \mathcal{D}_1(H)$. Assim, por (C2.3) e (C2.10) temos que:

(C2.11) $\boldsymbol{\sigma}(\mathcal{D}_2^{n+2})(A) = \mathcal{D}_1^{n+1}(A)$

$\boldsymbol{\sigma}(\mathcal{D}_2^{n+2})(H) = \mathcal{D}_1^{n+1}(H)$

Por (C2.3), (C2.9) e (C2.11) temos que:

(C2.12) $\mathcal{D}_1^{n+1} = \boldsymbol{\sigma}(\mathcal{D}_2^{n+2})$.

(C2.12) $\mathcal{B}_1^{n+1} = \boldsymbol{\sigma}(\mathcal{B}_2^{n+2})$.

Por (C2.2), (C2.7) e (C2.12) temos que: $\boldsymbol{u}(\mathcal{B}_1)=\boldsymbol{\sigma}(\boldsymbol{u}_\tau(\mathcal{B}_2))$

Conseqüentemente, $\boldsymbol{u}(\mathcal{B}_1) = \boldsymbol{\sigma}(\boldsymbol{\tau}(\boldsymbol{u})(\boldsymbol{\phi}(\mathcal{B}_1)))$.

As provas para os outros tipos de atualizações são similares e serão omitidas aquí.

B.3 Prova de que a primitiva de transformação *TED#1* preserva a informação.

Antes de provarmos que **TED#1** preserva a informação e a atualização das informações, enunciamos dois lemas que são usados nessa prova. Nessa seção, considere $S=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ um esquema.

<u>Suposição de Entidades Idênticas</u> (SEI): Suponha que A e B sejam classes em $\mathcal{C}$ onde $\{fk_1, ...fk_m\} \in$ ***chaves***(A) e $\{gk_1, ...,gk_m\} \in$ ***chaves***(B). Suponha, também, que $\mathcal{D}$ seja uma instância de esquema S onde $a \in \mathcal{D}(A)$ e $b \in \mathcal{D}(B)$. Se $\mathcal{D}(fk_i)(a) = \mathcal{D}(gk_i)(b)$, para $1 \leq i \leq m$, então $a=b$.

<u>Lema B.5</u>: Suponha que A e B sejam classes em $\mathcal{C}$ onde $\{fk_1, ...fk_m\} \in$ ***chaves***(A) e $\{gk_1, ...,gk_m\} \in$ ***chaves***(B). Para qualquer instância $\mathcal{D}$ de S, $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ é válida em $\mathcal{D}$ sss ($A \subset B$ *e* $fk_i \equiv gk_i$ para $1 \leq i \leq n$) for válida em $\mathcal{D}$.

<u>Prova</u>:

(1) Suponha que exista uma instância $\mathcal{D}$ de S onde $A[fk_1, ...,fk_m] \subset B[fk_1, ...,fk_m]$ seja válida em $\mathcal{D}$. Seja $a \in \mathcal{D}(A)$. Como $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ é válida em $\mathcal{D}$, existe $b \in \mathcal{D}(B)$ tal que $\mathcal{D}(fk_i)(a) = \mathcal{D}(gk_i)(b)$, para $1 \leq i \leq m$. Pela SEI, temos que $a=b$. Conseqüentemente, $a \in \mathcal{D}(B)$ e assim, se $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ for válida em $\mathcal{D}$, então $A \subset B$ é válida em $\mathcal{D}$.

(2) Suponha que exista uma instância $\mathcal{D}$ de S onde $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ seja válida em $\mathcal{D}$. Assim, existe $a \in \mathcal{D}(A)$ tal que ($a \in \mathcal{D}(B)$ e existe i $\in$ [1,m] tal que $\mathcal{D}(fk_i)(a) \neq \mathcal{D}(gk_i)(a)$).

Como $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ é válida em $\mathcal{D}$, existe $b \in \mathcal{D}(B)$ tal que $\mathcal{D}(fk_i)(a) = \mathcal{D}(gk_i)(b)$, para $1 \leq i \leq m$. Por SEI temos que $a=b$. Conseqüentemente, $\mathcal{D}(fk_i)(a) = \mathcal{D}(gk_i)(a)$, para $1 \leq i \leq m$, uma contradição. Assim, se $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ for válida em $\mathcal{D}$, então ($fk_i \equiv gk_i$ para $1 \leq i \leq m$) é válida em $\mathcal{D}$.

(3) Suponha que exista uma instância $\mathcal{D}$ de S onde ($A \subset B$ e $fk_i \equiv gk_i$ para $1 \leq i \leq n$) seja válida em $\mathcal{D}$. Assim, existe $a \in \mathcal{D}(A)$ tal que $\mathcal{D}(fk_i)(a) = \mathcal{D}(gk_i)(a)$, para $1 \leq i \leq m$, e $a \in \mathcal{D}(B)$. Logo, $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$. Conseqüentemente, se ($A \subset B$ e $fk_i \equiv gk_i$ para $1 \leq i \leq n$) for válida em $\mathcal{D}$, então $A[fk_1, ...,fk_m] \subset B[gk_1, ...,gk_m]$ é válida em $\mathcal{D}$.

<u>Lema B.6</u>: Suponha que A e B sejam classes em $\mathcal{C}$ onde$\{fk_1, ...,fk_m\} \in$ ***chaves***(A), $\{gk_1, ...,gk_m\} \in$ ***chaves***(B) e $fk_i \equiv gk_i$ para $1 \leq i \leq m$. Suponha que $f \in$ ***rel***(A) e $g \in$ ***rel***(B). Para qualquer instância $\mathcal{D}$ de S, $<A([fk_1, ...,fk_m] \rightarrow f), B([gk_1, ...,gk_m] \rightarrow g)>$ é válida em $\mathcal{D}$ sss $f \equiv g$ for válida em $\mathcal{D}$.

Prova:

(1) Suponha que exista uma instância $\mathscr{D}$ de $\boldsymbol{S}$ onde $\langle A([fk_1, \ldots, fk_m] \rightarrow f), B([gk_1, \ldots, gk_m] \rightarrow g)$ seja válida em $\mathscr{D}$ e onde $f \equiv g$ não seja válida em $\mathscr{D}$. Assim, existe $a \in \mathscr{D}(A)$ tal que $a \in \mathscr{D}(B)$ e $\mathscr{D}(f)(a) \neq \mathscr{D}(g)(a)$. Mas $\mathscr{D}(fk_i)(a) = \mathscr{D}(gk_i)(a)$ para $1 \leq i \leq m$. Conseqüentemente, pela DFU, $\mathscr{D}(f)(a) = \mathscr{D}(g)(a)$, uma contradição. Assim, se $\langle A([fk_1, \ldots, fk_m] \rightarrow f), B([gk_1, \ldots, gk_m] \rightarrow g)\rangle$ for válida em $\mathscr{D}$, então $f \equiv g$ é válida em $\mathscr{D}$.

(2) Suponha que exista uma instância $\mathscr{D}$ de $\boldsymbol{S}$ onde $f \equiv g$ seja válida em $\mathscr{D}$ e onde $\langle A([fk_1, \ldots, fk_m] \rightarrow f), B([gk_1, \ldots, gk_m] \rightarrow g)\rangle$ não seja válida em $\mathscr{D}$. Assim, existem $a \in \mathscr{D}(A)$ e $b \in \mathscr{D}(B)$ tais que $\mathscr{D}(fk_i)(a) = \mathscr{D}(gk_i)(b)$, para $1 \leq i \leq m$, and $\mathscr{D}(f)(a) \neq \mathscr{D}(g)(b)$. Pela SEI, temos que $a = b$. Conseqüentemente, $\mathscr{D}(f)(a) = \mathscr{D}(g)(b)$, uma contradição. Assim, se $f \equiv g$ for válida em $\mathscr{D}$, então $\langle A([fk_1, \ldots, fk_m] \rightarrow f), B([gk_1, \ldots, gk_m] \rightarrow g)\rangle$ é válida em $\mathscr{D}$.

Teorema B.3: ***TED#1*** preserva a informação.

Prova: Como ***TED#1*** modifica apenas as restrições do esquema, pelos lemas B.5 e B.6, as novas restrições são equivalents às antigas.

B.4 Prova de que a primitiva de transformação *TED#1* preserva a semântica das atualizações.

No restante dessa seção, considere $S_1=(\mathcal{C}, \mathcal{A}, \mathcal{R})$ um esquema, onde:

- $A[f_1,...,f_n] \subset B[g_1,...,g_n] \in \mathcal{R}$;
- $\{f_1, ..., f_n\} \in$ *S_chaves*(A);
- $\{fk_1, ...,fk_m\} \in$ *S_chaves*(A), onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ...,f_n\}$, sss $\{gk_1, ...,gk_m\} \in$ *S_chaves*(B); e
- Se $\{fk_1, ...,fk_m\} \in$ *S_chaves*(A), onde $\{fk_1, ...,fk_m\} \subseteq \{f_1, ...,f_n\}$, então $\mathcal{R} \models \langle A([fk_1, ...,fk_m] \rightarrow f_j),\ B([gk_1, ...,gk_m] \rightarrow g_j)\rangle$, para $1 \leq j \leq n$.

Seja $S_2=(\mathcal{C}', \mathcal{A}', \mathcal{R}')$ tal que $S_2=$**TED#1**$(A[f_1,...,f_n] \subset B[g_1,...,g_n])(S_1)$. Conseqüente-mente, $A \subset B \in \mathcal{R}'$ e $f_i \equiv g_i \in \mathcal{R}'$, para $1 \leq i \leq n$.

Nós provamos o teorema, mostrando que a semântica de qualquer atualização u em *atualizações*(S_1) é preservada em S_2. Para cada atualização u temos que considerar todos os estados distintos nos quais u tem tipos distintos de efeitos colaterais.

1- Seja u = "adicione a a A" (do esquema S_1). Consideremos $\mathcal{D}_1 \in$ *instâncias*(S_1) e $\mathcal{D}_2 \in$ *instâncias*(S_1) onde $\mathcal{D}_2 = \phi(\mathcal{D}_1)$. Como $\psi = A[f_1,...,f_n] \subset B[g_1,...,g_n]$ é a única restrição relevante em S_1 que não é diretamente transformada em uma restrição no esquema transformado, nós só precisamos considerar ψ.

<u>Caso A</u>. Suponha que exista $b \in \mathcal{D}(B)$ tal que $\mathcal{D}(fk_i)(a) = \mathcal{D}(gk_i)(b)$, para $1 \leq i \leq m$.

1. Cálculo de $u(\mathcal{D}_1)$

Seja $\mathcal{D}_1{}^1=$ *efeito*$(u, \mathcal{D}_1)$. $\mathcal{D}_1{}^1$ é idêntico a $\mathcal{D}_1$ exceto que:

(A.1) $\mathcal{D}_1{}^1(A)=\mathcal{D}_1{}^1(A) \cup \{a\}$

Como *relevantes*$(u)=\{\psi\}$, temos que:

*co-efeito**$(u,\ \mathcal{D}_1{}^1) =$ *co-efeito*$(u,\ \mathcal{D}_1{}^1, \psi)$.

Como *EC*="continue" é o efeito colateral requerido pela semântica de atualização de u com relação a ψ (ψ não é violada por u em $\mathcal{D}_1$), temos que:

co-efeito$(u,\ \mathcal{D}_1{}^1, \psi) = \mathcal{D}_1{}^1$.

Logo,

*co-efeito**$(u,\ \mathcal{D}_1{}^1) = \mathcal{D}_1{}^1$.

Conseqüentemente,

(A.2) $u(\mathcal{D}_1)=$*co-efeito**$(u,$ *efeito*$(u, \mathcal{D}_1)) = \mathcal{D}_1{}^1$.

2. Cálculo de $\tau(u)(\mathcal{D}_2)$

Seja $u_\tau = \tau(u)$ = "remova a de A " (do esquema S_2).

Pelo Lema B.5, $a \in \mathcal{D}(B)$.

Seja $\mathcal{B}_2{}^1$= *efeito*(u_τ, $\mathcal{B}_2$). $\mathcal{B}_2{}^1$ é idêntico a $\mathcal{B}_2$ exceto que:

(A.3) $\mathcal{B}_2{}^1(A)=\mathcal{B}_2(A) \cup \{a\}$

Como *relevantes*(u_τ)={$\psi = A \subset B$ }, temos que:

*co-efeito**(u_τ, $\mathcal{B}_2$) = *co-efeito*(u_τ, $\mathcal{B}_2{}^1$, ψ).

Como *EC*="continue" é o efeito colateral requerido pela semântica de atualização de u_τ com relação a ψ (ψ não é violada por u em $\mathcal{B}_2{}^1$), temos que:

co-efeito(u_τ, $\mathcal{B}_2{}^1$, ψ) = $\mathcal{B}_2{}^1$.

Logo,

*co-efeito**(u, $\mathcal{B}_2{}^1$) = $\mathcal{B}_2{}^1$

Conseqüentemente,

(A.4) $u_\tau(\mathcal{B}_2)$=*co-efeito**(u_τ, *efeito*(u_τ, $\mathcal{B}_2$)) = $\mathcal{B}_2{}^1$.

Como $\sigma(\mathcal{B}_2{}^1)(A) = \mathcal{B}_2{}^1(A)$, por (A.3) temos que:

(A.5) $\sigma(\mathcal{B}_2{}^1)(A) = \mathcal{B}_2(A) \cup \{a\}$,

Mas $\mathcal{B}_2(A)=\mathcal{B}_1(A)$. Assim, por (A.1) e (A.5) temos que:

(A.6) $\sigma(\mathcal{B}_2{}^1)(A) = \mathcal{B}_1{}^1(A)$

Por (A.1), (A.3) e (A.6) temos que:

(A.7) $\mathcal{B}_1{}^1 = \sigma(\mathcal{B}_2{}^1)$.

Por (A.2), (A.4) e (A.7) temos que:

$u(\mathcal{B}_1)=\sigma(u_\tau(\mathcal{B}_2))$.

Conseqüentemente,

$u(\mathcal{B}_1) = \sigma(\tau(u)(\phi(\mathcal{B}_1)))$.

Caso B. Suponha que não exista nenhum $b \in \mathcal{B}(B)$ tal que $\mathcal{B}(fk_i)(a) = \mathcal{B}(gk_i)(b)$, para $1 \leq i \leq m$. Nós temos que considerar todas as alternativas possíveis de semântica de atualização para $\psi = A[f_1,...,f_n] \subset B[g_1,...,g_n]$.

Caso B1: Suponha que $SI(A,\psi)$=*rejeita*,

1. Cálculo de $u(\mathcal{B}_1)$

Seja $\mathcal{B}_1{}^1$= *efeito*$(u, \mathcal{B}_1)$. $\mathcal{B}_1{}^1$ é idêntico a $\mathcal{B}_1$ exceto que:

(B1.1) $\mathcal{B}_1{}^1(A)=\mathcal{B}_1{}^1(A) \cup \{a\}$

Como *relevantes*$(u)=\{\psi\}$, temos que:

*co-efeito**$(u, \mathcal{B}_1{}^1)$ = *co-efeito*$(u, \mathcal{B}_1{}^1, \psi)$.

Como *EC*="rollback" é o efeito colateral requerido pela semântica de atualização de u com relação a ψ temos que:

co-efeito$(u, \mathcal{B}_1, \psi)$ = "aborte a presente transação"

Conseqüentemente,

(B1.2) $u(\mathcal{B}_1)=\mathcal{B}_1$.

2. Cálculo de $\tau(u)(\mathcal{B}_2)$.

Seja $u_\tau = \tau(u)$ = "remova a de A" (do esquema S_2).

Pelo Lema B.5, $a \notin \mathcal{B}(B)$.

Seja $\mathcal{B}_2{}^1$= *efeito*$(u_\tau, \mathcal{B}_2)$. $\mathcal{B}_2{}^1$ é idêntico a $\mathcal{B}_2$ exceto que:

(B1.3) $\mathcal{B}_2{}^1(A)=\mathcal{B}_2(A) \cup \{a\}$

Como *relevantes*$(u_\tau)=\{\psi = A \subset B\}$, temos que:

*co-efeito**$(u_\tau, \mathcal{B}_2{}^1)$ = *co-efeito*$(u_\tau, \mathcal{B}_2{}^1, \psi)$.

Como *EC*="rollback" é o efeito colateral requerido pela semântica de atualização de u_τ com relação a ψ ($SI(A, \psi)$=*rejeita*), temos que:

co-efeito$(u_\tau, \mathcal{B}_2{}^1, \psi)$ = "aborte a presente transação" .

Conseqüentemente,

(B1.4) $u_\tau(\mathcal{B}_2)=\mathcal{B}_2$.

mas $\mathcal{B}_1 = \sigma(\mathcal{B}_2)$.

Por (B1.2) e (B1.4) temos que:

$u(\mathcal{B}_1)=\sigma(\tau(u)(\mathcal{B}_2))$.

Conseqüentemente,

$u(\mathcal{B}_1) = \sigma(\tau(u)(\phi(\mathcal{B}_1)))$.

Caso B2: Suponha que $SI(A,\psi)$=*propaga*.

1. Cálculo de $u(\mathcal{B}_1)$

Seja $\mathcal{B}_1{}^1$= *efeito*$(u, \mathcal{B}_1)$. $\mathcal{B}_1{}^1$ é idêntico a $\mathcal{B}_1$ exceto que:

(B2.1) $\mathcal{B}_1{}^1(A)=\mathcal{B}_1{}^1(A) \cup \{a\}$

Como *relevantes*$(u)=\{\psi\}$, temos que:

*co-efeito**$(u, \mathcal{B}_1{}^1)$ = *co-efeito*$(u, \mathcal{B}_1{}^1, \psi_4)$.

$EC=\{u^1=$ "adicione $b(g_1=\mathcal{B}_1{}^1(f_1)(a), ..., g_n=\mathcal{B}_1{}^1(f_n)(a))$ a B "} é o efeito colateral requerido pela semântica de atualização de u com relação a ψ. Como discutimos na Seção 2.3, já que $b=a$ (pela SEI), a especificação_de_entidade $b(g_1=\mathcal{B}(f_1)(a), ..., g_n=\mathcal{B}(f_n)(a))$ é substituida por a, de forma que u^1= "adicione a a B ". Conseqüente-mente, temos que:

co-efeito$(u, \mathcal{B}_1{}^1, \psi) = u^1(\mathcal{B}_1{}^1)$

1.2 Cálculo de $u^1(\mathcal{B}_1{}^1)$.

Seja $\mathcal{B}_1{}^2$= *efeito*$(u^1, \mathcal{B}_1{}^1)$. $\mathcal{B}_1{}^2$ é idêntico a $\mathcal{B}_1{}^1$ exceto que:

(B2.2) $\mathcal{B}_1{}^2(B)=\mathcal{B}_1{}^1(B) \cup \{a\}$

Como *relevantes*$(u^1)=\varnothing$, temos que:

*co-efeito**$(u^1, \mathcal{B}_1{}^2) = \mathcal{B}_1{}^2$.

Conseqüentemente,

$u^1(\mathcal{B}_1{}^1)$= *co-efeito**$(u^1$, *efeito*$(u^1, \mathcal{B}_1{}^1)) = \mathcal{B}_1{}^2$.

(Continuação do cálculo de $u(\mathcal{B}_1)$

*co-efeito**$(u, \mathcal{B}_1)$ = *co-efeito*$(u, \mathcal{B}_1{}^1, \psi_4) = u^1(\mathcal{B}_1{}^1) = \mathcal{B}_1{}^2$

Conseqüentemente,

(B2.3) $u(\mathcal{B}_1)$=*co-efeito**$(u$, *efeito*$(u, \mathcal{B}_1)) = \mathcal{B}_1{}^2$.

Por (B2.1) e (B2.2), temos que $\mathcal{B}_1{}^2$ é idêntico a $\mathcal{B}_1$ exceto que:

(B2.4) $\mathcal{B}_1{}^2(A)=\mathcal{B}_1(A) \cup \{a\}$

$\mathcal{B}_1{}^2(B)=\mathcal{B}_1(B) \cup \{a\}$

2. Cálculo de $\tau(u)(\mathcal{B}_2)$

Seja $u_\tau = \tau(u)$ = "adicione a a A" (do esquema S_2).

Pelo Lema B.5, $a \notin \mathcal{B}(B)$.

Seja $\mathcal{B}_2{}^1$= *efeito*$(u_\tau, \mathcal{B}_2)$. $\mathcal{B}_2{}^1$ é idêntico a $\mathcal{B}_2$ exceto que:

(B2.5) $\mathcal{B}_2{}^1(A)=\mathcal{B}_2(A)\cup\{a\}$

Como *relevantes*$(u_\tau)=\{\psi = A \subset B\}$, temos que:

*co-efeito**$(u_\tau, \mathcal{B}_2{}^1)$ = *co-efeito*$(u_\tau, \mathcal{B}_2{}^1, \psi)$.

Como $EC=\{u^1=$ "adicione a a B "} é o efeito colateral requerido pela semântica de atualização de u com relação a ψ, temos que:

$co\text{-}efeito(u, \mathcal{B}_2^1, \psi) = u^1(\mathcal{B}_2^1)$

1.2 Cálculo de $u^1(\mathcal{B}_2^1)$.

Seja $\mathcal{B}_2^2 = efeito(u^1, \mathcal{B}_2^1)$. $\mathcal{B}_2^2$ é idêntico a $\mathcal{B}_2^1$ exceto que:

(B2.6) $\mathcal{B}_2^2(B) = \mathcal{B}_2^1(B) \cup \{a\}$

Como $relevantes(u^1) = \emptyset$, temos que:

$co\text{-}efeito^*(u^1, \mathcal{B}_2^2) = \mathcal{B}_2^2$.

Conseqüentemente,

$u^1(\mathcal{B}_2^1) = co\text{-}efeito^*(u^1, efeito(u^1, \mathcal{B}_2^1)) = \mathcal{B}_2^2$.

(Continuação do cálculo de $u_\tau(\mathcal{B}_2)$)

$co\text{-}efeito^*(u_\tau, \mathcal{B}_2) = co\text{-}efeito(u_\tau, \mathcal{B}_2^1, \psi) = u^1(\mathcal{B}_2^1) = \mathcal{B}_2^2$

Conseqüentemente,

(B2.7) $u_\tau(\mathcal{B}_2) = co\text{-}efeito^*(u_\tau, efeito(u_\tau, \mathcal{B}_2)) = \mathcal{B}_2^2$.

Por (B2.5) e (B2.6), temos que $\mathcal{B}_2^2$ é idêntico a $\mathcal{B}_2$ exceto que:

(B2.9) $\mathcal{B}_2^2(A) = \mathcal{B}_2(A) \cup \{a\}$

$\mathcal{B}_2^2(B) = \mathcal{B}_2(B) \cup \{a\}$

Como $\sigma(\mathcal{B}_2^2)(A) = \mathcal{B}_2^2(A)$, e $\sigma(\mathcal{B}_2^2)(B) = \mathcal{B}_2^2(B)$, por (B2.9) temos que:

(B2.10) $\sigma(\mathcal{B}_2^2)(A) = \mathcal{B}_2(A) \cup \{a\}$,

$\sigma(\mathcal{B}_2^2)(B) = \mathcal{B}_2(B) \cup \{a\}$.

Mas $\mathcal{B}_2(A) = \mathcal{B}_1(A)$ e $\mathcal{B}_2(B) = \mathcal{B}_1(B)$. Assim, por (B2.4) e (B2.10) temos que:

(B2.11) $\sigma(\mathcal{B}_2^2)(A) = \mathcal{B}_1^2(A)$

$\sigma(\mathcal{B}_2^2)(B) = \mathcal{B}_1^2(B)$

Por (B2.4), (B2.9) e (B2.11) temos que:

(B2.12) $\mathcal{B}_1^2 = \sigma(\mathcal{B}_2^2)$.

Por (B2.3), (B2.7) e (B2.12) temos que:

$u(\mathcal{B}_1) = \sigma(u_\tau(\mathcal{B}_2))$.

Conseqüentemente,

$u(\mathcal{B}_1) = \sigma(\tau(u)(\phi(\mathcal{B}_1)))$.

As provas para os outros tipos de atualizações são similares e serão omitidas aquí.