Universidade Federal da Paraíba

Centro de Ciências e Tecnologia

Departamento de Sistemas e Computação

DISSERTAÇÃO DE MESTRADO

UM PROTOCOLO DE CONTROLE DE CONCORRÊNCIA NÃO RESTRITIVO PARA BANCOS DE DADOS COOPERATIVOS

por

Almir Pereira Guimarães

Campina Grande, dezembro de 1997.

Universidade Federal da Paraíba

Centro de Ciências e Tecnologia

Departamento de Sistemas e Computação

Almir Pereira Guimarães

UM PROTOCOLO DE CONTROLE DE CONCORRÊNCIA NÃO RESTRITIVO PARA BANCOS DE DADOS COOPERATIVOS

Este trabalho foi apresentado à Pós-Graduação em Informática do Centro de Ciências e Tecnologia da Universidade Federal da Paraíba como requisito parcial para obtenção do grau de Mestre em Informática.

ORIENTADOR : Marcus Costa Sampaio

ÁREA DE CONCENTRAÇÃO : Ciência da Computação

SUB-ÁREA : Sistemas de Informação e Banco de Dados

Campina Grande, dezembro de 1997.

G963p Guimarães, Almir Pereira.
Um protocolo de controle de concorrência não restritivo para bancos de dados cooperativos / Almir Pereira Guimarães. - Campina Grande, 1997.
87 f.

Dissertação (Mestrado em Informática) - Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, 1997.
"Orientação : Prof. Dr. Marcus Costa Sampaio".
Referências.

1. Banco de Dados. 2. Protocolo de Controle. 3. Banco de Dados Cooperativos. 4. Dissertação - Informática. I. Sampaio, Marcus Costa. II. Universidade Federal da Paraíba - Campina Grande (PB). III. Título

CDU 004.65(043)

PROTOCOLO DE CONTROLE DE CONCORRÊNCIA NÃO-RESTRITIVO EM BANCOS DE DADOS COOPERATIVO

ALMIR PEREIRA GUIMARÃES

DISSERTAÇÃO APROVADA EM 22.12.1997

PROF. MARCUS COSTA SAMPAIO, Dr.
Presidente

PROF. LUIS CLÁUDIUS CORADINE , Dr.
Examinador

PROF. MAURILICIO MARTINIANO DOS SANTOS, M.Sc
Examinador

CAMPINA GRANDE - PB

A todas as pessoas que direta ou indiretamente contribuiram para a concretização deste trabalho.

Agradecimentos

Meus agradecimentos especiais ao Prof. Marcus Sampaio, pela sua amizade, apoio nas horas fundamentais e dedicação. Foi uma experiência gratificante ter tido a oportunidade de trabalhar sob a sua orientação.

Aos examinadores prof. Maurilúcio Martiniano dos Santos e prof. Luis Cláudio Coradine, meus agradecimentos pelas valiosas sugestões sobre o conteúdo e a forma desta dissertação.

A meu grande amigo Gustavo, fantástico, pelo seu apoio e presteza nas horas decisivas em que mais precisei.

Agradeço aos meus amigos do mestrado, em especial ao Genebaldo, pelos bons momentos que desfrutei nas conversas, nas brincadeiras, nos estudos, aonde juntos estamos conseguindo alcançar nossos objetivos.

Aos Profs. Evandro Barros e Marcus Braga do Departamento de Matemática Aplicada, MAP, da Universidade Federal de Alagoas pela colaboração prestada a esta tese.

A sobretudo meus pais, meus irmãos, Mário e André Luiz, Glória e minha querida Tatiana, que sempre estiveram ao meu lado tendo paciência e me dando apoio e conforto nas horas mais importantes.

Por fim, meu grande agradecimento e consideração às pessoas que fundamentalmente contribuiram para este trabalho, acreditanto na concretização do mesmo nas horas em que duvidei disto.

RESUMO

Para aplicações cooperativas, as propriedades ACID (Atomicidade, Consistência, Isolação, Durabilidade), apresentadas pelas transações que estão segundo o modelo convencional de transações, mostram-se muito restritivas para dar suporte às características apresentadas por estas aplicações como por exemplo, a grande interatividade, longo tempo de duração etc. Para dar suporte a estas características, tem surgido extensões do modelo convencional de transações como os diferentes modelos de transações cooperativas.

Esta dissertação está particularmente baseada em um modelo de transações cooperativas denominado de Modelo de Unidades Cooperativas, MUC, cuja originalidade repousa para assegurar a preservação da consistência do banco de dados, o controle de concorrência e a tolerância a falhas sob o conhecimento prévio das relações entre objetos e sua exploração. Sendo ainda apresentados os principais protocolos de controle de concorrência existentes para transações não-convencionais que deram suporte à elaboração de um protocolo de controle de concorrência para transações segundo o Modelo de Unidades Cooperativas que foi o principal objetivo desta dissertação.

Este protocolo, por sua vez, esta dividido em duas "grande" regras: uma para definir como adquirir um bloqueio em uma hierarquia "IS-A" ou É-PARTE-DE e a outra para definir herança e transferência de bloqueios entre transações no interior de uma unidade cooperativa.

PALAVRAS CHAVES: Transações Cooperativas, Controle de Concorrência, Granulagem de Bloqueios, Unidades Cooperativas, Transações Não-Convencionais, Objetos Complexos, Hierarquias, Validação Seletiva, Bloqueios Implícitos, Transações Hierárquicas.

ABSTRACT

With respect to the cooperative applications, the ACID properties (Atomicity, Consistency, Isolation, Durability) seems very restrictive, because this kind of applications have characteristics as a great interactivity, a long time alive etc. To give effort to these characteristics, has emerged extensions of conventional transaction model, as the different cooperative transaction model.

This dissertation particularly is based in a cooperative transaction model, called "Cooperative Unit Model", MUC. Also presents the more expressive concurrency control protocols for non-conventional transactions which give effort to elaboration of a concurrency control protocol for transactions under "the cooperative unit model".

This protocol, is divided in two great rules: one to gain a lock in a IS-A or IS-PART-OF hierarchy and other to define lock inheritance and transference between transactions in a cooperative unit.

KEY WORDS : Cooperative Transactions, Concurrency Control, Lock Granularity, Cooperative Unit, Non-Conventional Transactions, Complex-Objects, Implicit Locks.

ÍNDICE

Índice

LISTA DE FIGURAS

1. Introdução

1.1 MOTIVAÇÃO

O modelo clássico de transações ACID (Atomicidade, Consistência, Isolação, Durabilidade) [Bernstein-87] não é adequado para o caso de aplicações não-convencionais. Neste trabalho, trataremos de um tipo de aplicação não-convencional chamado de "Workgroup" ou "Groupware", que traduziremos por *aplicação cooperativa*. Numa aplicação cooperativa, as transações apresentam características bem distintas em relação às transações convencionais, como por exemplo, uma grande interatividade (usuário-sistema, usuário-usuário) e longo tempo de duração.

Transações cooperativas ocorrem em ambientes que envolvem geralmente equipes de projetistas trabalhando cooperativamente em estações de trabalho. Nelas os membros de uma equipe são representados por transações, que interagem entre si compartilhando dados situados em bancos de dados cooperativos.

Devido às características de transações cooperativas, as propriedades clássicas ACID deverão ser flexibilizadas com o objetivo de dar suporte à execução dessas transações. Em outras palavras, são necessários novos modelos de transações para atender às características de transações cooperativas.

A elaboração de novos modelos de transações irá levar tanto à necessidade de novos protocolos de controle de concorrência de transações, quanto à necessidade de novos protocolos de recuperação de falhas de transações.

1.2 OBJETIVO DA DISSERTAÇÃO

O nosso estudo está baseado em um novo modelo de transações cooperativas chamado de *Modelo de Unidades Cooperativas* [Sampaio-95]. O objetivo principal do trabalho é o de propor um novo protocolo de controle de concorrência de transações participando de uma unidade cooperativa, e de transações que não participam da unidade cooperativa.

1.3 ORGANIZAÇÃO DA DISSERTAÇÃO

Além deste capítulo introdutório, temos o segundo capítulo onde é tratado o Modelo de Unidades Cooperativas, proposto em [Sampaio-95], cujo conceito de cooperação é totalmente independente do domínio de uma aplicação, explorando unicamente as relações de dependência entre objetos de um banco de dados.

As propriedades das transações cooperativas deste modelo são apresentadas neste capítulo. Além disso, outros conceitos do modelo serão também apresentados, como por exemplo, o conceito de unidade cooperativa "fraca" ou "forte", conforme o grau de cooperação de transações em uma unidade cooperativa.

No terceiro capítulo tratamos de protocolos de Controle de Concorrência (CC) existentes na literatura para transações não-convencionais. Neste capítulo, são mostrados os principais protocolos de CC encontrados na literatura para transações não-convencionais, objetivando mostrar as diferenças entre as abordagens existentes e, sobretudo, ressaltar que nenhum desses protocolos atendem integralmente às exigências do modelo de unidades cooperativas. No entanto, diversos conceitos empregados nesses protocolos inspiraram o nosso protocolo de CC.

Na primeira parte do capítulo é mostrado um protocolo baseado em granulagem de bloqueios, sobre uma hierarquia de recursos de um sistema de banco de dados (Protocolo de Gray), podendo o protocolo ser estendido para o caso de grafos [Gray-78]. A

importância do mesmo reside no fato de que seu conceito de granulagem de bloqueios tem inspirado vários outros protocolos de CC, incluindo o nosso.

A seguir, são discutidos o protocolo de CC apresentado em [Garza-Kim-88], assim como uma nova versão do protocolo de Gray, protocolo II de Gray, proposto em [Gray-81], sobre hierarquias de objetos em um modelo orientado a objetos. Embora nestes protocolos as transações ainda sejam isoladas umas das outras (transações não-cooperativas), seu emprego em hierarquias de objetos nos foi de muita valia para a elaboração do nosso protocolo de CC.

Por fim, apresentamos um protocolo de CC para transações hierárquicas [Moss-85]. Uma transação hierárquica é decomposta em subtransações que, por sua vez, são também decompostas em subtransações, e assim por diante. As idéias de herança de bloqueios adotadas por este protocolo nos foram interessantes porque as unidades cooperativas são também hierárquicas.

No quarto capítulo, finalmente tratamos do protocolo de CC de transações cooperativas segundo o Modelo de Unidades Cooperativas. Este é o capítulo central da dissertação, onde é proposto um novo protocolo de CC para sincronizar as atividades de transações cooperativas nos diferentes níveis de uma unidade cooperativa hierárquica, e também para a sincronização entre unidades cooperativas e transações que não lhes pertencem.

O protocolo utiliza diferentes granulagens de bloqueio, hierarquias de objetos, hierarquias de classes e herança de bloqueios.

Por fim, no quinto capítulo apresentamos as conclusões e perspectivas. É apresentado um resumo dos assuntos abordados nesta dissertação, destacando as principais características do protocolo de CC de unidades cooperativas proposto.

A necessidade deste protocolo deveu-se ao fato de que os principais protocolos de CC existentes na literatura para transações não-convencionais, apresentados no terceiro capítulo, não suprem todas as exigências desta classe de transações cooperativas,

sendo necessário então um novo protocolo. Encerramos sugerindo algumas direções de trabalhos futuros.

2. O Modelo de Unidades Cooperativas

Na maior parte dos Sistemas Gerenciadores de Banco de Dados (SGBDs) existentes no mercado, as transações que são suportadas por estes são aquelas que apresentam em suas execuções as propriedades *ACID*, ou seja, as chamadas transações convencionais.

Em relação à propriedade A, de *Atomicidade,* o sistema garante que se algum erro ocorrer durante a execução de uma transação, esta será desfeita e nenhum de seus efeitos permanecerá no banco de dados.

As transações são *Consistentes,* propriedade C, no sentido de que, quando executadas por completo, o SGBD assume que nenhuma restrição de integridade foi violada e que o banco de dados foi levado a um (novo) estado consistente.

Já a propriedade I, de *Isolação*, garante que as transações têm um comportamento semelhante ao que elas teriam caso estivessem executando sozinhas.

Por fim, a propriedade D, de *Durabilidade*, diz respeito ao fato de que os resultados gerados por transações que terminaram corretamente (*commited transactions*) devem ser permanentes no banco de dados mesmo na ocorrência de falhas do sistema.

As propriedades A e D, de *atomicidade* e *durabilidade,* formam a base para a manutenção da integridade e da consistência do banco de dados em caso de falhas. Entretanto, não tratamos de falhas nesse nosso estudo. Dito de outro modo, fazemos a hipótese de que nenhuma falha ocorre durante o processamento das transações.

A grande maioria dos SGBDs convencionais implementam a propriedade I com base na teoria da *serializabilidade* [Bernstein-87]. A idéia chave desta teoria é que se uma execução concorrente de um conjunto de transações for *serializável*, então ela produzirá os mesmos efeitos sobre o banco de dados e sobre as transações que se teria se a execução das transações fosse em série.

No entanto, a propriedade I, assim como também a propriedade C, respectivamente de isolação e de consistência, mostram-se muito restritivas para as aplicações ditas não-convencionais. Como conseqüência, nos últimos tempos, pesquisadores têm buscado extensões do modelo clássico de transações para contemplar as exigências das novas aplicações, como por exemplo aplicações CAD, CASE, automação de escritório, etc. Transações para esses tipos de aplicações podem cooperar em grupo e, suas durações são incertas, geralmente longas (o que poderia gerar longas esperas, se as transações fossem isoladas umas das outras).

Para levar em consideração estas novas exigências, têm surgido novos modelos de dados (ex: *modelo de dados orientado a objetos*), novas arquiteturas de sistemas (ex: *arquitetura cliente-servidor*) e extensões do modelo clássico de transações (ex: *diferentes modelos de transações cooperativas*).

O nosso estudo está baseado particularmente em um modelo de transações cooperativas proposto em [Sampaio-95], chamado de Modelo de Unidades Cooperativas (MUC), cuja originalidade reside em assegurar a preservação da consistência do banco de dados, no conhecimento prévio das relações entre objetos e a sua exploração. Este modelo é uma evolução do modelo clássico de transações, e que junto com seus conceitos fundamentais será visto nas seções subseqüentes deste capítulo.

As unidades cooperativas que serão vistas adiante, dentro do modelo MUC, são uma das funcionalidades do Ambiente Distribuído Integrado Cooperativo (ADIC), um protótipo em desenvolvimento no DSC/UFPB. Outra funcionalidade deste ambiente é o seu modelo orientado a objetos, que se mostra adequado à representação de objetos complexos ou estruturados, muito comuns em ambientes cooperativos. Uma

apresentação sucinta deste modelo se faz necessária para uma melhor compreensão das diferentes granulagens de bloqueio suportadas pelo protocolo proposto.

2.1 O MODELO DE DADOS ORIENTADO A OBJETOS DO ADIC.

Como já citamos, o ADIC é um ambiente distribuído e integrado para aplicações cooperativas suportando um modelo orientado a objetos. Antes de entrarmos no modelo de dados do ADIC, vamos apresentar os principais conceitos de orientação a objeto nos quais se apoiam os modelos orientados a objetos em geral, dentre eles o modelo do ADIC, a fim de uma melhor compreensão deste.

2.1.1 PRINCIPAIS CONCEITOS DE MODELOS DE DADOS ORIENTADOS A OBJETOS

Todos os modelos de dados orientados a objetos se utilizam de pelo menos um conjunto mínimo de conceitos de orientação a objeto que são mostrados abaixo:

Objetos e Identificação de Objeto - Todas as entidades de um sistema são modeladas como objetos, e a todo objeto o sistema atribuirá um identificador que o distinguirá univocamente dentro do mesmo.

Classes - Na maior parte dos sistemas orientados a objetos, classes são consideradas como objetos, exatamente como instâncias de classes são objetos [Kim-90]. Uma classe é definida como um meio de agrupar todos os objetos do sistema que possuem as mesmas características (atributos) e a mesma interface (métodos), sendo que a relação entre um objeto e a classe a qual pertence, é a relação "É-INSTÂNCIA-DE". Na Figura 2.1, Embarcação, Embarcação a Motor e Embarcação à Vela são classes e por isto cada uma agrupa instâncias semelhantes.

Atributos - São características inerentes a um objeto, sendo que o conjunto de atributos define o estado deste objeto no sistema. Como exemplo, ilustra-se na Figura 2.1, onde altura, comprimento e capacidade são atributos da classe Embarcação.

Métodos - Definem o comportamento do objeto dentro do sistema. Os métodos operam em cima do estado (atributos) do objeto. Estes métodos definem a interface da classe com o resto do sistema, sendo que a única forma de se acessar os atributos de um determinado objeto é através de seus correspondentes métodos públicos. Como ilustração, suponha que a classe Embarcação tenha um método público, m1, para modificar o comprimento de uma instância. Desta forma, qualquer modificação do atributo comprimento de uma instância desta classe só será possível através do método m1.

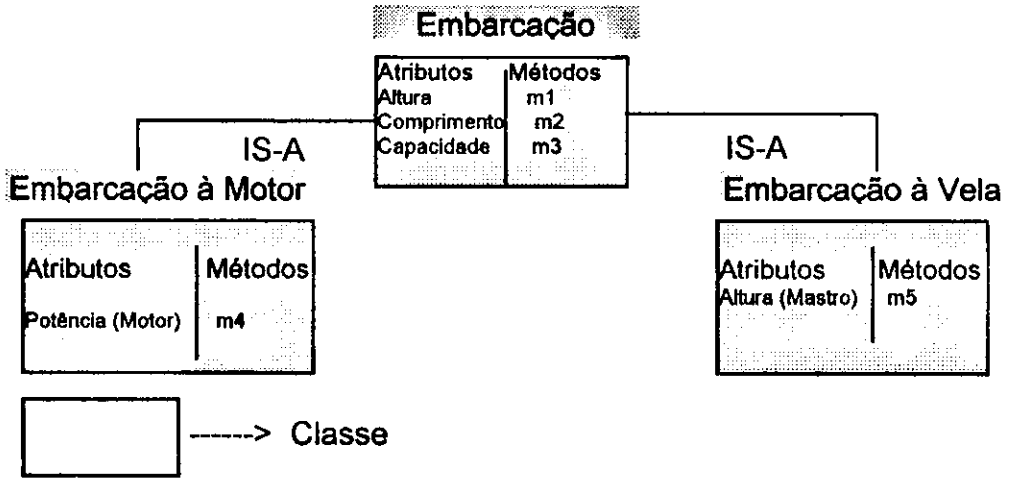


Figura 2.1 - " Hierarquia "IS-A" cuja Classe-Raiz é a Classe Embarcação.

Herança e Hierarquia de Classes - Em um modelo de dados orientado a objetos, uma nova classe poderá ser derivada de alguma outra classe que já existe no sistema. Esta nova classe derivada, que é denominada de subclasse, herda desta classe anteriormente existente, que é denominada de superclasse, todos os seus métodos e atributos, adicionando a estes, novos métodos e atributos que a distinguirá de outras classes do sistema, formando assim, uma hierarquia entre as

classes, segundo uma relação entre elas denominada de "IS-A"[1]. Como exemplo, temos a Figura 2.1 onde as classes Embarcação a Motor e Embarcação à Vela são subclasses da classe Embarcação e por isto, herdam desta todos os seus atributos e métodos, adicionando a estes, os seus próprios métodos e atributos. Nesta hierarquia vemos que as classes Embarcação a Motor e Embarcação por exemplo, mantêm entre si a relação "IS-A". A seguir, falaremos finalmente sobre o modelo de dados do ADIC.

Projetos de engenharia, de arquitetura etc, são hierárquicos, isto é, um projeto é decomposto em subprojetos recursivamente até que um subprojeto não pode mais ser decomposto. De maneira semelhante, objetos são hierárquicos ou complexos, isto é, um objeto é constituído de outros objetos recursivamente até que um subobjeto é primitivo ou atômico. Cada projeto tem uma equipe de projetistas, arquitetos, engenheiros etc, que manipulam objetos complexos como plantas de engenharia, arquitetura, necessitando consultá-los e atualizá-los.

As classes de objetos no modelo de dados do ADIC são organizadas em uma hierarquia "IS-A". Se o domínio de um atributo de uma classe A é uma outra classe B, então existe a relação B É-DOMÍNIO-DE (um atributo de) A. As classes às quais os objetos componentes de um objeto complexo pertencem são organizadas em uma hierarquia É-PARTE-DE. Dado que a relação B É-PARTE-DE A é verdadeira, então tanto B É-DOMÍNIO-DE A e B É-EXCLUSIVAMENTE-DEPENDENTE-DE A (isto é, B existe se e apenas se A existe) são verdadeiras. Tanto a relação "IS-A" quanto a relação É-PARTE-DE são binárias e não-simétricas.

Domínios de objetos incluem textos, gráficos, imagens e outros dados multimídia. Objetos multimídia são criados e manipulados por software específicos como PowerPoint, CorelDraw ou AutoCad. No modelo de dados, esses objetos multimídia (arquivos) são tratados como dados do tipo *external_file*. Suponha o atributo *conteúdo* de um objeto da classe Fachada com o domínio *external_file*: o seu valor é

[1] Preferimos manter o nome da relação em inglês porque em português o artigo indefinido tem gênero. Assim desejamos evitar dizer "É-UM", "É-UMA".

o *path* do sistema operacional c:\user\.....\fachada.ppt. Sempre que um método "lê" este valor, ele na realidade chama o utilitário Viewer do software PowerPoint que exibe o conteúdo do objeto em uma janela. Por outro lado, sempre que um método "atualiza" este valor , ele na realidade chama o editor gráfico do PowerPoint que interage com o usuário com o objetivo de editar o conteúdo do objeto.

O ADIC não implementa diretamente outra importante relação inter-objetos em ambientes cooperativos, denominada COMPLEMENTA. Duas classes A e B que estão no mesmo nível de uma hierarquia É-PARTE-DE são complementares (em outras palavras, A COMPLEMENTA B e B COMPLEMENTA A) se a modificação em um objeto de A pode ter conseqüência no objeto de B; relações COMPLEMENTA são binárias e simétricas. Em compensação, o ADIC permite que objetos complementares sejam monitorados por meio de regras ativas [Sampaio-97].

O modelo de dados do ADIC são os tipos definidos em C++ com duas extensões: o domínio *external_file* e a associação entre classes É-EXCLUSIVAMENTE-DEPENDENTE-DE (ou simplesmente DEPENDENTE-DE). A Figura 2.2 é um exemplo de um esquema no modelo de dados do ADIC.

```
class Pessoa {                          class Planta dependent-of Projeto
public :                                public :
Pessoa (Nome&, ...);                    Fachada fachada ();
Nome nome ();                           Interior interior ();
. . .};                                 . .. . .};

class Cliente: public Pessoa{           class Fachada dependent-of Planta
public :                                public :
Set <Projeto> projetos ();              void mostre-conteúdo ();
. . .};                                 void edite-conteúdo ();
                                        . . . .};

class Projeto {                         class Interior dependent-of Planta
public:                                 public:
int identificação ();                   void mostre-conteúdo ();
Planta planta ();                       void edite-conteúdo ();
. . . };                                . . . .};
```

Figura 2.2 Esquema de um Banco de Dados ADIC

Os objetos são persistentes, distribuídos e integrados (isto é, a distribuição é transparente para os usuários). Uma declaração de consulta a um esquema de objetos é uma extensão da linguagem de consulta SQL, definida como segue:

```
Set <Tipo><resultado> = SELECT <objetos>
                        FROM  <faixa de variáveis> IN <classe>
                        WHERE <predicado>
```

A consulta, "recupere fachadas dos projetos do cliente X", ilustra o uso de faixa de variáveis, herança de funções-membros, e composição de objetos na linguagem de consulta do ADIC.

```
Set <Fachada> resultado = SELECT f FROM f IN Fachada, c IN Cliente
                          WHERE c.nome () == "X" AND
                           EXISTS (SELECT * FROM pr IN c.projetos ()
                            WHERE pr.planta().fachada() == f ) );
```

2.2 O CONCEITO DE COOPERAÇÃO DO ADIC.

Nos tipos de aplicações que manipulam grandes objetos complexos, é impraticável que apenas uma única transação manipule sozinha todo o objeto. Ela deverá dividir esta tarefa com outras transações que irão manipular, cada uma, um subconjunto de objetos componentes de um objeto complexo. Desta forma, estas transações associadas ao objeto complexo irão constituir uma unidade cooperativa, interagindo entre si, podendo ainda existir subunidades cooperativas a fim de preservar a consistência de todo o objeto complexo (unidade cooperativa hierárquica).

A definição formal de unidades cooperativas será vista ainda neste capítulo na seção 2.3.4. Antes, para exemplificar, considere a Figura 2.3a, onde se tem o objeto complexo Projeto. Este objeto é constituído pelos objetos Contrato e Planta, sendo que este último, por sua vez, é constituído pelos objetos Fachada e Interior.

Associada a este objeto, temos uma transação hierárquica U(U(Ta1,Ta2),Tfa), mostrada na Figura 2.3b, constituindo uma unidade cooperativa hierárquica. Esta transação é constituída pelas subtransações Tfa, que representa o usuário U3 que é um funcionário administrativo de um escritório de arquitetura e U(Ta1,Ta2), sendo que esta última constitui uma subunidade cooperativa dentro da unidade cooperativa hierárquica U(U(Ta1,Ta2),Tfa). A subunidade cooperativa U(Ta1,Ta2) é constituída pelas subtransações Ta1 e Ta2, que representam os usuários U1 e U2 que são respectivamente, os arquitetos de Fachada e Interior, também de um escritório de arquitetura. Neste exemplo, cada objeto componente é manipulado por apenas uma transação. No entanto, isto não é uma restrição do ADIC.

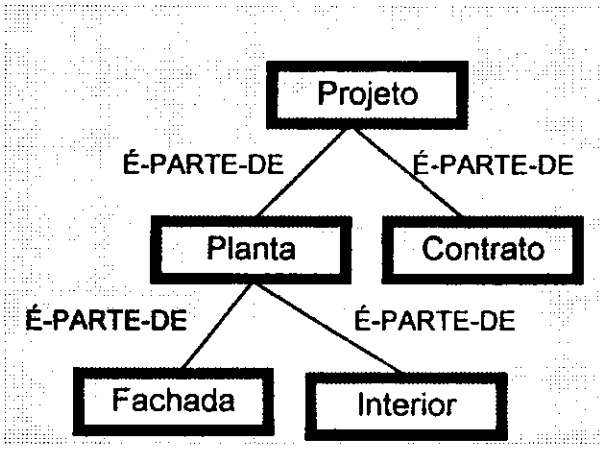


Fig.2.3a Objeto Complexo "Projeto".

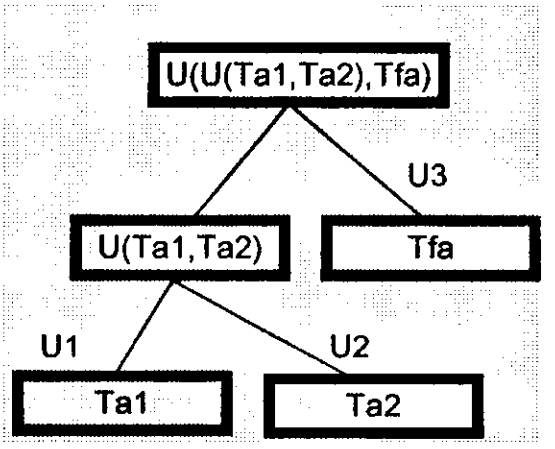


Fig.2.3bTransação Hierárquica associada ao Objeto Complexo

Os objetos Fachada e Interior, que estão no mesmo nível dentro do objeto Projeto, e que foram originados pelo mesmo objeto (Planta), mantêm entre si uma relação de dependência direta (fachada complementa interior, e interior complementa fachada), ou seja, as modificações sobre Fachada irão repercutir diretamente sobre Interior. Por sua vez, as transações Ta1 e Ta2, que manipulam estes objetos, deverão interagir (cooperar) no interior da subunidade cooperativa U(Ta1,Ta2), a fim de preservar a consistência mútua dos objetos.

Por outro lado, as transações Tfa e U(Ta1,Ta2) também deverão cooperar no interior da unidade cooperativa hierárquica U(U(Ta1,Ta2),Tfa) a fim de preservar a consistência mútua dos objetos Planta e Contrato.

Para melhor moldar este novo conceito de interação (cooperação) entre transações, faz-se necessária a elaboração de um novo modelo de cooperação entre transações denominado de Modelo de Unidades Cooperativas, MUC, que neste caso será dirigido pelos dados (Objetos Complementares), em oposição a outros modelos de cooperação existentes, que são dirigidos pelas aplicações.

2.3 O MODELO DE UNIDADES COOPERATIVAS (MUC)

No modelo de unidades cooperativas proposto em [Sampaio-95], um conceito fundamental é o de unidade cooperativa. Segundo este modelo, a manutenção da consistência de um conjunto de objetos componentes de um objeto complexo, que são conectados por relações de dependência (objetos complementares), é compartilhada entre várias transações que cooperam no intuito de preservar a consistência dos mesmos. Porém, antes de chegarmos ao conceito de unidades cooperativas, precisamos definir outros conceitos importantes, como por exemplo, os conceitos de Objetos Complementares, o Princípio de Responsabilidades Separadas, Transações Cooperativas. É o que faremos nas próximas seções.

2.3.1 OBJETOS COMPLEMENTARES

Os objetos Fachada e Interior mostrados na Figura 2.3a, do mesmo modo que os objetos Planta e Contrato, mantêm entre si uma relação de complementaridade (Relação Complementa, ver seção 2.1), ou seja, uma atualização sobre um desses objetos irá ter repercussão sobre o outro objeto.

Dois objetos compostos, X e Y, são diretamente complementares (ou simplesmente complementares) indicado por X © Y, se não for possível dizer que o estado de um dos objetos é consistente sem conhecer o estado do outro objeto. Formalmente, X © Y ⇔ (estado consistente (X) ⇔ estado consistente (Y)), sendo que X © Y é uma relação binária e simétrica.

Um conjunto OC de objetos complementares é definido como segue. Dado dois objetos de OC, eles são (transitivamente) complementares, isto é, $\forall X_i, X_J \{ X_i , X_J \} \in OC \Leftrightarrow (X_i \copyright^* X_j)$[2]. O conjunto OC é maximal se não houver qualquer objeto que pertence a OC, e qualquer outro objeto que não pertence a OC que sejam (transitivamente) complementares, ou seja, $\forall X, Y (X \in OC) \wedge (Y \notin OC) \Rightarrow \neg (X \copyright^* Y)$. De outra forma, é um subconjunto de um conjunto maximal de objetos complementares , ou simplesmente um conjunto de objetos complementares.

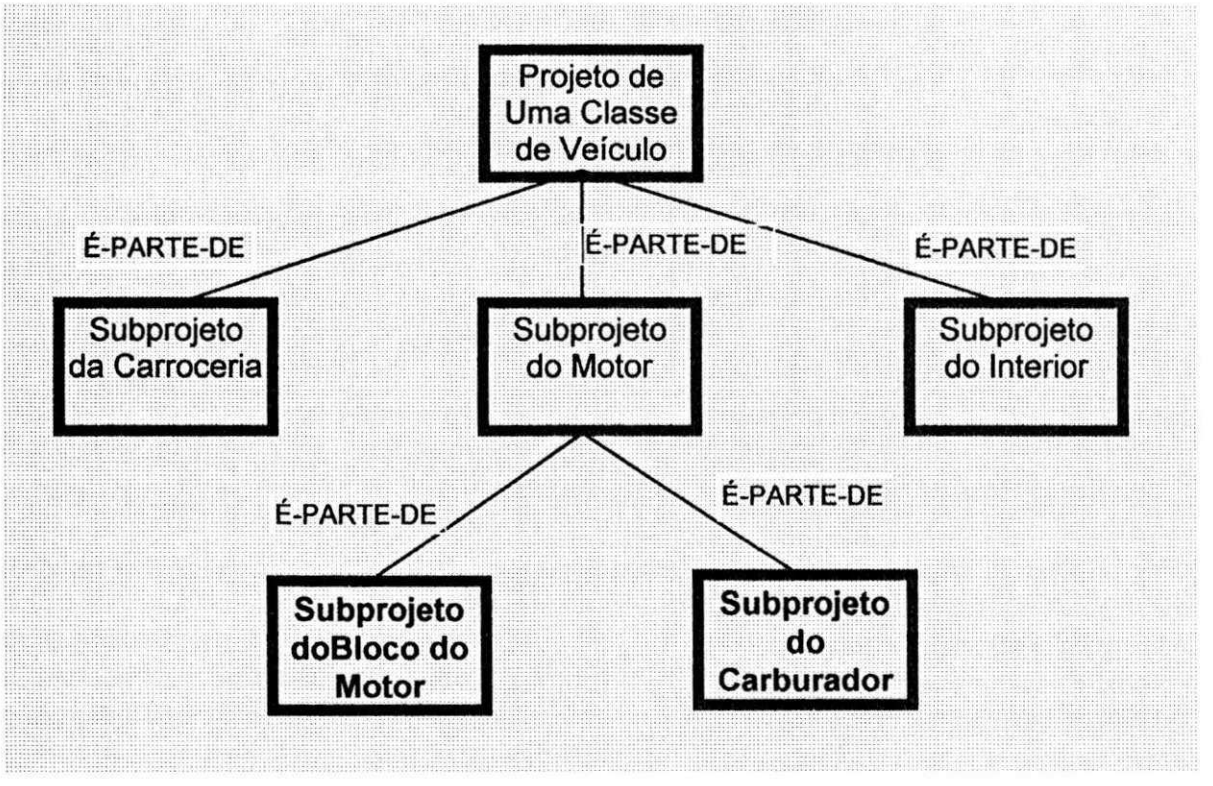


Figura 2.4 - Objeto Complexo Projeto de uma Classe de Veículo.

[2] ©* indica Complementaridade Transitiva, isto é, se X1©X2, X2©X3 e ¬(X1©X3), então (X1©*X3).

Como um outro exemplo de objetos que são complementares no interior de um objeto complexo, consideremos o Projeto de uma Classe de Veículo. O projeto de uma Classe de veículo é composto pelos objetos Subprojeto da Carroceria, Subprojeto do Interior e Subprojeto do Motor, sendo que este por sua vez é constituído pelos Subprojeto do Carburador e pelo Subprojeto do Bloco do Motor. A Figura 2.4 ilustra o objeto complexo Projeto de uma Classe de Veículo. Dentro deste objeto, vemos que os objetos componentes Subprojeto do Bloco do Motor e Subprojeto do Carburador são diretamente complementares, ou seja, não se pode dizer que o objeto Subprojeto do Bloco do Motor é consistente sem conhecer o estado do outro objeto componente Subprojeto do Carburador, e vice-versa.

2.3.2 PRINCÍPIO DE RESPONSABILIDADES SEPARADAS

No MUC, a cooperação entre as transações se dá segundo o Princípio de Responsabilidades Separadas. Este princípio afirma que a atualização de um conjunto maximal de objetos que são complementares se dá através do compartilhamento desta atualização por um conjunto de usuários, sendo que cada objeto é atualizado apenas por um usuário. A definição formal deste princípio é a seguinte:

Seja UC, um conjunto de usuários que compartilham a atualização de um conjunto maximal de objetos complementares OC. Seja $UC_i \in UC$ um usuário o qual atualiza os objetos de $OC_i \subset OC$. Seja $UC_J \in UC$, $UC_J \neq UC_i$ um usuário o qual atualiza os objetos de $OC_J \subset OC$. Então, deve-se ter: $\forall$ UCi ,UCJ $\{ UC_i, UC_J \} \subseteq UC \Rightarrow OC_i \cap OC_J = \varnothing$. Desta forma, um objeto de um conjunto (maximal) de objetos complementares só será atualizado por um único usuário, digamos Ui.

2.3.3 TRANSAÇÕES COOPERATIVAS

Usuários, cooperando sob o Princípio de Responsabilidades Separadas, são representados por transações cooperativas. Essas transações cooperativas, ao contrário das transações convencionais, não poderão ser programadas previamente porque elas precisam interagir de forma imprevisível, e sua duração é indeterminada.

As propriedades I e C dessas transações, necessitam ser adaptadas às suas características (trabalho cooperativo ou em grupo).

Voltemos ao exemplo ilustrado na Figura 2.3. Temos a transação Ta1 modificando o objeto Fachada e a transação Ta2 modificando o objeto Interior (Fachada e Interior são complementares) dentro da subunidade cooperativa U(Ta1,Ta2). Temos também a transação U(Ta1,Ta2) e a transação Tfa, modificando respectivamente os objetos complementares Planta e Contrato.

Com respeito à propriedade de consistência de transações cooperativas, diz-se que uma transação cooperativa é *parcialmente consistente* (Cp), ou seja, ela deverá preservar a consistência apenas de um subconjunto de objetos complementares, junto com outras transações cooperativas, **não** sendo sua função a preservação da consistência do conjunto maximal de objetos. As transações cooperativas Ta1 e Ta2 compartilham entre si, através do Princípio de Responsabilidades Separadas, a tarefa de atualizar o objeto Planta, o qual é constituído pelos objetos Fachada e Interior. Como conseqüência, cada uma dessas transações deverá ser apenas parcialmente consistente (Cp), pois irão preservar a consistência de apenas uma parte do objeto Planta.

Quanto ao isolamento entre as transações cooperativas, este irá depender diretamente do nível de cooperação apresentado entre as mesmas. Estas transações poderão cooperar "fracamente" ou "fortemente", conforme a execução destas transações apresente a propriedade da serializabilidade ou não. A definição desses tipos de cooperação será vista a seguir.

TRANSAÇÕES COOPERANDO FRACAMENTE

Neste tipo de cooperação, a execução das transações cooperativas é *serializável,* ou seja, produzem o mesmo efeito em relação à execução deste mesmo conjunto de transações em série. No entanto, a grande novidade com relação a essas transações que cooperam fracamente se deve ao fato de que suas validações são seletivas, ou seja, uma transação cooperativa irá validar seus efeitos finais para uma determinada transação selecionada previamente, que irá ler estes resultados e adequar os seus resultados de acordo com eles, sem no entanto poder questionar os resultados finais da primeira transação.

Como exemplo de transações cooperando fracamente, vamos mostrar uma seqüência ordenada de operações correspondentes a cada uma das transações, a qual nós denominaremos de "história" das transações.

Iremos para isto nos basear na Figura 2.3, onde as transações Ta1 e Ta2 irão compartilhar a tarefa da atualização do objeto Planta, constituído pelos objetos Fachada e Interior. As transações cooperam fracamente, segundo o Princípio de Responsabilidades Separadas. Na Figura 2.5 é mostrado a "história" correspondente a essas transações.

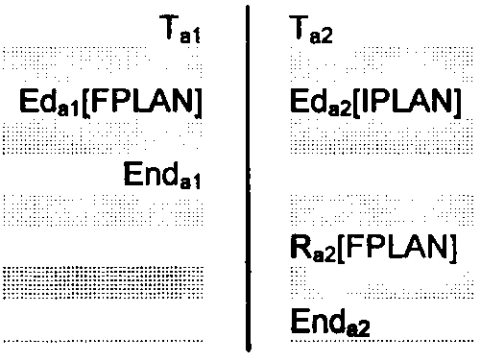


Figura 2.5 Transações Cooperando Fracamente.

As operações $Ed_i[X]$, End_i e $R_i[X]$significam respectivamente a operação de edição da transação i sobre o objeto X, a operação de validação definitiva da transação i e a operação de leitura do objeto X pela transação i.

Vemos que as transações Ta1 e Ta2, que cooperam fracamente, apresentam uma execução serializável (primeiro Ta1, e depois Ta2) e que a transação Ta1 irá cooperar com Ta2, fazendo apenas os seus resultados finais visíveis para esta, que irá ler os seus efeitos sem no entanto poder questioná-los.

TRANSAÇÕES COOPERANDO FORTEMENTE

As transações cooperativas que cooperam fortemente não mais apresentarão a característica de serializabilidade como encontrado nas transações que cooperam fracamente. Ao contrário destas, as transações que cooperam fortemente se caracterizam principalmente por fazer os seus efeitos intermediários visíveis apenas a transações selecionadas.

Neste sentido, a terminação de uma transação cooperando fortemente é hipotética, ou seja, os seus resultados poderão ou não serem ainda questionados pelas transações selecionadas, caso estes estejam ou não consistentes com os resultados destas. Por outro lado, caso os resultados estejam consistentes, a primeira transação, junto com as outras transações cooperando fortemente, irá então, validar definitivamente. Para o caso em que os resultados de uma transação que coopera fortemente com outras transações forem questionados, então a sua validação será hipotética e os seus resultados serão apenas intermediários, e como conseqüência, esta transação deverá refazer as suas operações (volta ao seu estado ativo) de forma que os seus resultados estejam de acordo (consistentes) com os resultados das transações selecionadas e vice-versa.

Para um melhor entendimento deste tipo de cooperação, vamos nos basear novamente na Figura 2.3 onde as transações Ta1 e Ta2 irão cooperar a fim de atualizarem o objeto Planta. A "história" destas transações cooperando fortemente, é mostrada na Figura 2.6.

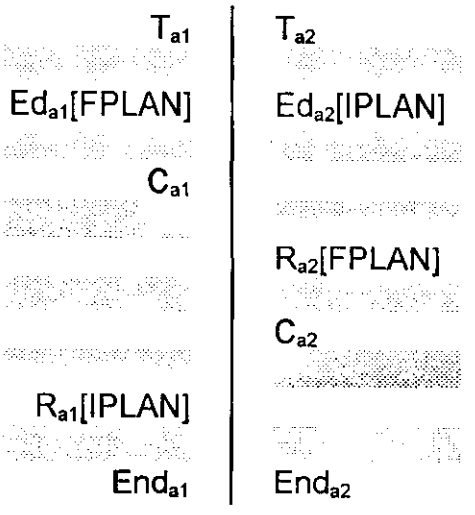


Figura 2.6 - Transações Cooperando Fortemente.

A operação Ci mostrada na Figura 2.6, diferentemente de Endi, irá validar apenas hipoteticamente uma transação i. Em outras palavras, uma transação que executou uma operação Ci poderá ainda retornar ao seu estado ativo, caso os seus resultados não estejam consistentes com os resultados das transações selecionadas, de forma que a sua validação é apenas hipotética.

Neste exemplo vemos que a transação Ta1 coopera fortemente com a transação Ta2, pois as duas transações só irão validar definitivamente seus resultados depois de que cada uma delas mostra os seus resultados para a outra (validação seletiva). Para isto, temos a operação Ci que é o commit de uma transação i, por exemplo, Ta1, que significa que a transação Ta1 fará visíveis os seus resultados apenas para transações selecionadas, por exemplo Ta2 e vice-versa.

Caso estes resultados estejam mutuamente consistentes, eles deverão ser validados definitivamente e simultaneamente, fazendo-se visíveis para todas as outras transações do sistema. Caso contrário, se os resultados não estiverem consistentes, as transações voltam ao seu estado ativo, retornando as suas respectivas operações de edição para que finalmente os resultados possam estar mutuamente consistentes.

2.3.4 UNIDADE COOPERATIVA

Uma unidade cooperativa é um conceito lógico que representa a execução de um conjunto de transações cooperativas, junto com uma transação de controle, que dividem a tarefa de manter a consistência dos objetos associados com esta unidade cooperativa.

A transação de controle, por sua vez, tem a função de coordenar a execução das transações cooperativas associadas com a unidade cooperativa, executando, por exemplo, a transferência de bloqueios entre essas transações. Todas as transações que não pertencem a uma unidade cooperativa hierárquica são consideradas transações externas a mesma. Do ponto de vista destas transações, a unidade cooperativa hierárquica deverá manter as propriedades de durabilidade, consistência e isolação.

As unidades cooperativas são classificadas de acordo com o grau de cooperação das transações cooperativas em seu interior, de modo que estas transações, como visto em seções anteriores, poderão cooperar fracamente ou fortemente.

Como conseqüência, uma unidade cooperativa fraca ou uma unidade cooperativa forte é aquela cujas transações em seu interior, cooperam fracamente ou cooperam fortemente.

Para melhor ilustrar estas unidades cooperativas, vamos mostrar dois exemplos de transações que estão cooperando fracamente ou fortemente no interior de uma unidade cooperativa fraca ou forte respectivamente, junto com uma transação externa, Te, a estas unidades cooperativas.

EXEMPLO 1 : UNIDADE COOPERATIVA FRACA

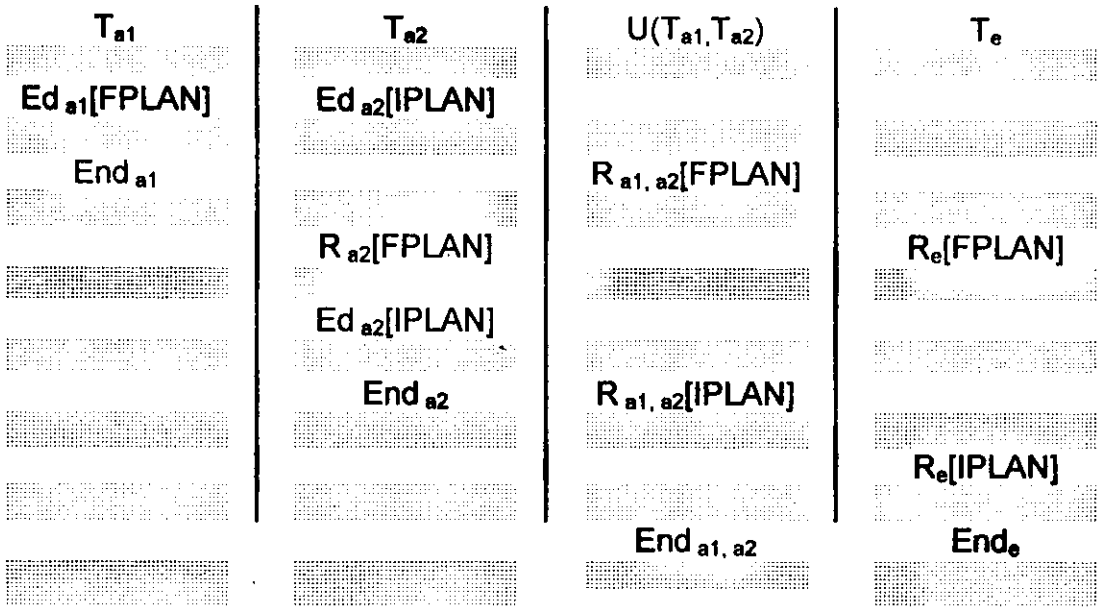


Figura 2.7 Unidade Cooperativa Fraca

Com relação à Figura 2.7, vemos que a transação Ta1 irá cooperar com a transação Ta2, fazendo os seus resultados finais visíveis a esta transação, a fim de que os resultados finais da transação Ta2 sejam consistentes com os resultados finais da transação Ta1. Além disso, quando uma transação valida definitivamente em uma unidade cooperativa fraca, os efeitos desta transação não poderão mais serem questionados por parte de qualquer outra transação.

Ainda nesta figura, vemos que para uma transação que é externa a esta unidade cooperativa, ela irá fazer visíveis uma parte dos seus resultados antes da sua terminação definitiva. Com isto, uma unidade cooperativa fraca é não-estrita, ou seja, a transação externa a esta unidade não terá de esperar a terminação definitiva das transações pertencentes à ela para ver uma parte dos seus efeitos.

EXEMPLO 2 : UNIDADE COOPERATIVA FORTE

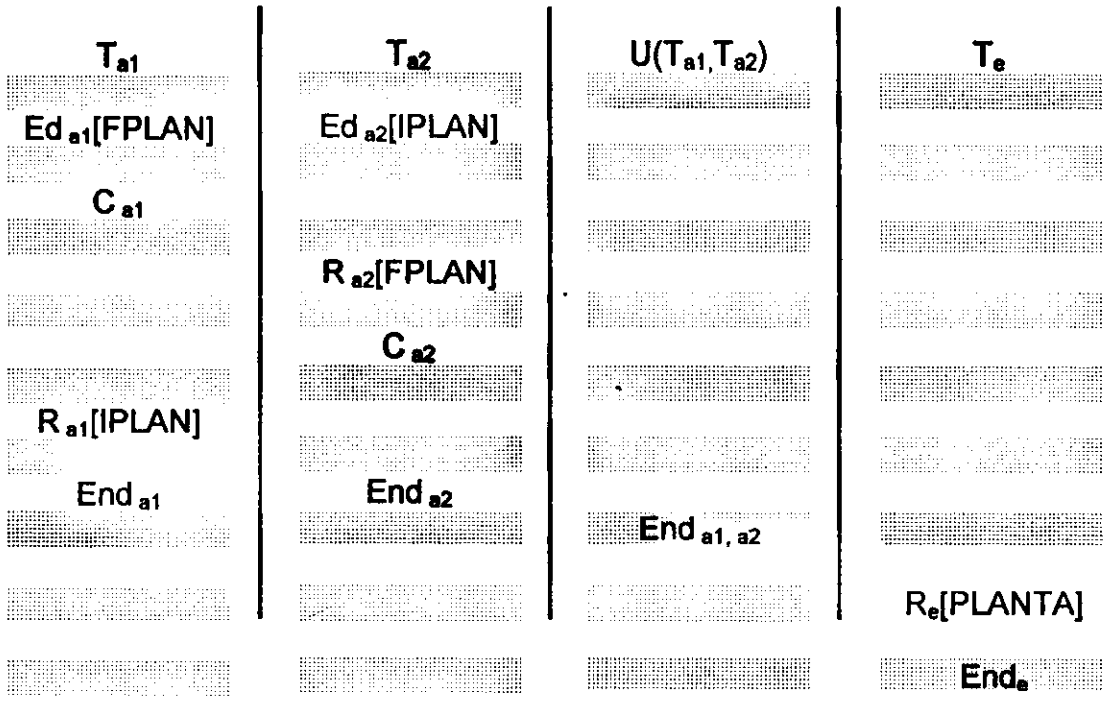


Figura 2.8 - Unidade Cooperativa Forte

Vemos nesta figura que uma transação que é externa à esta unidade cooperativa não terá acesso aos seus resultados antes do término definitivo da mesma. Isto é devido ao fato de que as transações que estão em uma unidade cooperativa forte irão terminar definitivamente de forma simultânea, liberando os seus resultados para transações externas à unidade cooperativa. Em outras palavras, sob o ponto de vista de uma transação externa à unidade cooperativa, ela é estrita, pois os seus resultados serão apenas visíveis após o seu término definitivo.

Ainda com relação à Figura 2.8, vemos que tanto a transação Ta1 quanto a transação Ta2, farão visíveis os seus resultados parciais, que poderão ser finais ou não caso estes sejam questionados ou não, apenas para transações selecionadas que estão dentro da unidade cooperativa. Como ilustração, a transação Ta1 terminará hipoteticamente, podendo ainda retornar ao seu estado ativo, caso os seus resultados não estejam consistentes com os resultados das outras transações dentro da unidade cooperativa, no caso Ta2, e inversamente os resultados parciais de Ta2 terão que estar consistentes com os resultados de Ta1.

Novamente com relação à Figura 2.8, vemos que os resultados não foram questionados por qualquer uma das partes após as suas terminações hipotéticas.

Como neste trabalho, o foco é com relação a objetos complexos, as transações cooperativas que manipulam os objetos componentes destes, constituem uma árvore de transações que poderá refletir a estrutura do objeto complexo. Por sua vez, as unidades cooperativas associadas com estas transações deverão ser estruturadas hierarquicamente, a fim de também refletir a estrutura do objeto complexo ao qual está associada, constituindo assim em uma unidade cooperativa hierárquica.

Como exemplo, seja a Figura 2.3 mostrada anteriormente, onde associada ao objeto complexo Projeto tem-se uma árvore de transações, cuja transação U(U(Ta1,Ta2), Tfa) é a transação-raiz. Além disso, as transações U(Ta1,Ta2) e Tfa atuando nos objetos de Planta e de Contrato, respectivamente, irão cooperar fracamente a fim de atualizarem o objeto de Projeto, constituindo uma unidade cooperativa, que é controlada pela transação-raiz U(U(Ta1,Ta2),Tfa).
Por sua vez, as transações Ta1 e Ta2 manipulando os objetos de Fachada e de Interior irão cooperar fortemente, a fim de atualizarem o objeto de Planta, constituindo assim uma subunidade cooperativa, ou seja, que está dentro de uma outra unidade cooperativa, sendo controlada pela transação U(Ta1,Ta2)

Vamos, finalmente, mostrar a seqüência ordenada de operações correspondente a cada uma dessas transações que pertencem à unidade cooperativa hierárquica mostrada na Figura 2.3. Iremos supor que as transações Ta1 e Ta2 estão no interior de uma unidade cooperativa forte, e que as transações U(Ta1,Ta2) e Tfa estão no interior de uma unidade cooperativa fraca.

EXEMPLO 3 : UNIDADE COOPERATIVA HIERÁRQUICA

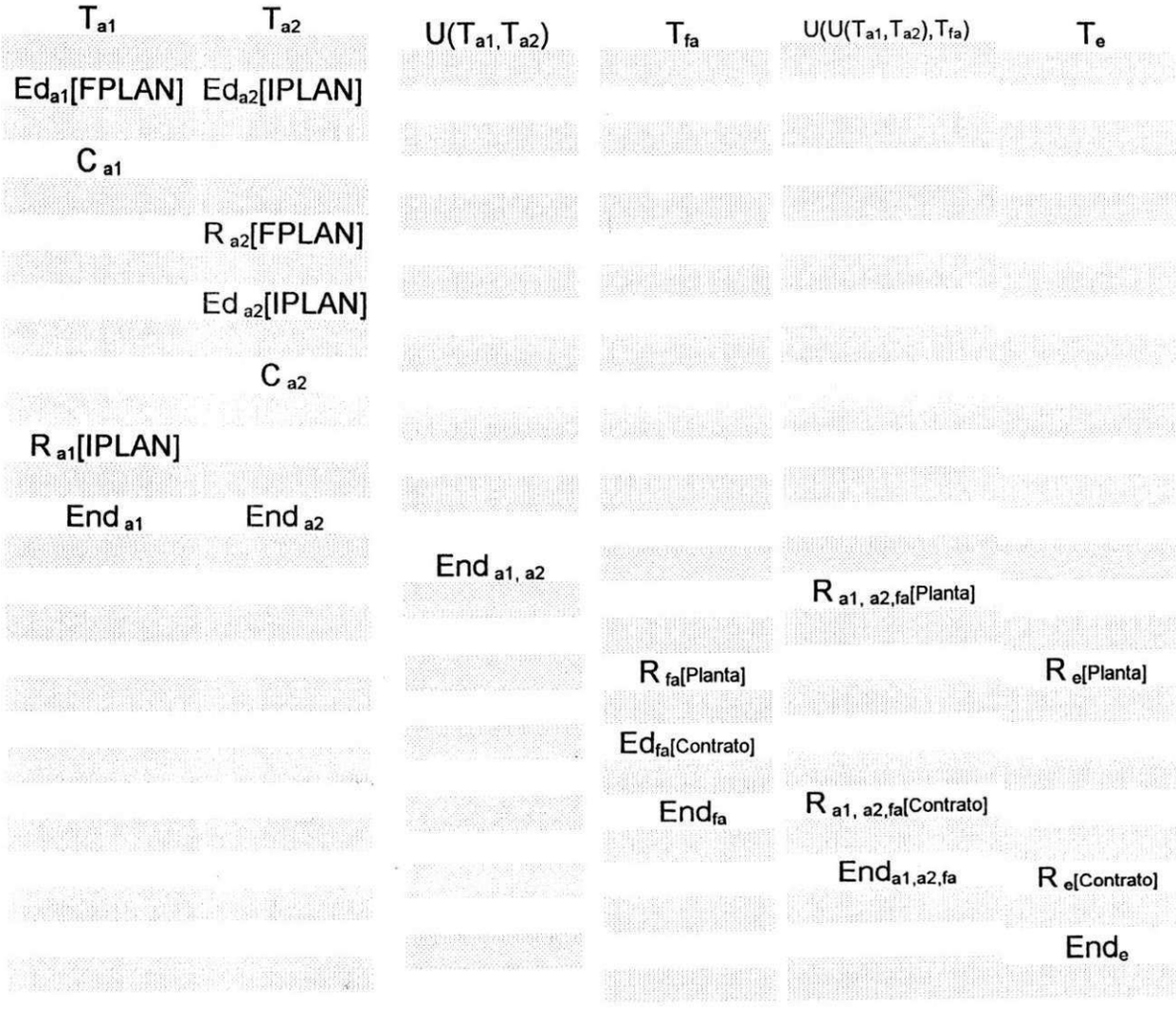


Figura 2.9 Unidade Cooperativa Hierárquica

3 Protocolos de Controle de Concorrência Existentes na Literatura

O modelo de transações cooperativas proposto em [Sampaio-95], e apresentado no capítulo anterior, tem características bem diferentes com relação ao conhecido modelo de transações convencionais. Segundo este novo modelo, as transações cooperativas que atualizam simultaneamente partes de um mesmo objeto estruturado, segundo o Princípio de Responsabilidades Separadas, se situam logicamente no interior de uma unidade cooperativa, que poderá ser classificada em "forte" ou "fraca", conforme o grau de cooperação seja "forte" ou "fraco", respectivamente. Estas transações apresentam em suas execuções características distintas das transações convencionais, como em relação à duração, à interatividade etc.

Em virtude de características tão distintas, estas transações têm diferentes exigências para protocolos de controle de concorrência. Nós discutiremos alguns protocolos de CC existentes na literatura para transações não-convencionais e mostraremos que eles não são adequados para transações cooperativas em geral, e em particular para as transações cooperativas segundo o modelo proposto em [Sampaio-95] (Modelo MUC). No entanto, nós nos inspiramos em várias das idéias contidas nesses protocolos para elaborar o nosso protocolo de CC, adequado ao MUC.

3.1 PROTOCOLOS BASEADOS EM GRANULAGENS DE BLOQUEIO

A motivação fundamental de um protocolo baseado em granulagens de bloqueio[3] [Gray78] é minimizar o número de bloqueios a serem adquiridos para acessar objetos de um banco de dados . Por exemplo, considere a Figura 3.1, que mostra os recursos de um banco de dados, estruturados segundo as suas granulagens. Quando a maior parte dos registros de um arquivo estão para serem acessados por uma transação, será mais eficiente bloquear o arquivo inteiro, ao invés de bloquear cada registro separadamente, já que o bloqueio em um arquivo implicitamente bloqueia cada registro deste. Nós dizemos que, neste caso, a granulagem de bloqueio é o arquivo. Por outro lado, quando apenas poucos registros de um arquivo necessitam ser acessados, será melhor bloquear os registros separadamente, de modo que outras transações possam acessar concorrentemente outros registros do arquivo. Neste caso nós dizemos que a granulagem de bloqueio é o registro.

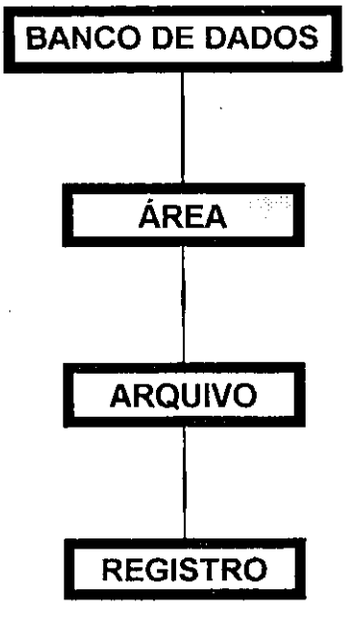


Figura 3.1 Recursos de um Banco de Dados estruturados Hierarquicamente.

[3] Usaremos também a palavra unidade para indicar o tamanho do bloqueio.

Com respeito à definição da granulagem de bloqueio, esta deve representar um compromisso entre o nível de concorrência desejado e o *overhead* do sistema, o qual se relaciona com o tamanho ou granulagem de bloqueio.

Um nível maior de concorrência no sistema será atingido se for definido uma unidade de bloqueio menor. Unidades de bloqueio menores mostram-se adequadas a transações simples, ou seja, que acessam poucos objetos simples ou atômicos. Neste caso, tomando como base novamente a Figura 3.1, se uma transação for acessar poucos registros de um determinado arquivo, será mais adequado, como já mencionado, bloquear cada registro isoladamente. Por outro lado, e ainda tomando como base a Figura 3.1, para o caso de transações complexas, que acessam um grande número de objetos simples, uma pequena unidade de bloqueio mostra-se pouco adequada, pois a transação irá ter que acessar um grande número de registros pertencentes a um arquivo. Tais tipos de transações necessitam de uma maior unidade de bloqueio, pois uma unidade menor irá obrigá-la a bloquear uma grande quantidade de registros, acarretando então um *overhead* devido ao grande número de acessos ao "gerenciador de bloqueios" e devido à representação e controle desses bloqueios.

Pode-se concluir que como coexistem diversos tipos de transações dentro de um mesmo sistema, é desejável que se tenha unidades de bloqueio de diversos tamanhos coexistindo.

Na próxima seção, vamos apresentar um protocolo de controle de concorrência, proposto em [Gray-78], que se apoia no conceito de unidade de bloqueio.

3.1.1 PROTOCOLO DE GRAY

O protocolo em [Gray-78] é um protocolo aplicável somente a transações de um só nível que são serializáveis, e que não dividem a tarefa de manter a consistência do banco de dados. Em outras palavras, cada transação neste protocolo é considerada consistente.

Inicialmente, vamos assumir que o conjunto de recursos que vão ser bloqueados em um banco de dados estão organizados segundo a hierarquia mostrada na Figura 3.1. Esses recursos são do tipo área, arquivo, registro, sem relação portanto com o modelo de dados do banco de dados.

Todos os nós da hierarquia poderão ser explicitamente bloqueados. Caso uma transação faça uma requisição de bloqueio de atualização (modo exclusivo) em um determinado nó na hierarquia, então os nós descendentes são implicitamente bloqueados.

Da mesma maneira, se alguma transação do sistema fizer uma requisição de bloqueio de leitura (modo compartilhado) em um determinado nó, todos os nós descendentes do nó bloqueado são implicitamente bloqueados em modo leitura. Disto, conclue-se que estes modos de bloqueio bloqueiam uma sub-árvore completa com raiz no nó no qual foi colocado o bloqueio.

Para que se bloqueie uma sub-árvore completa, deve-se ainda prevenir que haja bloqueios sobre os nós ancestrais da sub-árvore de forma conflitante. Em função disto, um novo modo de bloqueio deve ser introduzido: o modo Intenção (I). Este modo de bloqueio é utilizado para marcar todos os ancestrais de um determinado nó que foi bloqueado em modo exclusivo (X) ou em modo compartilhado (S), de modo que se previna a ocorrência de bloqueios de forma conflitante sobre o mesmo.

Serão mostrados adiante os modos de bloqueio utilizados no protocolo e as características associadas com esses modos.

MODOS DE BLOQUEIO

Diz-se que duas requisições de bloqueio para um mesmo recurso em um sistema são compatíveis se elas podem ser concedidas concorrentemente, sendo que o modo da requisição determina sua compatibilidade ou não com a requisição já feita por outras transações.

A incompatibilidade ou não entre os diversos modos de bloqueio existentes irá depender da semântica associada a cada um destes modos de bloqueio. Para melhor compreendermos os modos de bloqueio aos diversos recursos do sistema, vamos nos basear ainda na Figura 3.1.

O modo de intenção de bloqueio, I, foi introduzido para ser incompatível com os modos compartilhado e exclusivo. Este modo pode ser ainda refinado para os modos IS (Intenção de Leitura) ou IX (Intenção de Escrita), significando por exemplo que, se uma transação T1 colocar um bloqueio em modo IX sobre um nó da Figura 3.1 (um arquivo, por exemplo), esta deverá bloquear algum nó descendente deste na hierarquia em modo X ou mesmo em modo IX. Ao mesmo tempo, uma outra transação não poderá bloquear o arquivo em modo X ou em modo S, mas poderá fazê-lo apenas nos modos IX ou IS.

Por outro lado, se uma transação T2 colocar um bloqueio em modo IS sobre um nó (um arquivo) da Figura 3.1, esta deverá bloquear algum nó descendente deste na hierarquia em modo S ou em modo IS, ao mesmo tempo, uma outra transação poderá bloquear o arquivo nos modos S, IS ou IX, não sendo permitido que ela bloqueie o arquivo em modo X.

O modo compartilhado (S) permite, por parte da transação requisitante, e também por parte de uma outra transação concorrente, a leitura simultânea do recurso correspondente. Por exemplo, segundo a Figura 3.1, será permitido que duas transações, T1 e T2, bloqueiem simultaneamente em modo compartilhado, um mesmo registro pertencente a um mesmo arquivo.

No modo exclusivo (X), a transação requisitante poderá ler ou modificar o recurso, ao passo que nenhuma outra transação poderá em modo algum ter acesso ao mesmo. Como exemplo, segundo a Figura 3.1, se uma transação T1 detiver um bloqueio em modo exclusivo sobre um registro de um arquivo, nenhuma outra transação poderá ler ou modificar este registro enquanto a transação T1 detiver este bloqueio.

Define-se ainda um refinamento suplementar dos modos de bloqueios, que é o modo compartilhado com intenção de modo exclusivo (SIX). Através deste modo de

bloqueio, uma transação poderá bloquear uma sub-árvore inteira para leitura intensiva e atualização de apenas alguns nós desta sub-árvore. Por exemplo, na Figura 3.1, se uma transação T1 colocar um bloqueio em modo SIX sobre um arquivo, significará que ela bloqueará o arquivo em modo de leitura e poderá ainda atualizar alguns registros pertencentes ao arquivo. Neste caso, cada registro a ser atualizado é bloqueado em modo X, explicitamente.

A tabela da Figura 3.2 abaixo nos mostra a compatibilidade dos diferentes modos de bloqueio apresentados nesta seção. Na figura, s é "sim", significando por exemplo a compatibilidade entre os modos de bloqueio por parte de uma transação que obteve um bloqueio sobre um recurso do banco de dados e por parte de uma outra transação que tenta obter um bloqueio sobre este recurso, e n é "não" significando a incompatibilidade entre os modos de bloqueio das transações.

	IS	IX	S	SIX	X
IS	s	s	s	s	n
IX	s	s	n	n	n
S	s	n	s	n	n
SIX	s	n	n	n	n
X	n	n	n	n	n

Figura 3.2 Matriz de Compatibilidade - Protocolo de Gray

Iremos finalmente resumir a semântica associada a cada um dos modos de bloqueio discutidos até aqui.

IS - Permite a transação requisitante bloquear os nós descendentes em modo S ou modo IS (não tem bloqueio implícito).

IX - Permite a transação requisitante bloquear os nós descendentes nos modos IX, S, SIX, IX ou IS. (não tem bloqueio implícito).

S - Dá à transação requisitante um acesso compartilhado a um determinado nó e a todos os descendentes do referido nó, sem a necessidade de bloqueios adicionais (a

transação implicitamente coloca bloqueio do tipo S em todos nós descendentes do referido nó).

X - Dá à transação requisitante um acesso exclusivo a um determinado nó e a todos os descendentes do referido nó, sem a necessidade de bloqueios adicionais (a transação implicitamente coloca bloqueio do tipo X em todos nós descendentes do referido nó.).

SIX - Dá à transação requisitante um acesso compartilhado e de intenção de modo exclusivo a um determinado nó (a transação implicitamente bloqueia todos os descendentes do nó em modo compartilhado e permite bloquear explicitamente nós descendentes em modo X, SIX ou IX).

O PROTOCOLO DE BLOQUEIO DE HIERARQUIAS DE RECURSOS

O protocolo de bloqueio sobre uma hierarquia de recursos que foi apresentado em [Gray-78], que é apresentado a seguir, garante que, numa execução concorrente de transações, se cada uma delas for regida por este protocolo, então a consistência do banco de dados será preservada.

1 - Antes de requisitar um bloqueio do tipo S ou IS em um nó, todos os ancestrais do nó deverão ser bloqueados em modo IX ou IS pela transação requisitante.

2 - Antes de requisitar um bloqueio do tipo X, SIX ou IX em um nó, todos os ancestrais do nó deverão ser bloqueados em modo SIX ou IX pela transação requisitante.

3 - Bloqueios poderiam ser liberados tanto ao final de uma transação em qualquer ordem, como do nó-folha para o nó-raiz, respeitados as duas fases, aquisição de bloqueios e liberação de bloqueios, respectivamente.

Para um melhor entendimento deste protocolo, considere o exemplo a seguir que está baseado na Figura 3.1.

EXEMPLO 1: Bloquear um registro R em modo compartilhado S.

(a) - Bloqueio do banco de dados em modo IS.

(b) - Bloqueio da área contendo o registro em modo IS.

(c) - Bloqueio do arquivo contendo o registro em modo IS.

(d) - Bloqueio do registro em modo S.

O PROTOCOLO DE BLOQUEIO DE GRAFOS DE RECURSOS

Uma hierarquia de recursos é um caso particular de um grafo, onde diferentemente de uma hierarquia, um nó do grafo pode ter vários ancestrais.

Os conceitos mencionados até aqui poderão ser estendidos para se referir a um grafo de recursos, que muitas vezes, é mais adequado para representar os recursos de um sistema e suas inter-relações. A Figura 3.3 a seguir mostra um grafo de recursos, que da mesma forma que uma hierarquia, não tem qualquer relacionamento com o modelo de dados utilizado pelo banco de dados.

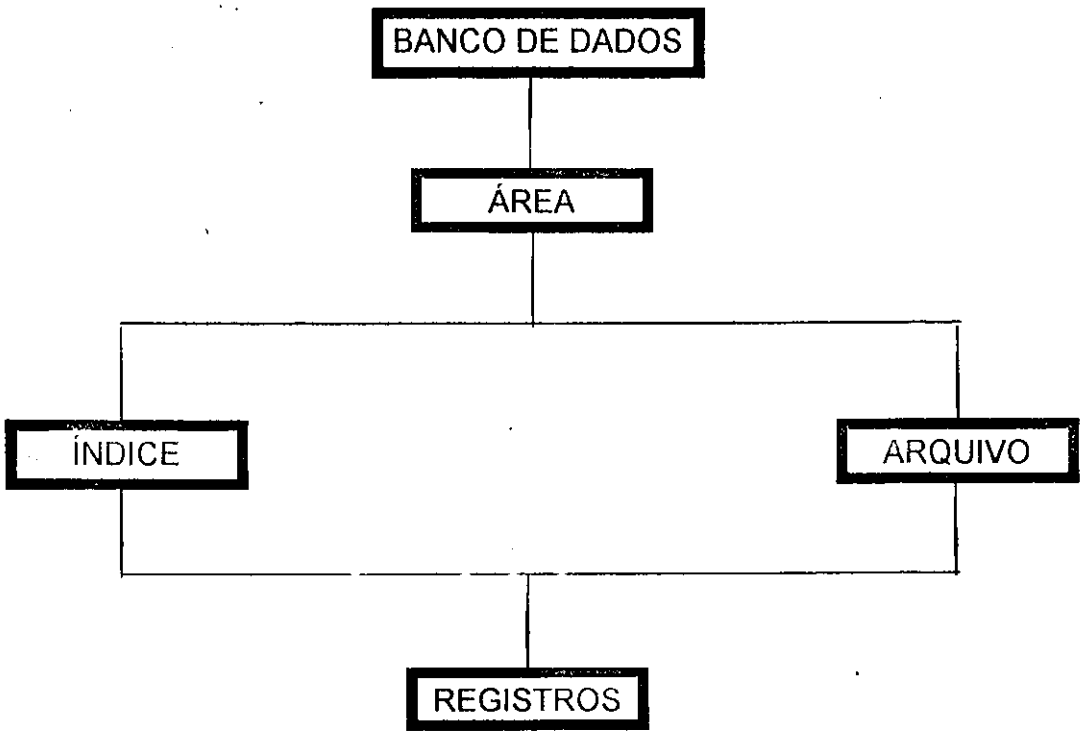


Figura 3.3 - Grafo de Recursos segundo suas Granulagens.

Com base na Figura 3.3, vemos que o banco de dados é constituído de um conjunto de áreas. Por sua vez, cada área é uma conjunto de arquivos e índices. Cada registro pertence a um arquivo e é acessado através de um índice.

Um nó em um grafo poderá ser bloqueado explicitamente (o bloqueio é sobre o próprio nó) ou implicitamente (o bloqueio explícito é sobre um nó ancestral deste).
A seguir, apresentamos o protocolo de bloqueio sobre grafos, que é uma extensão do protocolo de bloqueio sobre hierarquias.

1 - Para requisitar um bloqueio do tipo S ou IS em um nó, uma transação deverá possuir um bloqueio em modo IS em pelo menos um ancestral do nó (e por indução um caminho para a raiz). Como uma conseqüência, nenhum dos ancestrais ao longo deste caminho poderá ser bloqueado por outra transação em modo incompatível com IS.

2 - Para requisitar um bloqueio do tipo IX, SIX ou X em um nó, uma transação deverá possuir um bloqueio em modo IX em todos os nós ancestrais do nó. Como uma conseqüência todos os ancestrais do nó serão bloqueados em modo IX e não poderão ser bloqueados por outra transação em modo incompatível com IX.

3 - Bloqueios poderiam ser liberados tanto ao final de uma transação em qualquer ordem, como do nó-folha para o nó-raiz, respeitados as fases de aquisição e de liberação de bloqueios.

Para uma melhor compreensão deste protocolo, vamos considerar os exemplos abaixo, agora tomando-se como base a Figura 3.3

EXEMPLO 1: Bloquear um arquivo A em modo compartilhado S.

(a) - Bloqueio do banco de dados em modo IS
(b) - Bloqueio da área contendo o arquivo A em modo IS
(c) - Bloqueio do arquivo A em modo S

EXEMPLO 2: Atualizar um registro R em um arquivo F com índice I em modo exclusivo X.

(a) - Bloqueio do banco de dados em modo IX
(b) - Bloqueio da área contendo F em modo IX
(c) - Bloqueio do arquivo F em modo IX
(d) - Boqueio do índice I em modo IX
(e) - Bloqueio do registro R em modo X

ANÁLISE DO PROTOCOLO

Os protocolos de CC baseados em bloqueios para hierarquias e grafos discutidos neste capítulo apresentam um mecanismo de aquisição de bloqueios sobre os recursos de um banco de dados estruturados em hierarquias e grafos mostrados nas Figuras 3.1 e 3.3 que são independentes do modelo de dados do banco de dados. Entretanto, essas hierarquias ou grafos de recursos são ainda muito pobres semanticamente para suportar o rico modelo de dados do ADIC, sob o qual o MUC é implementado.

Relembremos que no modelo de dados do ADIC existem relações entre objetos como "IS-A", É-PARTE-DE, COMPLEMENTA etc. Além disso, o protocolo de Gray é aplicável apenas às transações de um só nível, isoladas umas das outras, ao contrário das unidades cooperativas, que são multiníveis e onde as transações em seu interior não são isoladas umas das outras (transações nem sempre serializáveis).

Entretanto, a noção de unidades de bloqueio e de bloqueios implícitos do protocolo de Gray influenciou decisivamente muitos outros protocolos de CC, inclusive o protocolo objeto desta dissertação.

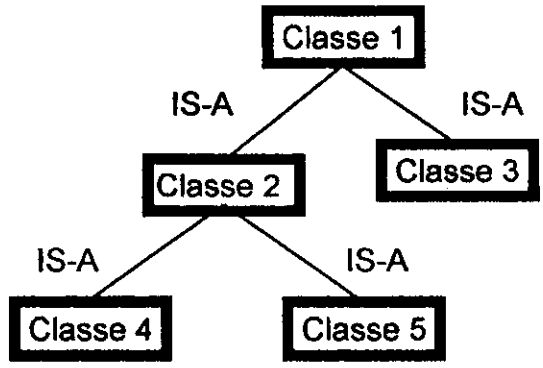
Na apresentação dos demais protocolos, nós nos restringiremos às hierarquias de objetos, sem considerar portanto grafos de objetos. A razão para isto é que o modelo de dados do ADIC não contempla grafos de objetos, mas unicamente hierarquia de objetos.

3.1.2 O PROTOCOLO II DE GRAY DE BLOQUEIO DE HIERARQUIAS DE OBJETOS

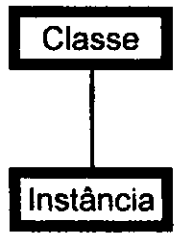
Este protocolo contempla dois aspectos principais: o primeiro, refere-se a aquisição/liberação de bloqueios em hierarquias de classes cujas classes mantêm entre si a relação "IS-A"; o segundo, refere-se a aquisição/liberação de bloqueios em hierarquias de classes, cujas classes mantêm entre si a relação É-PARTE-DE.

HIERARQUIAS "IS-A"

O protocolo para hierarquias "IS-A" é apenas uma extensão do protocolo de Gray para hierarquia de recursos. A Figura 3.4a, abaixo, mostra uma hierarquia cujas classes mantêm entre si a relação "IS-A". Esta hierarquia representa o caso onde uma classe só poderá possuir uma superclasse. Já a Figura 3.4b, mostra a hierarquia "Classe-Instância", onde uma *instância* (um objeto pertencente a uma classe) poderá pertencer a apenas uma classe.



3.4a) Hierarquia "IS-A"



3.4b) Hierarquia Classe-Instância

Os modos de bloqueio utilizados por esta abordagem são os mesmos propostos em [Gray-78], isto é, IS, IX, S, X e SIX.

Para o caso de hierarquias "Classe-Instância", as instâncias poderão ser bloqueadas apenas nos modos S e X, para indicar se elas serão lidas ou atualizadas respectivamente. Contudo, as classes poderão ser bloqueadas em qualquer um dos modos de bloqueio. Um bloqueio em modo IS em uma classe, significa que instâncias da classe poderão ser explicitamente bloqueadas em modo S quando necessário. Ao passo que um bloqueio em modo IX em uma classe significa que instâncias da classe poderão ser explicitamente bloqueadas nos modos S e X quando necessário. Um bloqueio em modo S em uma classe, significa que a definição da classe e todas as suas instâncias são bloqueadas em modo S. Por sua vez, um bloqueio em modo X em uma classe significa que a definição da classe e todas as suas instâncias são bloqueadas em modo X. Já um bloqueio em modo SIX em uma classe significa que a definição da classe é bloqueada em modo S, e todas as instâncias da classe são implicitamente bloqueadas em modo S, e algumas instâncias pertencentes à classe poderão ser bloqueadas em modo X.

Com relação a hierarquias "IS-A", de classes, os dois tipos de operações que podem ser definidas sobre essas hierarquias são: consultas (por exemplo, mostrar os atributos de uma classe) e modificações de definições de classe.

Note que as estruturas das Figuras 3.4a e 3.4b se fundem em uma só, considerando que toda classe de uma hierarquia "IS-A" tem suas instâncias. Desta forma, tanto consultas como modificações podem também se referir a instâncias de classes.

Com respeito às consultas, o espaço de pesquisa de uma consulta junto a uma classe C poderá ser tanto as instâncias de C, como também poderá incluir as instâncias das classes pertencentes a hierarquia de classes enraizada em C. Também, o domínio D de um atributo de uma classe C poderá ser uma outra classe D e todas as suas subclasses, significando que o espaço de pesquisa para uma consulta junto a uma classe inclui a hierarquia de classes enraizada na classe-domínio de cada um de seus atributos.

Nessas hierarquias de classes, uma classe C, que é uma subclasse de uma classe D, herda desta todos os seus atributos e métodos, acrescentando a estes os seus próprios atributos e métodos. Por esta razão, C é uma especialização da classe D e por sua vez a classe D é uma generalização de C, de modo que se forem adicionados ou removidos atributos e métodos de uma classe pertencente a uma hierarquia, estes também deverão ser adicionados ou removidos das suas correspondentes subclasses.

Desta forma, para que seja executada uma consulta sobre uma classe C, ou para o caso de uma modificação da definição da classe C, deverá ser colocado um bloqueio não apenas sobre a classe, mas também sobre cada uma de suas classes descendentes (subclasses) na hierarquia de classes.

Apresentamos a seguir a parte do protocolo II de Gray, proposto em [Gray-81], para o bloqueio em uma hierarquia de classes do tipo "IS-A".

PROTOCOLO II DE GRAY PARA HIERARQUIAS DO TIPO "IS-A".

1 - Para se bloquear uma classe em modo S ou IS, deverão ser colocados bloqueios nos modos S ou IS respectivamente sobre as suas subclasses.

2 - Para se bloquear uma classe em modo SIX, deverão ser colocados bloqueios nos modos SIX sobre as suas subclasses.

3 - Para se bloquear uma classe em modo X ou IX, deverão ser colocados bloqueios nos modos X ou IX respectivamente sobre as suas subclasses.

4 - Os bloqueios devem ser liberados em qualquer ordem ao término da transação.

Para melhor exemplificar este protocolo, vamos considerar a Figura 3.5, que representa o caso de uma hierarquia de classes, onde a classe Pessoa representa a classe-raiz dessa hierarquia, e as outras classes são subclasses da classe Pessoa.

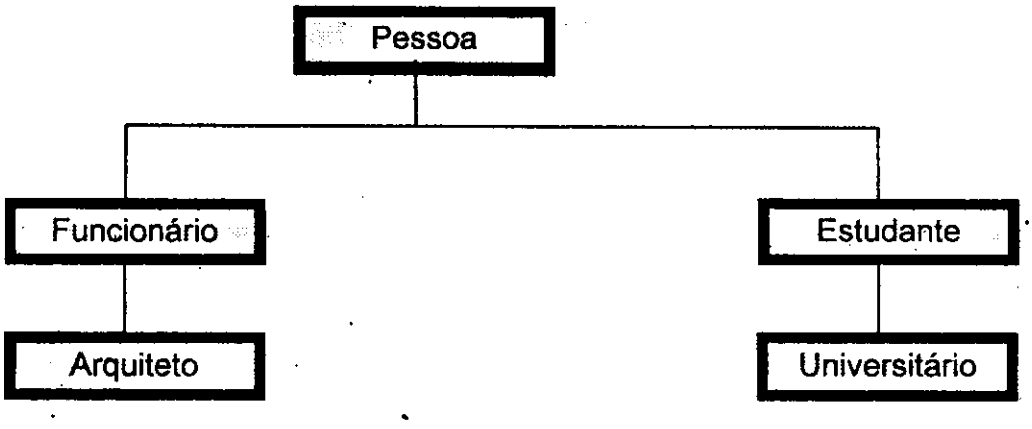


Figura 3.5 Hierarquia "IS-A" cuja Classe-Raiz é a Classe Pessoa

Os exemplos a seguir, serão baseados na Figura 3.5.

EXEMPLO 1: Selecionar todas as instâncias da classe Pessoa e suas subclasses tal que ...

(a) - Bloquear a classe Pessoa e aquelas subclasses de Pessoa em modo IS ou IX.

(b) - Bloquear cada instância selecionada em modo S.

EXEMPLO 2: Modificar a definição da classe Pessoa

(a) - Bloquear a classe Pessoa e aquelas subclasses de Pessoa em modo X

EXEMPLO 3: Suponha que uma transação T1 esteja modificando a definição da classe Funcionário, enquanto que uma outra transação T2 quer consultar algumas instâncias da classe Arquiteto. Os bloqueios por parte destas duas transações são mostrados a seguir:

T1 :	T2 :
X na classe Funcionário.	Tenta colocar IS na classe Arquiteto, mas
X na classe Arquiteto.	terá de esperar por T1.

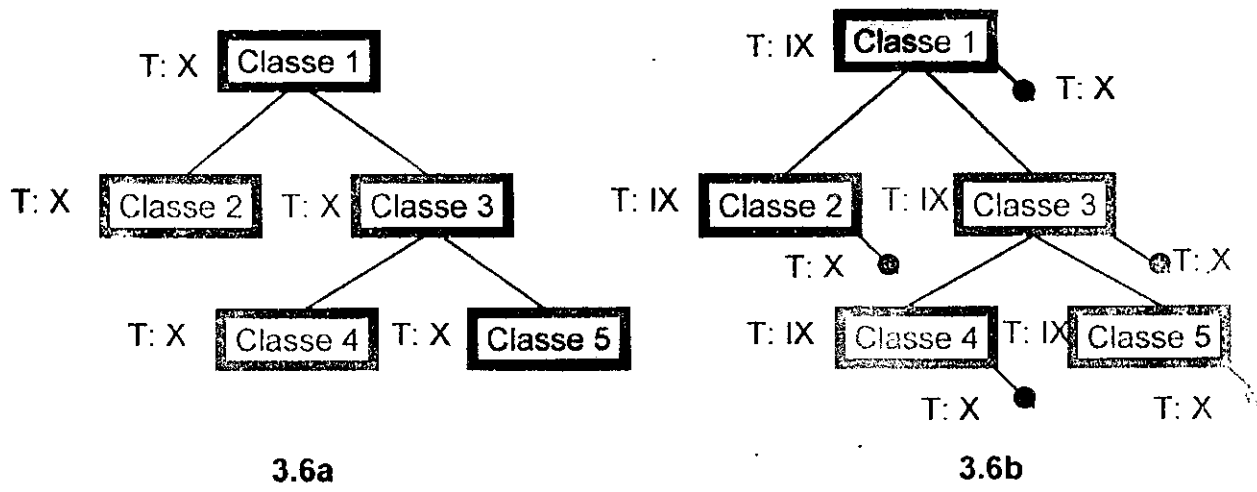
Segundo [Kim-90], esta abordagem é adequada para tipos de aplicações que acessem freqüentemente classes próximas ao nível mais baixo de uma hierarquia de classes. Para o caso de consultas ou mudanças de definição junto a uma classe que

está próximo à classe-raiz de uma profunda hierarquia de classes, esta abordagem mostra-se pouco adequada, pois irá acarretar um grande *overhead* para o "gerenciador de bloqueios", devido ao controle dos bloqueios colocados sobre as classes. Por exemplo, considerando-se ainda a Figura 3.5, o bloqueio da classe Pessoa, que é a classe-raiz da hierarquia, irá acarretar um maior overhead ao "gerenciador de bloqueios" do que o bloqueio da classe Arquiteto que pertence ao nível mais baixo da hierarquia.

PROTOCOLO II DE GRAY PARA HIERARQUIAS DO TIPO É-PARTE-DE

Um outro aspecto desta abordagem é aplicado a hierarquia de classes cujas classes mantêm entre si a relação É-PARTE-DE. Para estes casos, como também não são previstos bloqueios implícitos sobre a hierarquia de classes, para que se bloqueie uma instância de um objeto complexo em modo exclusivo ou compartilhado, deverá se proceder da seguinte maneira: ou bloquear todas as classes componentes que pertencem à estrutura em modo exclusivo ou compartilhado ou bloquear todos os objetos componentes pertencentes ao objeto complexo em modo exclusivo ou compartilhado.

Para que possamos visualizar melhor estas duas alternativas, considere o objeto complexo da Figura 3.6



3.6a **3.6b**

Figura 3.6 Objeto Complexo cuja Classe-Raiz é a Classe 1

Na Figura 3.6a, para que uma transação T bloqueie uma instância do objeto complexo em modo exclusivo, ela deverá bloquear todas as classes da estrutura em modo X. Por outro lado, na Figura 3.6b, para que uma transação T bloqueie uma instância de um objeto complexo em modo exclusivo, ela deverá bloquear todos os objetos componentes do objeto complexo em modo X. Para isto, a transação T deverá colocar um bloqueio nos modos IX sobre as classes pertencentes à hierarquia do objeto complexo e, um bloqueio em modo X nas instâncias selecionadas.

Entretanto, nenhuma das opções é satisfatória, sendo então necessária uma outra abordagem. Na primeira, teremos como conseqüência, o bloqueio de todos os objetos que estão baseados nesta estrutura acarretando uma degradação do nível de concorrência do sistema. A segunda, irá resultar num grande número de bloqueios para serem colocados e gerenciados.

ANÁLISE DO PROTOCOLO

Esta abordagem apresentou um mecanismo de aquisição de bloqueios sobre hierarquias de classes que apresentam a relação "IS-A" entre elas. Para o caso de hierarquias de classes que mantêm entre si a relação É-PARTE-DE ela não apresenta uma solução satisfatória.

Como no modelo MUC, apresentado no capítulo 2, as transações lidam tanto com hierarquias de classes que mantêm entre si as relações "IS-A" como com hierarquias de classes que mantêm a relação É-PARTE-DE, vemos claramente que o protocolo II de Gray supre apenas uma parte uma das exigências de bloqueio para o modelo MUC.

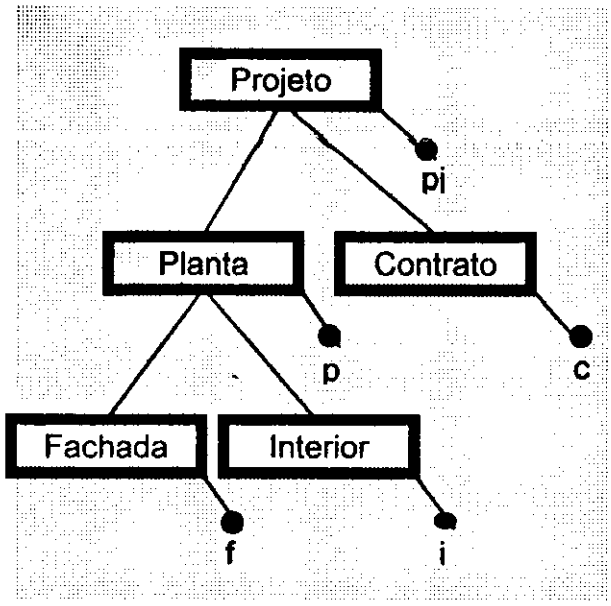


Figura 3.7 O Objeto Complexo Projeto junto com uma Instância de Projeto pj

Como ilustração, seja a Figura 3.7, onde temos a instância do objeto complexo Projeto, pj. Se uma transação Ta1 tentar bloquear o objeto componente f em modo exclusivo utilizando o protocolo II de Gray, ela deverá bloquear toda a instância pj do objeto complexo Projeto em modo exclusivo, não permitindo que nenhuma outra transação bloqueie qualquer objeto componente de pj em nenhum modo.

Desta forma, vemos que este protocolo não vai permitir que duas transações bloqueiem em modos incompatíveis diferentes objetos componentes de uma mesma instância de um objeto complexo, o que impossibilita o conceito de unidades cooperativas encontrado no modelo MUC, onde várias transações precisam bloquear, ao mesmo tempo e em modo incompatível, vários subobjetos de um objeto complexo.

Entretanto, embora este protocolo não satisfaça às transações segundo o modelo MUC, alguns aspectos apresentados por ele poderão ser ainda aproveitados, como por exemplo, o bloqueio sobre hierarquias de classes que mantêm a relação "IS-A"

Vale ainda ressaltar que esta adaptação do protocolo de Gray para o bloqueio em estruturas de classes, aplica-se também a grafos cujas classes em seu interior mantêm entre si a relação "IS-A". Este caso mais geral não foi mostrado nesta seção em virtude do fato de que no modelo MUC, mostrado no capítulo 2 e no qual se baseia o protocolo que será apresentado no próximo capítulo, as classes são organizadas em hierarquias "IS-A".

A outra abordagem para a aquisição de bloqueios sobre uma hierarquia de classes, refere-se àquela proposta em [Garza-Kim 88] que será mostrada a seguir.

3.1.3 PROTOCOLO DE GARZA-KIM PARA HIERARQUIAS "IS-A" e *É-PARTE-DE.*

Esta abordagem, que também está baseada no protocolo de granulagens de bloqueio proposto em [Gray-78], está mais voltada para hierarquias de classes, capturando melhor a sua semântica, sendo subdividida em dois aspectos: aquisição/liberação de bloqueios em hierarquias de classes cujas relações entre elas é do tipo "IS-A"; e a aquisição/liberação de bloqueios em hierarquias de classes cujas relações entre elas é do tipo É-PARTE-DE.

Após uma discussão sobre o mecanismo de bloqueio sobre hierarquias "IS-A", iremos examinar a maior contribuição deixada por este protocolo, que foi a elaboração de um mecanismo de bloqueio sobre hierarquias de classes, cujas classes mantêm entre si a relação É-PARTE-DE, sendo o primeiro protocolo a tratar convenientemente deste tipo de hierarquias de classes.

HIERARQUIAS "IS-A"

Para o caso de aplicações envolvendo consultas ou mudanças de definição de classes próximas à classe-raiz de uma profunda hierarquia de classes "IS-A", foi proposto um protocolo de CC em [Garza-Kim-88]. Para este protocolo foram definidos dois novos modos de bloqueio que são: o modo R (Leitura em uma

hierarquia) e o modo W (Escrita em uma hierarquia). As semânticas associadas a estes novos modos de bloqueio, que serão mostradas adiante, deverão ser definidas com o objetivo de reduzir o número de bloqueios sobre a hierarquia "IS-A", por parte das transações.

O modo R em uma classe C de uma hierarquia, significa um bloqueio explícito em modo S na classe C, e bloqueios implícitos em modo S em todas as subclasses da classe C. Como exemplo, seja a hierarquia de classes "IS-A" mostrada na Figura 3.8. Se uma transação T colocar um bloqueio em modo R sobre a classe C, ela a bloqueará explicitamente em modo S e as classes D, E e F implicitamente em modo S.

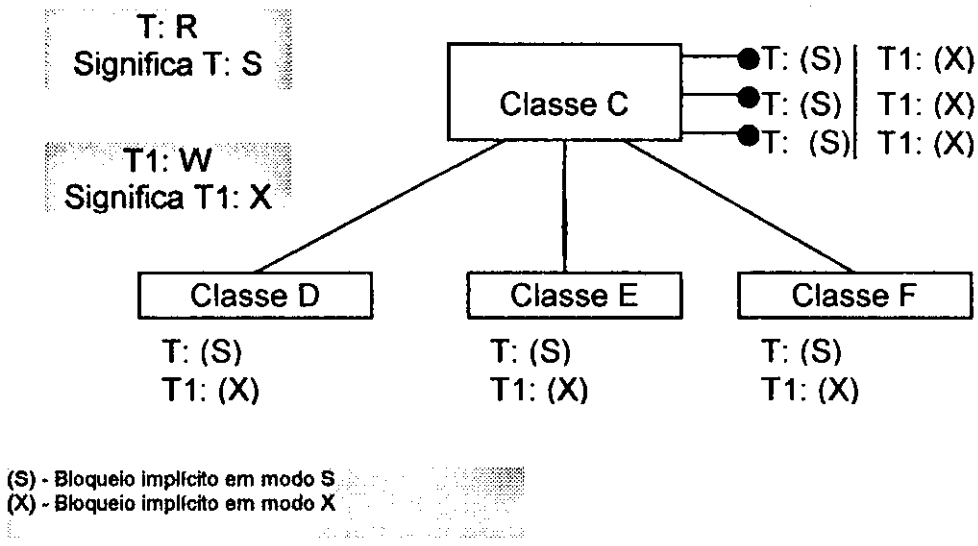


Figura 3.8 - Hierarquia "IS-A" cuja Classe-Raiz é a Classe C

O modo W em uma classe C, significa um bloqueio explícito em modo X na classe C, e implícitos bloqueios em modo X em todas as subclasses da classe C. Tomando-se novamente a Figura 3.8 como exemplo, se uma transação T1 bloquear a classe C em modo W, ela bloqueará a classe C explicitamente em modo X e as classes D, E e F implicitamente em modo X.

Note que o modo de bloqueio X, ao contrário do modo W, bloqueia apenas uma classe, significando por exemplo, que uma transação poderá atualizar as instâncias de uma classe. Com relação à definição da classe, esta não poderá ser atualizada com um bloqueio em modo X, sendo necessário o bloqueio em modo W sobre a classe.

Por sua vez, o modo de bloqueio S, ao contrário do modo R, e da mesma forma que o modo de bloqueio X também bloqueia apenas uma classe em uma hierarquia.

As instâncias de uma classe, por sua vez, poderão apenas serem bloqueadas tanto no modo compartilhado (S) como no modo exclusivo (X).

Finalmente, para se colocar um bloqueio em uma classe em modo W ou R pertencente a uma hierarquia de classes "IS-A", deverão ser colocados bloqueios em modo IR ou IW sobre a seqüência de superclasses da classe correspondente. Além disso, todos os modos de bloqueio discutidos nas seções anteriores também irão exigir, de forma semelhante, os modos de bloqueios IW e IR sobre as suas correspondentes superclasses.

Vamos mostrar finalmente o protocolo de Garza-Kim para hierarquias de classes do tipo "IS-A".

PROTOCOLO DE GARZA-KIM PARA HIERARQUIA DE CLASSES "IS-A".

1 - Para o bloqueio de uma classe em modo R ou W, deverão ser colocados bloqueios em modo IR ou IW respectivamente sobre a seqüência de superclasses da classe.

2 - Para o bloqueio de uma classe nos modos S ou IS, deverão ser colocados bloqueios no modo IR sobre a seqüência de superclasses da classe.

3 - Para o bloqueio de uma classe nos modos X ou IX, deverão ser colocados bloqueios no modo IW sobre a seqüência de superclasses da classe.

4 - Os bloqueios serão liberados ao término da transação em qualquer ordem.

Como ilustração desta parte do protocolo proposto por Garza-Kim para hierarquias do tipo "IS-A" e considerando ainda a estrutura mostrada na Figura 3.5, temos os seguintes casos que serão mostrados a seguir

EXEMPLO 1: Uma transação T quer selecionar todas as instâncias da classe Funcionário e suas subclasses.

(a) - Bloqueia a classe Funcionário em modo R e as superclasses da classe Funcionário sobre a seqüência de superclasses da classe em modo IR.

Este modo de bloqueio R sobre a classe Funcionário pela transação T, significa o mesmo que um bloqueio explícito em modo S sobre a classe Funcionário e bloqueios implícitos em modo S sobre as subclasses de Funcionário.

EXEMPLO 2: Uma transação T1 quer mudar a definição da classe Funcionário.

(a) - Bloqueia a classe Funcionário em modo W e as superclasses da classe Funcionário sobre a seqüência de superclasses em modo IW.

Por sua vez, este modo de bloqueio W sobre a classe Funcionário pela transação T1, significa o mesmo que um bloqueio explícito em modo X sobre a classe Funcionário e bloqueios implícitos em modo X sobre as subclasses de Funcionário.

EXEMPLO 3: Uma transação T2 quer selecionar instâncias da classe Estudante tal que ...

(a) - Bloqueia as superclasses da classe Estudante, Pessoa, em modo IR.
(b) - Bloqueia a classe Estudante em modo IS
(c) - Bloqueia as instâncias selecionadas de Estudante em modo S.

EXEMPLO 4: Supondo que uma transação T1 esteja modificando a definição da classe Funcionário, enquanto que uma outra transação T2 tenta modificar a definição da classe Pessoa. Os bloqueios por parte destas duas transações são mostrados a seguir:

T1	**T2**
IW na classe Pessoa.	Tenta colocar W na classe Pessoa,
W na classe Funcionário.	mas terá de esperar por T1.

De acordo com [Kim-90], a desvantagem deste protocolo é que se uma transação for bloquear apenas uma classe, ela terá que bloquear todas as suas superclasses. Também, mesmo quando uma classe C está para ser bloqueada nos modos IX, IS, S ou X, suas superclasses terão de ser bloqueadas nos modos IR ou IW. Isto não é desejável se a classe C estiver próximo ao último nível de uma hierarquia com um grande número de níveis, acarretando um *overhead* para o "gerenciador de bloqueios".

HIERARQUIA É-PARTE-DE

Muitas aplicações exigem a capacidade de definir e manipular um conjunto de objetos como uma única entidade lógica, para propósitos de integridade semântica, e para uma maior eficiência na recuperação e armazenagem dos objetos [Lori83,IEEE85,Kim87].

Um objeto complexo é uma coleção de objetos heterogêneos que mantêm entre si a relação É-PARTE-DE.

Com relação a protocolos de controle de concorrência para objetos complexos, o bloqueio sobre uma instância de um objeto complexo deverá comprometer o mínimo possível o bloqueio de outras instâncias que são baseadas na mesma hierarquia deste objeto, por parte de outras transações dentro do sistema.

A parte do protocolo de Garza-Kim para hierarquias É-PARTE-DE considera uma instância de um objeto complexo como uma única entidade lógica a ser bloqueada por uma transação, de modo a evitar que uma segunda transação venha bloquear algum objeto que seja componente da mesma.

O protocolo suporta o bloqueio implícito de todos os objetos componentes de um objeto bloqueado explicitamente, permite que diferentes transações acessem

concorrentemente outros objetos complexos que estão baseados na mesma hierarquia É-PARTE-DE, mas que não compartilhem qualquer objeto componente.

Finalmente, uma instância de uma classe que pertence a hierarquia do objeto complexo, e que não faz parte do objeto complexo que está sendo bloqueado por uma transação, poderá vir a ser bloqueada por outra transação em algum modo que não seja incompatível com os modos de bloqueio colocados por parte da primeira.

A parte do protocolo de [Garza-Kim-88] para hierarquias É-PARTE-DE adiciona três novos modos de bloqueio que são : ISO, IXO e SIXO, correspondendo aos modos IS, IX e SIX, respectivamente. As semânticas associadas a estes novos modos de bloqueio foram definidas com o objetivo de garantir a integridade dos objetos complexos diante do acesso concorrente de diferentes transações. A matriz de compatibilidade mostrada na Figura 3.9, define a semântica associada aos modos de bloqueio utilizados no protocolo, onde s é "sim" significando a compatibilidade entre os modos de bloqueios e n é "não", significando a incompatibilidade entre os modos de bloqueios.

A idéia chave do protocolo de Garza-Kim, é forçar o modo ISO conflitar com o modo IX, e os modos IXO e SIXO conflitarem com os modos IS e IX. Então, para bloquear um objeto complexo, a classe-raiz será bloqueada, como antes, em modo IS, IX, S, SIX ou X. Contudo, cada uma das classes componentes da hierarquia será agora bloqueada nos modos ISO, IXO, S, SIXO, ou X respectivamente.

	IS	IX	S	SIX	X	ISO	IXO	SIXO
IS	s	s	s	s	n	s	n	n
IX	s	s	n	n	n	n	n	n
S	s	n	s	n	n	s	n	n
SIX	s	n	n	n	n	n	n	n
X	n	n	n	n	n	n	n	n
ISO	s	n	s	n	n	s	s	s
IXO	n	n	n	n	n	s	s	n
SIXO	n	n	n	n	n	s	n	n

Figura 3.9 Matriz de Compatibilidade - Protocolo de Garza-Kim

A seguir, vamos mostrar a parte do protocolo de Garza-Kim que trata de hierarquias É-PARTE-DE.

PROTOCOLO DE GARZA-KIM PARA HIERARQUIA DE CLASSES É-PARTE-DE.

1 - Para se bloquear um objeto complexo em modo exclusivo, X, se deverá bloquear a classe-raiz em modo IX ou SIX, e então bloquear todas as classes componentes da hierarquia em modo IXO ou SIXO respectivamente.

2 - Para se bloquear um objeto complexo em modo compartilhado, S, se deverá bloquear a classe-raiz em modo IS e então bloquear todas as classes componentes da hierarquia em modo ISO.

3 - Os bloqueios serão liberados ao término da transação em qualquer ordem.

Para melhor exemplificar este protocolo, vai-se considerar os seguintes exemplos que irão se basear na Figura 3.10 :

EXEMPLO 1: Selecione o objeto complexo Projeto, id1, tal que ...

(a) - Bloqueie a classe Projeto em modo IS.
(b) - Bloqueie a instância id1 em modo S.
(c) - Bloqueie as classes Contrato, Planta, Fachada e Interior em modo ISO.

Implicitamente, as instâncias c1, p1, f1 e i1 são bloqueadas em modo S. Portanto, a transação bloqueará em modo S todo o objeto complexo cuja raiz é id1, e nenhuma outra transação poderá atualizar o objeto id1, ou parte dele. Em compensação, uma outra transação poderia atualizar o objeto id2 concorrentemente.

EXEMPLO 2: Atualize todos os projetos e seus componentes tal que ...

(a) - Bloqueie a classe Projeto em modo IX.
(b) - Bloqueie as instâncias selecionadas de Projeto em modo X.
(c) - Bloqueie as classes componentes da hierarquia em modo IXO.

EXEMPLO 3: Suponha que uma transação T1 bloqueie em modo exclusivo a instância id1 do objeto complexo Projeto e que uma outra transação T2 bloqueia em modo exclusivo a instância id2 do objeto complexo Projeto. O bloqueio por parte destas duas transações são mostrados abaixo.

T1	T2
IX na classe Projeto	IX na classe Projeto
X na instância id1 de Projeto.	X na instância id2 de Projeto
IXO nas classes Contrato, Planta, Fachada e Interior	IXO nas classes Contrato, Planta, Fachada e Interior.

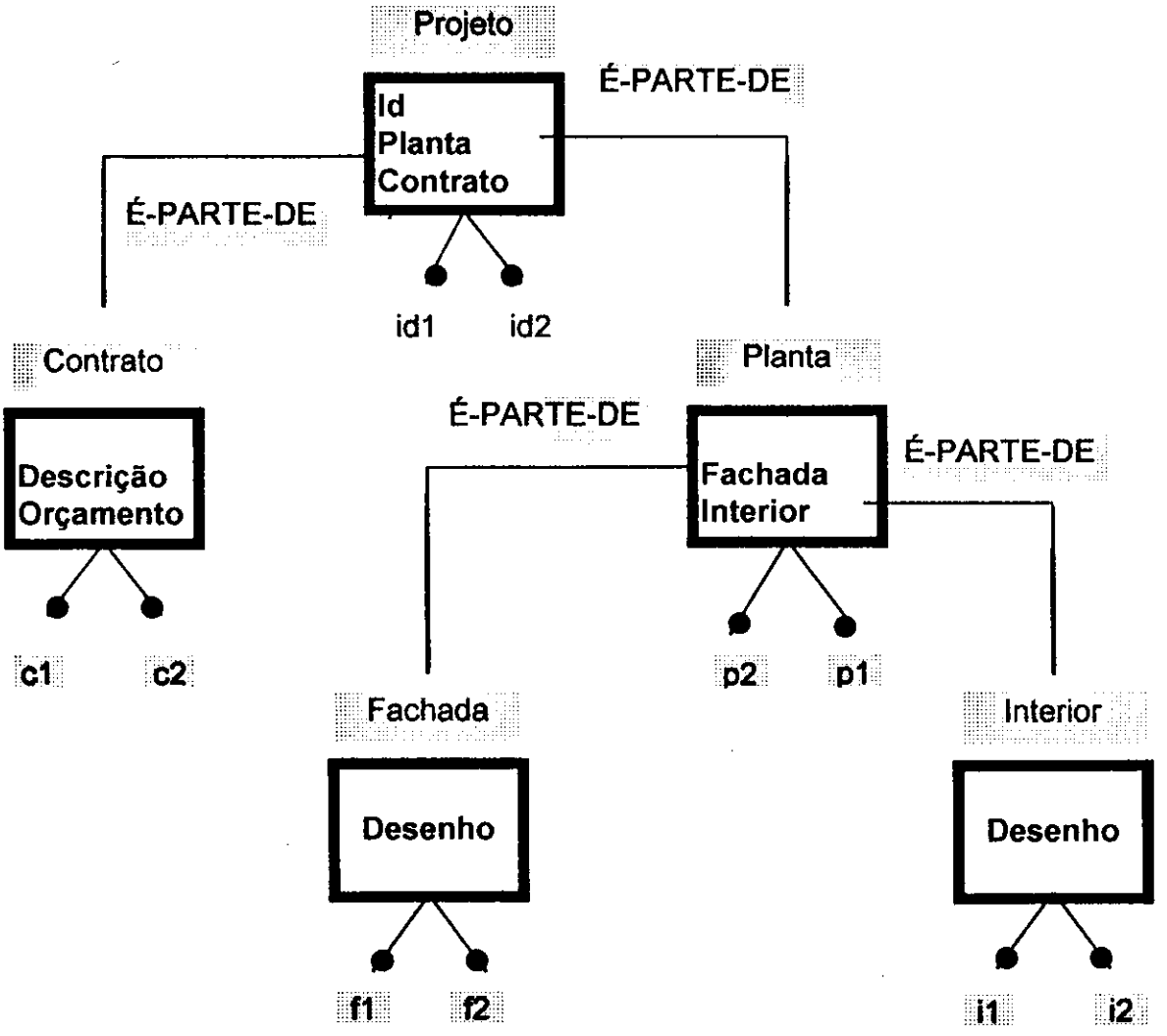


Figura 3.10 - O Objeto Complexo Projeto.

Este protocolo permite que várias transações leiam e atualizem diferentes objetos complexos que possuem a mesma classe-raiz de uma hierarquia É-PARTE-DE. Entretanto, se houver alguma transação executando uma operação de leitura de um objeto complexo, não poderá haver outra transação atualizando qualquer instância de uma classe componente desta hierarquia, assim como, se houver alguma transação executando uma operação de leitura sobre instâncias de uma classe componente da hierarquia É-PARTE-DE, não poderá haver outra transação executando uma operação de atualização em qualquer objeto complexo pertencente à hierarquia.

ANÁLISE DO PROTOCOLO

O Protocolo de Garza-Kim é baseado em bloqueios tanto em hierarquias de classes "IS-A", quanto em hierarquias de classes É-PARTE-DE. Isto é de grande importância para as aplicações que atuam sobre um banco de dados com o modelo de dados orientado a objetos.

Em relação ao modelo MUC, objeto desta dissertação de mestrado, o protocolo de Garza-Kim seria aplicável se não existissem dois grandes empecilhos:

- O Protocolo só funciona com transações *planas* ou de um só nível. Relembremos que, no MUC, as unidades cooperativas são transações *não-planas*, ou *hierárquicas* (vários níveis).

- O Protocolo não permite que duas ou mais transações atualizem um mesmo objeto complexo simultaneamente. No entanto, para que duas transações possam cooperar, no interior de uma unidade cooperativa, é primordial que elas possam atualizar simultaneamente partes de um mesmo objeto complexo.

Entretanto, as idéias contidas no protocolo sobre como bloquear, de modo eficiente, hierarquias de classes "IS-A" e É-PARTE-DE, nos foram de muita valia para a elaboração do nosso protocolo, que será discutido no próximo capítulo.

Devemos também ressaltar que a abordagem de Garza-Kim aplica-se também a grafos "IS-A" e É-PARTE-DE, com ligeiras modificações. Estes casos mais gerais não foram mostrados nesta seção em virtude de que, no modelo MUC, no qual se baseia o nosso protocolo, as classes são organizadas somente em hierarquias "IS-A" ou É-PARTE-DE.

Na próxima seção, discutiremos um protocolo de controle de concorrência de transações hierárquicas.

3.2 O PROTOCOLO DE CONTROLE DE CONCORRÊNCIA DE TRANSAÇÕES HIERÁRQUICAS

Para ambientes que envolvem aplicações estruturadas para ambientes distribuídos, o modelo convencional de transações mostra-se pouco adequado, visto que ele não satisfaz às principais exigências destes tipos de aplicações como por exemplo, suporte ao trabalho cooperativo entre transações, suporte à execução de transações estruturadas sobre objetos complexos, etc. Como conseqüência, um novo modelo de transações que atende às principais exigências destas classes de aplicações, junto com um protocolo de controle de concorrência e um protocolo de recuperação, foi proposto em [Moss-85].

Neste modelo, também chamado de *Modelo de Transações Aninhadas* ou *Modelo de Transações Hierárquicas* qualquer transação poderá ter *subtransações*, e estas, por sua vez, podem também recursivamente ter *subtransações*, constituindo assim uma *árvore de transações*, onde a transação de mais alto nível é chamada *de transação-raiz*. Uma subtransação deverá ser iniciada após o início de sua *transação-mãe* e o seu término deverá acontecer antes do término da transação-mãe.

Para a árvore de transações, a terminologia utilizada é a padrão para árvores em geral, tais como mãe, filho, ancestral, descendente, etc. Por conveniência, Moss define uma transação como sendo descendente e ancestral dela própria, de modo

que quando desejarmos nos referir aos descendentes (ancestrais) de uma transação, excluindo ela própria, nós utilizaremos o termo inferior (superior). Desta forma, uma transação *plana* também pode ser vista como um caso muito particular de uma árvore de transações.

Na Figura 3.11 abaixo, é mostrada uma árvore de transações, onde T1 é a transação-raiz. Nesta árvore, as transações filhas da transação T4 são as transações T8, T9 e T10. A transação-mãe de T4 é a transação T2. As transações inferiores de T4 são as transações T8, T9 e T10. As transações superiores são T1 e T2. Transações não descendentes de T4 são as transações T1, T2, T3, T5, T6 e T7.

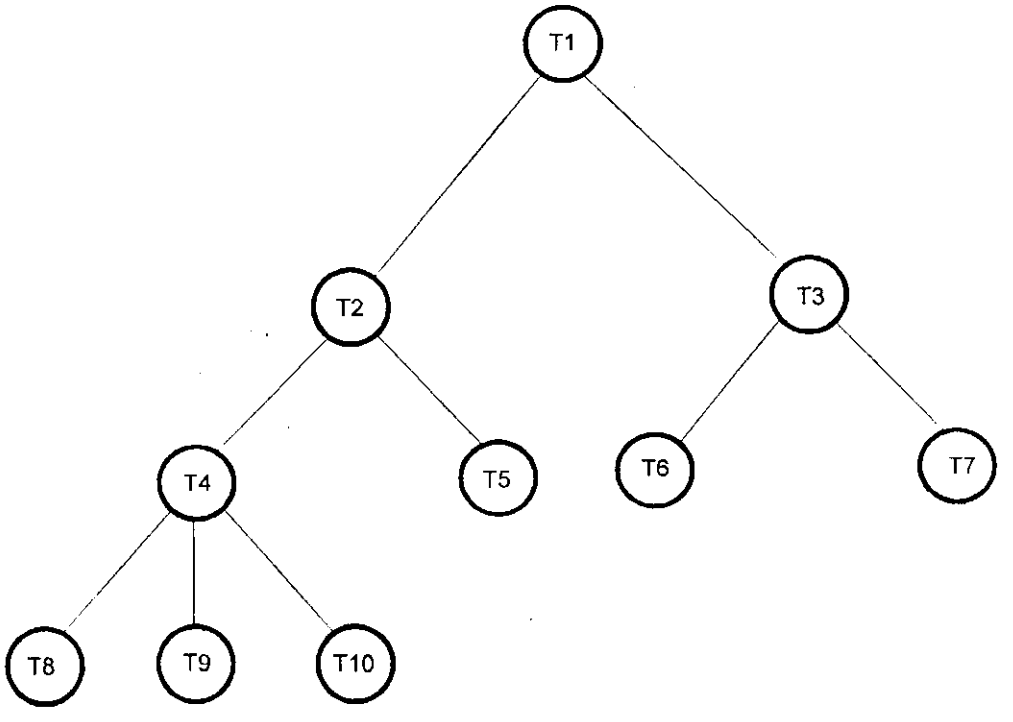


Figura 3.11 A Transação Hierárquica T1

Esta seção irá se deter apenas nos aspectos relativos à sincronização de transações no interior de uma árvore de transações e entre árvores de transações, mostrando para isso um protocolo de controle de concorrência para transações hierárquicas. As transações de mais alto nível (transação-raiz) são sincronizadas segundo o modelo

de transações convencionais. Além disso, as subtransações serão também sincronizadas no interior de uma árvore de transações.

A transação-raiz de uma árvore de transações define uma unidade ACID de trabalho. Desta forma, essas transações executam de forma isolada em relação a outras árvores de transações. Já com relação às transações no interior de uma árvore de transações, deverão ser preservadas as suas propriedades de atomicidade e isolamento. Com relação à propriedade de durabilidade de subtransações, esta deverá ser condicionada à durabilidade das transações superiores.

Por exemplo, tomando como base a Figura 3.11, a transação T1 irá definir uma unidade ACID de trabalho, de modo que transações externas a ela não poderão visualizar os seus efeitos até que ela finalize os seus trabalhos. A transação T2, que é uma subtransação (transação filha) de T1, deverá apresentar as propriedades de atomicidade e isolamento. A durabilidade de seus efeitos é condicionada à durabilidade dos efeitos de sua transação-mãe, que é T1.

Com respeito à sincronização de subtransações que estão em um mesmo nível no interior de uma árvore de transações, e que possuem as mesmas transações superioras, estas deverão ser sincronizadas como se cada uma delas fosse a própria transação-raiz no seu próprio universo, ou seja, o universo proporcionado por suas transações-mães. Desta forma, o acesso concorrente aos dados pertencentes ao universo dessas transações, ou seja, que são compartilhados por elas, serão realizados de forma segura. Já com relação aos bloqueios que uma transação-mãe possui, esta não irá interferir com qualquer uma de suas subtransações, apenas interferindo com as suas transações-irmãs, com as quais compartilham o acesso aos dados pertencentes ao universo de sua transação-mãe. Isto significa que uma transação irá ter acesso irrestrito aos objetos obtidos por suas transações ancestrais.

Como exemplo deste mecanismo de herança de bloqueios, e novamente tomando como base a Figura 3.11, vamos supor que a transação T4, tendo como subtransações as transações T8, T9 e T10, detenha um bloqueio sobre um

determinado objeto. Desta forma, essas suas subtransações terão também acesso irrestrito a este objeto, pois a transação-mãe, T4, não irá interferir com elas.

Duas subtransações que estão intimamente relacionadas no contexto da aplicação, poderão ser processadas concorrentemente sob a mesma transação-mãe, sem que haja riscos de surgirem problemas caso elas às vezes (ou sempre) acessem alguns dos mesmos itens de dados. Por ausência de riscos, queremos dizer que se uma das subtransações acessar um objeto em modo exclusivo, as demais transações irmãs irão ter que esperar pelo seu término e disputar o bloqueio.

O protocolo de bloqueio de árvores de transações proposto em [Moss-85] é uma extensão do protocolo clássico de bloqueio de transações convencionais. Será mostrado primeiramente o protocolo para transações convencionais e, posteriormente, o protocolo para transações aninhadas proposto por Moss.

Temos aqui então, o protocolo para bloqueios dos tipos Compartilhado/Exclusivo sobre os objetos por parte de transações convencionais (transações ACID), chamado de Protocolo de Bloqueio em Duas Fases.

Regra 1: Uma transação poderá ler um objeto se, e somente se, ela detém o bloqueio correspondente no modo Compartilhado sobre este objeto.

Regra 2: Uma transação poderá escrever (modificar) um objeto se, e somente se, ela detém o bloqueio correspondente em modo Exclusivo sobre este objeto.

Regra 3: Uma transação poderá deter um bloqueio em modo Compartilhado sobre um objeto se, e somente se, nenhuma outra transação possui o bloqueio em modo Exclusivo.

Regra 4: Uma transação poderá deter um bloqueio em modo Exclusivo sobre um objeto se, e somente se, nenhuma outra transação possui bloqueio em qualquer modo sobre o mesmo.

Regra 5: Uma transação não liberará qualquer bloqueio, até que a transação esteja completa.

Finalmente, vamos apresentar o protocolo de Moss para bloqueios por parte de transações aninhadas.

O Protocolo de Moss faz a suposição simplificadora de que apenas as transações que estão no último nível da hierarquia (transações que não possuem subtransações ou transações filhas) poderão manipular os objetos diretamente.

Regra 1: Uma transação poderá ler um objeto se, e somente se, ela detém o bloqueio no modo Compartilhado sobre este objeto.

Regra 2: Uma transação poderá escrever (modificar) um objeto se, e somente se, ela detém o bloqueio correspondente em modo Exclusivo sobre este objeto.

Regra 3: Uma transação poderá possuir um bloqueio em modo Compartilhado sobre um objeto se, e somente se, todos as transações proprietárias de bloqueios em modo Exclusivo sobre este objeto são ancestrais da transação requisitante.

Regra 4: Uma transação poderá possuir um bloqueio em modo Exclusivo sobre um objeto se, e somente se, todas as transações proprietárias de bloqueios em qualquer modo sobre este objeto são ancestrais da transação requisitante.

Regra 5: Quando uma transação termina definitivamente (*Commit*), sua transação-mãe herdará então os bloqueios da transação filha. Se a transação-mãe já detém os bloqueios, ela irá então deter o bloqueio no modo que for o mais exclusivo dos modos da transação-mãe e da transação filha que terminou. Quando uma transação aborta, seus bloqueios serão descartados.

Para melhor ilustrar a utilização do modelo de transações aninhadas, iremos mostrar aqui um exemplo prático de sua aplicação, onde uma transação T1 detém um bloqueio em modo exclusivo sobre uma instância id, do objeto complexo Projeto (ver Figura 3.12), que por sua vez é constituída pelos objetos componentes c, p(f, i), que

são instâncias respectivamente das classes Contrato, Planta, Fachada e Interior da Figura 3.12. Desta forma, T1 detém um bloqueio explícito sobre cada objeto componente do objeto complexo id, de modo que, se alguma outra transação T2 quiser deter um bloqueio em modo exclusivo sobre o objeto c por exemplo, ela terá de ser uma subtransação da transação T1, ou seja, ela terá que pertencer à árvore de transações enraizada em T1. De forma similar, se uma transação T3 quiser deter um bloqueio em modo exclusivo sobre o objeto componente p, ela também terá de ser uma subtransação da transação T1, ou seja, ela terá de pertencer à árvore de transações enraizada em T1.

Desta forma, seguindo o protocolo de Moss, para a sincronização de transações aninhadas, teremos a árvore de transações T1(T2,T3(T4,T5)), também mostrada na Figura 3.12.

Como ilustração da sincronização de transações no interior de uma árvore de transações, se a transação T5 acessa em modo exclusivo o objeto componente i, a sua transação irmã, T4, não poderá acessar este objeto enquanto T5 detiver um bloqueio sobre o objeto i.

Quando a transação T5 termina definitivamente, ela libera o bloqueio que ela detinha sobre o objeto i, sendo este herdado por sua transação-mãe, que é a transação T3. Entretanto, os seus efeitos só serão feitos permanentes após o término de suas transações superioras.

Desta forma, se a transação T4, que é uma subtransação de T3, quiser bloquear o objeto i em qualquer modo, isto será possível pois T3 e T4 não irão interferir uma com a outra já que T4 é uma subtransação de T3, isto é, T3 é uma transação ancestral de T4.

Por outro lado, se uma transação, que é externa à árvore de transações, quiser bloquear algum objeto componente da instância do objeto complexo id, em qualquer modo, isto não será possível, pois, para transações que são externas, a árvore de transações T1(T2,T3(T4,T5)) define uma unidade ACID de trabalho.

Como conseqüência, a transação externa terá que esperar o término definitivo da árvore de transações para acessar algum objeto componente do objeto complexo id.

Esta árvore de transações reflete a estrutura do objeto complexo id, que está sendo atualizado por ela, embora a similaridade do objeto complexo e da transação hierárquica que o manipula, não seja obrigatória.

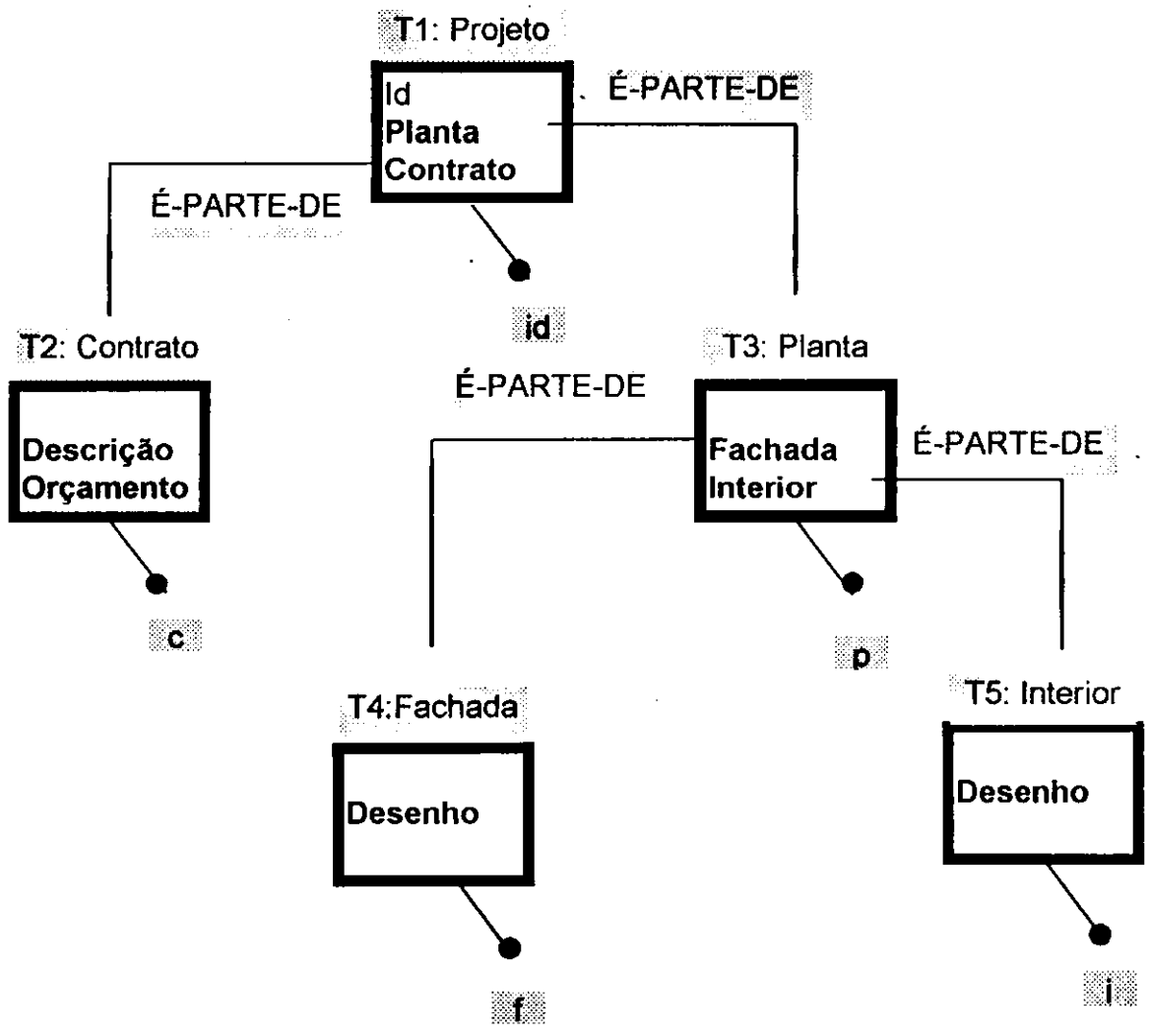


Figura 3.12 O Objeto Complexo Projeto

Vemos finalmente, que o modelo de transações aninhadas forma uma boa base para a construção de aplicações em sistemas distribuídos, pois ele consegue encapsular

de uma forma muito clara os aspectos relativos à sincronização e às propriedades de falhas, cujos aspectos não foram considerados em nosso estudo.

ANÁLISE DO PROTOCOLO DE MOSS

O protocolo de controle de concorrência de transações hierárquicas proposto em [Moss-85] é um protocolo de sincronização da aquisição de bloqueios por parte das subtransações no seio de uma transação hierárquica. Desta forma, este protocolo se mostra adequado a ambientes onde existem atividades que desenvolvem sub-atividades, como em ambientes de desenvolvimento de projetos de engenharia, de arquitetura etc. A principal contribuição deste protocolo é o seu mecanismo de herança e transferência de bloqueios entre transações. Sua idéia de herança nos foi de grande valia para o nosso protocolo de controle de concorrência de transações cooperativas que estão segundo o modelo MUC.

Como, segundo este protocolo, as transações irmãs são isoladas umas das outras (serializáveis), ele irá melhor se adequar às características das unidades cooperativas fracas, onde as transações que cooperam em seu interior serão serializáveis. Quanto ao caso de unidades cooperativas fortes, onde as transações que cooperam em seu interior são não-serializáveis, ele não se mostrará adequado.

Além disso, este protocolo não se aplica a modelos de dados orientados a objetos. Mais precisamente, ele não suporta a noção de bloqueio em hierarquias de classes "IS-A" e É-PARTE-DE.

Vemos, portanto, que este protocolo satisfaz apenas algumas poucas exigências que são necessárias para um protocolo de controle de concorrência de transações que cooperam segundo o modelo MUC proposto em [Sampaio-95].

3.3 ANÁLISE GERAL DOS PROTOCOLOS APRESENTADOS

Neste capítulo, nós fizemos a apresentação das principais características de protocolos de controle de concorrência existentes na literatura para aplicações ditas não-convencionais, que de uma forma ou de outra contribuíram para a elaboração do nosso protocolo de controle de concorrência de transações cooperativas que estão segundo o modelo MUC, apresentado no capítulo 2.

O Protocolo de Gray e o Protocolo II de Gray, contribuíram principalmente com a noção de diferentes unidades de bloqueio (granulagem de bloqueio), dentro de um mesmo sistema. Com relação ao protocolo II de Gray, ele apresentou um mecanismo para a aquisição de bloqueios sobre hierarquias "IS-A", enquanto que com relação a hierarquias É-PARTE-DE ele não tratou convenientemente deste tipo de hierarquia. Já o protocolo de Garza-Kim, que é uma extensão do Protocolo II de Gray, teve a sua grande contribuição na elaboração de um mecanismo para a aquisição de bloqueios sobre hierarquias É-PARTE-DE.

Por último, o protocolo proposto por Moss veio inovar com relação à classe de transações à qual ele se aplica: transações hierárquicas. Até então na literatura, não havia um protocolo que suportasse transações multiníveis. Como as transações do modelo MUC são multiníveis, as suas características se aproximam bastante daquelas do modelo de Moss, no que diz respeito a herança de bloqueios.

Como não existe na literatura um protocolo que satisfaça todas as características e exigências das transações cooperativas do modelo MUC, foi elaborado um novo protocolo de transações cooperativas que será mostrado no próximo capítulo, e que é o objetivo central desta dissertação de mestrado.

4. PROTOCOLO DE CC DE TRANSAÇÕES COOPERATIVAS SEGUNDO O MODELO MUC

Nos bancos de dados que adotam um modelo de dados orientado a objetos, as classes são estruturadas por relações do tipo "IS-A", É-PARTE-DE, É-DOMINIO-DE etc.

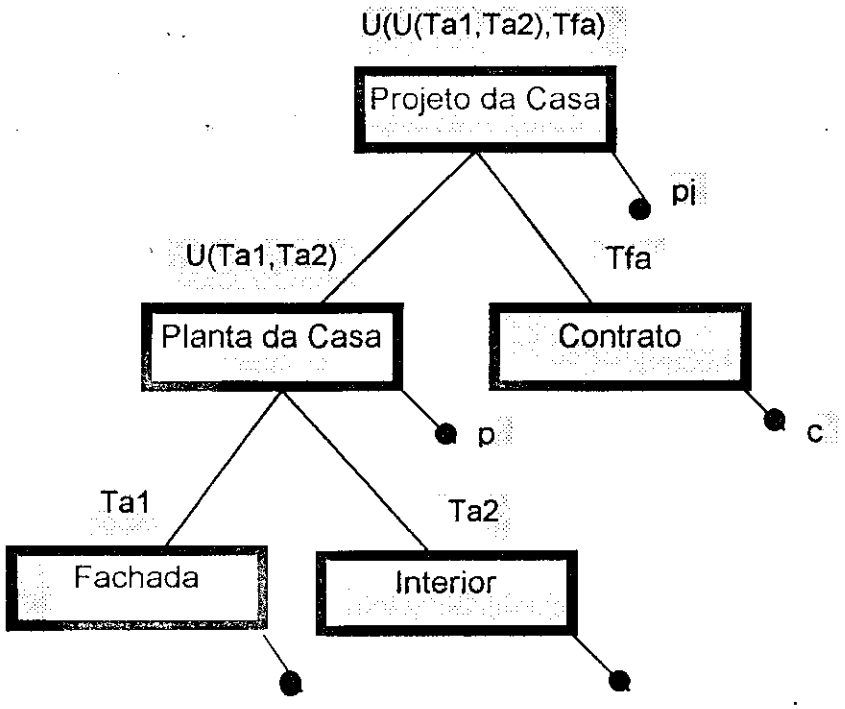
Um protocolo de controle de concorrência que seja baseado em bloqueios, para ser adotado por transações que atuam sobre um banco de dados orientado a objetos, terá que necessariamente abranger técnicas de bloqueio sobre as principais estruturas existentes nesses bancos de dados.

Particularmente, para o caso de um ambiente de aplicações sobre um banco de dados orientado a objetos, temos o caso das aplicações cooperativas que envolvem uma equipe de pessoas trabalhando cooperativamente, sendo que essas pessoas, cada uma representada por uma transação cooperativa, interagem direta ou indiretamente a fim de obter a atualização de um conjunto maximal de objetos. Cada uma das transações é responsável pela atualização de apenas uma parte do conjunto maximal de objetos. Portanto, estamos falando aqui de cooperação entre transações segundo o modelo MUC, que foi o tema do capítulo 2.

Mais especificamente, considere, por exemplo, uma equipe de arquitetos em um escritório de arquitetura dividindo a tarefa de projetar a casa de um cliente. O Projeto da Casa é composto pelos objetos Planta da Casa e Contrato. Por sua vez, o objeto

Planta da Casa é formada pelos objetos Fachada e Interior. Planta da Casa e Contrato, da mesma forma que Fachada e Interior, mantêm entre si uma relação mútua de dependência, ou seja, a atualização em um desses objetos pode ter conseqüências sobre o outro. Suponha então que um arquiteto, representado por uma transação Ta1, está atualizando uma instância de Fachada, f, enquanto que um outro arquiteto, representado por outra transação Ta2, está atualizando uma instância de Interior, i. Uma das transações poderá influenciar o resultado da outra, interagindo ou cooperando quando necessário.

De maneira semelhante, seja um funcionário do setor administrativo, representado pela transação Tfa, que está atualizando uma instância de Contrato, c, de modo que esta transação irá também interagir com a transação U(Ta1,Ta2), mostrada na Figura 4.1, representando a subunidade cooperativa dos arquitetos. Esta figura mostra o objeto complexo Projeto da Casa junto com a árvore de transações que o atualiza, ou seja, a unidade cooperativa U(U(Ta1,Ta2),Tfa)).



Como visto no capítulo 3, os principais protocolos de controle de concorrência encontrados na literatura para aplicações não-convencionais não suprem todos os requisitos para a sincronização das transações no interior de uma unidade cooperativa e para a sincronização das unidades cooperativas com as transações que lhe são externas.

Diante deste cenário, será proposto um novo protocolo de controle de concorrência que é baseado em bloqueios que suporta todas as exigências do modelo MUC, visto no capítulo 2.

Estas exigências, que serão detalhadas na próxima seção são as seguintes: bloqueio a nível de objetos e a nível de atributos de objetos; bloqueio em uma hierarquia de classes "IS-A"; bloqueio em uma hierarquia de classes É-PARTE-DE; Mecanismo de transferência e herança de bloqueios no interior de uma unidade cooperativa. Vamos ver cada uma delas com maiores detalhes na próxima seção.

Iremos primeiramente sumarizar estas novas exigências para o controle de concorrência das transações cooperativas segundo o modelo MUC. Posteriormente, descreveremos o protocolo de controle de concorrência dessas transações junto com exemplos que possibilitam uma melhor compreensão do mesmo.

4.1 - AS NOVAS EXIGÊNCIAS PARA O CONTROLE DE CONCORRÊNCIA.

Técnicas de bloqueio nos ambientes de aplicações cooperativas cujas transações cooperam segundo o modelo MUC, proposto em [Sampaio-95], necessitam de quatro novas importantes exigências.

- ***EXIGÊNCIA 1: Bloqueio a nível de objeto e a nível de atributo de objeto.***

Para permitir que transações que estão no interior de uma unidade cooperativa (que poderá ser fraca ou forte conforme estas transações em seu interior cooperem fracamente ou fortemente), compartilhem a atualização de um mesmo objeto complexo, as transações segundo o modelo proposto em [Sampaio-95] apresentam um forte argumento para bloqueios a nível de atributos de objetos (equivalente ao nível de colunas em banco de dados relacionais). Mais precisamente, o bloqueio poderá ser tanto a nível de objetos simples inteiros (equivalente ao nível de linhas em banco de dados relacionais) quanto a nível de atributos de objetos. Observemos que em bancos de dados orientados a objetos "convencionais", como o ORION [Kim-90b], a menor *granulagem* de bloqueio ocorre a nível de objetos ao invés de a nível de atributos de objetos.

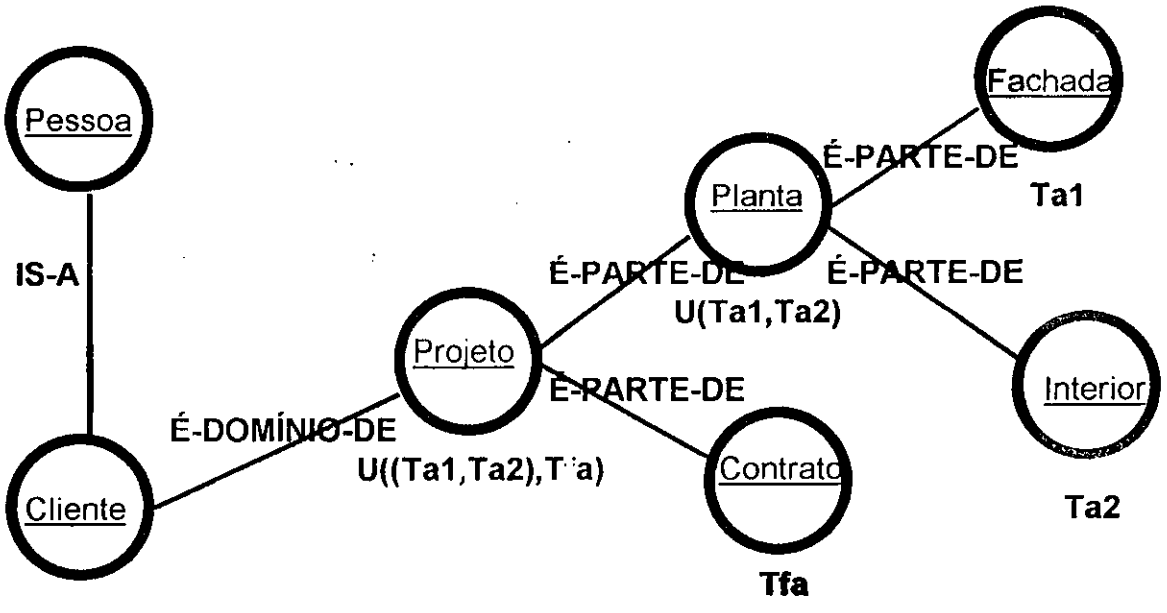


Figura 4.2 Esquema de um Banco de Dados de Projeto.

Tomemos como exemplo a consulta SQL-Like a seguir, executada sobre o esquema de banco de dados mostrado na Figura 4.2. Nesta consulta, apenas os objetos planta (incluindo os objetos fachada e interior) de cada objeto projeto de cliente_A com

orçamento maior que R$ 50.000 serão bloqueados (bloqueio do tipo leitura, neste caso). O objeto planta é um atributo do atributo projeto de cliente_A. A linguagem de consulta utiliza a especificação de predicados sobre uma seqüência aninhada de atributos, devido à hierarquia É-PARTE-DE.

CONSULTA

SELECT EACH projeto.planta

FROM Cliente

WHERE nome = "cliente_A" AND

projeto.contrato.orçamento >= 50.000

No mesmo instante, uma outra transação poderia sem problema algum atualizar o atributo endereço da instância de cliente_A

Como um outro exemplo, seja a unidade cooperativa dos arquitetos e do funcionário administrativo, para atualizar um dos projetos de cliente_A. Dado um projeto de cliente_A, em um determinado instante, o atributo fachada da planta do projeto deve ser bloqueado por Ta1, ao passo que o atributo interior da planta do projeto deve ser bloqueado por Ta2, enquanto que o atributo contrato do projeto deve ser bloqueado por Tfa, sendo todos os bloqueios do tipo exclusivo, sobre a mesma instância de um objeto complexo, um dos projetos de cliente_A.

- ***EXIGÊNCIA 2: Bloqueio em uma Hierarquia "IS-A".***

Como em todo banco de dados orientado a objetos, a existência de hierarquia de classes cujas relações entre as classes é do tipo "IS-A", irá exigir uma outra mudança nas técnicas convencionais de bloqueio. Enquanto uma transação acessa instâncias ou atributos de instâncias de uma classe, uma outra transação não poderá modificar a definição da classe ou de qualquer uma de suas correspondentes superclasses, pois uma modificação na definição de uma classe irá ter conseqüências em todas as suas subclasses correspondentes.

Como exemplo, enquanto a consulta anterior estiver sendo processada (ou enquanto a correspondente transação está em execução), as definições das classes Pessoa e Cliente não poderão ser modificadas por qualquer outra transação.

- ***EXIGÊNCIA 3: Bloqueio em uma Hierarquia É-PARTE-DE***

Como as classes às quais os objetos componentes de um objeto complexo pertencem são organizadas em uma hierarquia É-PARTE-DE, este tipo de hierarquia introduz o terceiro tipo de modificação na forma de bloquear objetos. Enquanto uma transação estiver manipulando um (sub-)objeto complexo, as definições das classes componentes da hierarquia do objeto complexo não poderão ser modificadas por qualquer outra transação.

Para ilustrar, enquanto a consulta anterior estiver sendo processada (ou enquanto a correspondente transação está em execução), as definições das classes Projeto, Planta, Contrato, Fachada e Interior não poderão ser modificadas por qualquer outra transação.

- ***EXIGÊNCIA 4: Transferência de Bloqueios no Interior de uma Unidade Cooperativa***

Finalmente, para tornar os seus efeitos visíveis a transações no interior de sua unidade cooperativa, uma transação cooperativa irá validar seletivamente, isto é, uma transação externa à unidade cooperativa não pode interferir entre as transações "selecionantes" e as transações "selecionadas". Para o caso de cooperação forte, a validação seletiva de uma transação não a finalizará necessariamente, visto que, uma das transações "selecionadas" ainda poderá questionar os efeitos da transação "selecionante". Se for este o caso, os efeitos da transação são ditos intermediários, a validação torna-se então uma validação hipotética, e a transação continuará. Em um determinado instante, se seus efeitos não forem mais questionados, então a transação validará *verdadeiramente* e então seus efeitos serão finais. A rigor, para estes casos, devemos falar dos efeitos finais de uma etapa da transação. Para uma

transação externa à unidade cooperativa, apenas os efeitos da validação da última etapa da transação serão visíveis.

A validação seletiva sugere claramente um mecanismo de herança e transferência de bloqueios similar àquele definido em [Moss-85] para transações aninhadas e que foi mostrado do capítulo anterior. Em caso de validação hipotética, a transação em questão deverá transferir seus bloqueios a outras transações selecionadas, e readquiri-los na sua próxima etapa, segundo o mecanismo de transferência de bloqueios de [Moss-85].

Como um exemplo, seja a equipe de arquitetos de um escritório de arquitetura, dividindo a tarefa de atualização do objeto complexo projeto da casa de um cliente. Os arquitetos de fachada e de interior representados pelas transações Ta1 e Ta2 respectivamente, cooperam a fim de preservar a consistência do objeto planta. As duas transações, Ta1 e Ta2, irão constituir então uma unidade cooperativa, que será controlada pela transação de controle U(Ta1,Ta2), como mostra a Figura 4.1. Supondo um grau de cooperação fraco entre as duas transações, quando Ta1 terminar o seu trabalho ela irá mostrar os seus resultados para a transação selecionada Ta2, que irá ler os resultados desta e adequar os seus de acordo com aqueles. Para isto, a transação Ta1 deverá liberar seus bloqueios para a sua transação-mãe, que no caso é U(Ta1,Ta2), e então a transação Ta2, que também é filha de U(Ta1,Ta2), deverá adquirir o bloqueio em modo compartilhado sobre o objeto fachada que foi liberado por Ta1.

Para que um protocolo de controle de concorrência possa ser utilizado em um ambiente onde as transações cooperam segundo o Modelo de Unidades Cooperativas, MUC, proposto em [Sampaio-95], este deverá satisfazer inteiramente às exigências que acabamos de relacionar. Os protocolos de controle de concorrência vistos no capítulo anterior, não satisfazem estas exigências em sua plenitude, o que torna necessário um novo protocolo de controle de concorrência que atenda a todas as exigências. A tabela da Figura 4.3 mostra as deficiências

encontradas em cada protocolo de controle de concorrência do capítulo 3, com relação às exigências do MUC e às características que o novo protocolo deverá ter.

	Protocolo de Moss	Protocolo de Gray	Protocolo de Garza-Kim	Novo Protocolo
Hierarquia "IS-A"	-------	X	X	X
Hierarquia É-PARTE-DE	-------	-------	X	X
Bloqueio a Nível de Subobjetos.	-------	-------	-------	X
Herança e Transferência de Bloqueios	X	-------	-------	X

Figura 4.3 Tabela Ilustrativa comparando os Protocolos de CC apresentados neste Estudo com o Novo Protocolo Proposto

Embora os protocolos mostrados no capítulo 3 supram apenas parte das exigências para mecanismos de controle de concorrência voltado para esses ambientes de aplicações cooperativas, as suas principais características deverão ser ainda aproveitadas junto com as características adicionais necessárias para aplicações cooperativas, como o bloqueio a nível de subobjetos e novos modos de bloqueio a objetos complexos que possibilite o seu compartilhamento por diferentes transações.

A seguir iniciaremos a discussão do novo protocolo de controle de concorrência para transações cooperativas que estão segundo o Modelo de Unidades Cooperativas, MUC, proposto em [Sampaio-95].

4.2 - PROTOCOLO DE CONTROLE DE CONCORRÊNCIA DE TRANSAÇÕES COOPERATIVAS.

Iniciamos a discussão do novo protocolo de controle de concorrência com uma discussão dos diferentes tipos de bloqueio a objetos.

4.2.1 - GRANULAGEM DE BLOQUEIOS

A Figura 4.4 mostra como estão estruturados os recursos segundo as suas granulagens, em um banco de dados onde atuam transações cooperativas. A menor granulagem considerada corresponde a atributos de objetos.

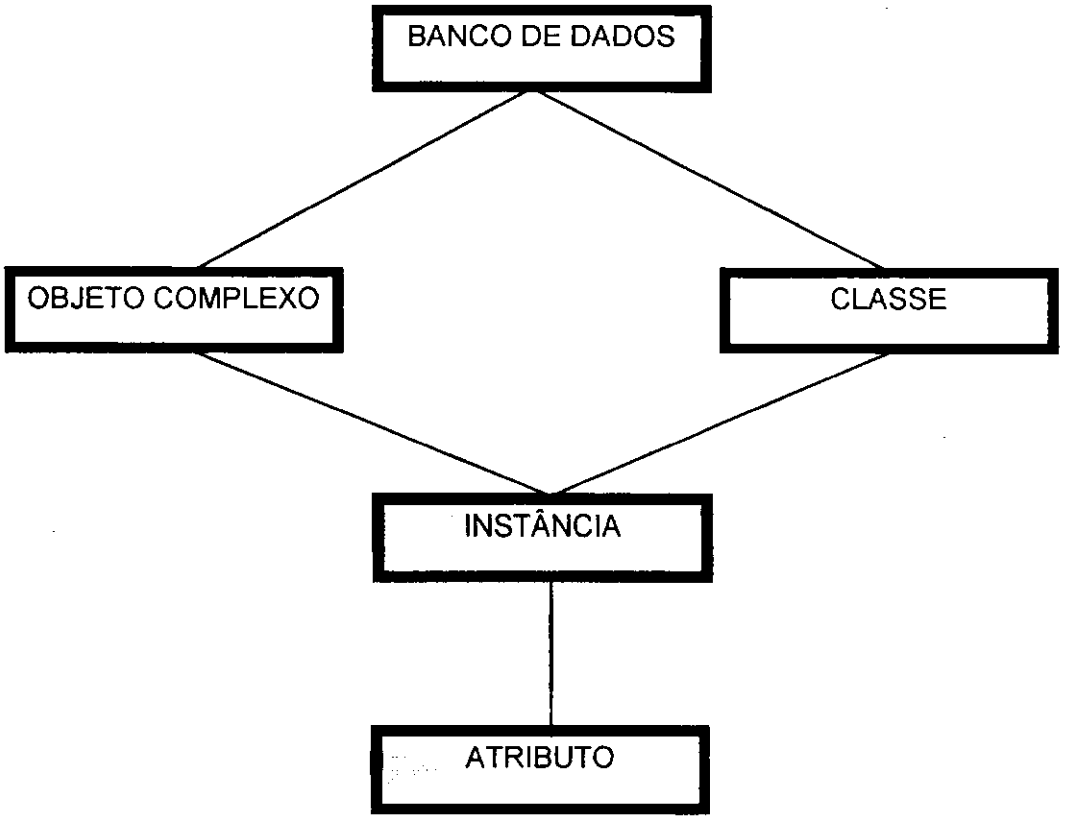


Figura 4.4 Grafo de Recursos em Banco de Dados onde atuam Transações Cooperativas que estão segundo o MUC.

Tomando como base a Figura 4.4, se uma transação T tiver que bloquear um grande número de instâncias de uma classe, ela deverá bloquear a classe toda, ao invés de bloquear cada instância da classe separadamente. Um bloqueio em uma classe implicará um bloqueio implícito em cada instância pertencente à classe. Por outro lado, quando apenas algumas poucas instâncias da classe necessitam ser bloqueadas, será melhor bloquear as instâncias individualmente.

Da mesma forma, se uma transação T tiver que bloquear um grande número de atributos de uma instância, será melhor bloquear toda a instância, visto que um bloqueio sobre uma instância irá implicitamente bloquear todos os atributos da instância. Por outro lado, se uma transação desejar bloquear apenas alguns atributos pertencentes a uma instância, esta deverá bloquear cada atributo separadamente.

Esses bancos de dados suportam tanto bloqueios lógicos como bloqueios físicos. Iremos considerar aqui apenas os bloqueios lógicos nos objetos mostrados na Figura 4.4.

A seguir, mostraremos os modos de bloqueios sobre as hierarquias em bancos de dados onde atuam transações cooperativas que estão segundo o MUC

Modos de Bloqueio em uma Hierarquia de Classes "IS-A".

Uma classe em uma hierarquia "IS-A" poderá ser bloqueada em qualquer um dos cinco modos de bloqueio definidos em [Gray-78], que são: IS, IX, SIX, S e X.

Os atributos serão apenas bloqueados nos modos S ou X (compartilhado ou exclusivo, respectivamente). Contudo, instâncias e classes poderão ser bloqueadas em qualquer um dos cinco modos mencionados.

Um bloqueio do tipo IS (Intenção de bloqueio compartilhado, em inglês "Intention Shared") em uma classe significa que instâncias da classe poderão ser explicitamente bloqueadas em modo IS ou S, quando necessário. Um bloqueio do tipo IS em uma instância significa que atributos da instância podem ser

explicitamente bloqueados em modo S, quando necessário. Por exemplo, seja a Figura 4.5a. Para que uma transação T coloque um bloqueio do tipo S em um atributo, digamos B3, da instância B, os seguintes bloqueios devem ser colocados: T bloqueia a classe C em modo IS, T bloqueia a instância B em modo IS e finalmente T bloqueia a instância B3 em modo S.

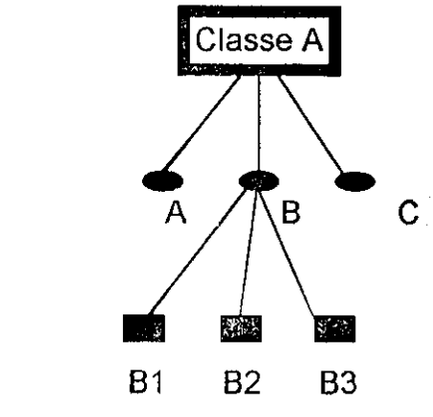
Um bloqueio do tipo IX (Intenção de bloqueio Exclusivo, em inglês "Intention eXclusive") em uma classe significa que instâncias da classe poderão ser explicitamente bloqueadas em modo IX, X, S ou SIX, quando necessário. Um bloqueio do tipo IX em uma instância significa que atributos da instância podem ser explicitamente bloqueados em modos S ou X, quando necessário. Tomando-se novamente como exemplo a Figura 4.5a, para que uma transação T coloque um bloqueio do tipo X em um atributo, digamos B1, da instância B, os seguintes bloqueios devem ser colocados: T bloqueia a classe A em modo IX, T bloqueia a instância B em modo IX, e T bloqueia B1 em modo X.

Um bloqueio do tipo S (bloqueio compartilhado) em uma instância significa que todos os atributos da instância são implicitamente bloqueados em modo S. Um bloqueio do tipo S em uma classe significa que a definição da classe é bloqueada em modo S, e todas as instâncias da classe serão implicitamente bloqueadas em modo S. Como ilustração, e considerando-se agora a Figura 4.5b, para que se bloqueie uma classe em uma hierarquia do tipo "IS-A", como mostrado na figura, em modo S, deverão ser colocados bloqueios nos modos IS sobre as suas correspondentes superclasses da hierarquia. Desta forma, para que uma transação T coloque um bloqueio em modo compartilhado S sobre a classe D, ela terá que colocar bloqueios do tipo IS sobre as correspondentes superclasses da classe D, que são B e A.

Um bloqueio do tipo X (bloqueio exclusivo) em uma instância significa que todos os atributos da instância são implicitamente bloqueados em modo X. Um bloqueio do tipo X em uma classe, significa que a definição da classe poderá ser lida ou atualizada e todas as instâncias da classe serão implicitamente bloqueadas em modo X. Novamente, tomando-se como base a Figura 4.5b, para que uma transação T

bloqueie uma classe em modo exclusivo, por exemplo a classe E, ela terá que colocar bloqueios do tipo IX sobre as correspondentes superclasses da classe E, que são as classes B e A.

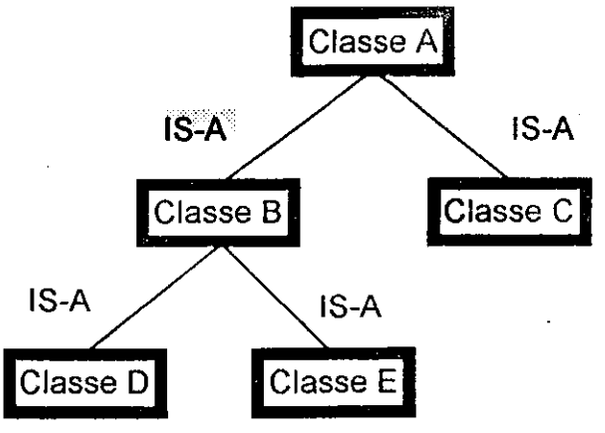
Um bloqueio do tipo SIX (bloqueio compartilhado e intenção de bloqueio exclusivo) em uma classe implica que a definição da classe é bloqueada em modo S, e todas as instâncias da classe são implicitamente bloqueadas em modo S, e as instâncias que serão atualizadas (pela transação que possui o bloqueio do tipo SIX) serão explicitamente bloqueadas em modo X. Um bloqueio do tipo SIX em uma instância implica que a instância é bloqueada em modo S e todos os atributos da instância são implicitamente bloqueaaos em modo S e os atributos que serão atualizados (pela transação possuindo o bloqueio SIX) serão explicitamente bloqueados em modo X. Novamente, tomando-se como base a Figura 4.5b, para que uma transação T bloqueie uma classe em modo SIX, por exemplo, a classe E, ela terá que colocar bloqueios do tipo SIX ou IX sobre as correspondentes superclasses da classe E, que são B e A.



- Instância.

- Atributo.

Figura 4.5a Classe A e suas Instâncias junto com seus Atributos



4.5b Hierarquia "IS-A cuja Classe-Raiz é a Classe A

Um bloqueio do tipo IS, IX, S, ou SIX em uma classe implicitamente previne a definição da classe de ser atualizada, e um bloqueio do tipo IS, IX, S ou SIX em uma instância, implicitamente previne a instância de ser atualizada. A matriz de compatibilidade da Figura 4.7 define completamente a semântica dos modos de bloqueio mencionados e dos modos de bloqueio que serão definidos adiante.

BLOQUEIO DE UMA CLASSE EM UMA HIERARQUIA É-PARTE-DE

Para que uma classe seja bloqueada em uma hierarquia É-PARTE-DE, serão incluídos dois novos modos de boqueio, que são: os modos CIS (que representa um IS em uma classe de uma hierarquia É-PARTE-DE de um objeto complexo, por analogia com o modo de bloqueio IS) e o modo CIX (que representa um IX em uma classe de uma hierarquia É-PARTE-DE de um objeto complexo, por analogia com o modo de bloqueio IX).

A semântica associada com estes novos modos de bloqueio será baseada na idéia de que diferentes transações podem possuir simultaneamente bloqueios sobre subobjetos distintos pertencentes a uma mesma instância de um objeto complexo em modos incompatíveis, ao mesmo tempo não permitindo que quaisquer outras transações bloqueiem o objeto complexo em modo incompatível com qualquer um dos bloqueios existentes nos subobjetos. Sem esses novos modos de bloqueio, não seria possível o bloqueio por parte de diferentes transações, de partes distintas (objetos componentes) de uma mesma instância de um objeto complexo.

Para que se bloqueie um objeto componente de uma instância de um objeto complexo em modo exclusivo ou compartilhado, será necessário o bloqueio em modo CIX ou CIS sobre as classes ancestrais da classe à qual pertence o objeto componente, o bloqueio em modo IX ou IS sobre a correspondente classe e finalmente o bloqueio em modo exclusivo ou compartilhado sobre o objeto componente.

Como ilustração, seja a Figura 4.6 a seguir. Para que uma transação T bloqueie um objeto componente, e, de uma instância de um objeto complexo a, em modo

exclusivo, ela deverá bloquear as classes A e C em modo CIX, a classe E em modo IX, e o objeto componente e, em modo X. Ao mesmo tempo, é possível que uma outra transação T1 atualize simultaneamente o objeto componente d, do objeto complexo a.

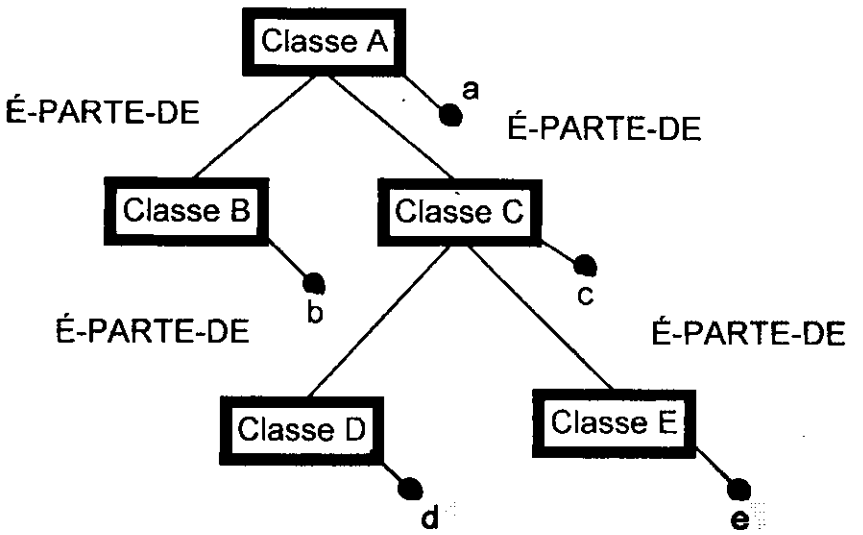


Figura 4.6 O Objeto Complexo A

Como um outro exemplo, tomemos como base novamente a Figura 4.1, onde uma árvore de transações reflete a estrutura do objeto complexo Projeto da Casa. Seja então as transações Ta1, Ta2 e Tfa, que representam respectivamente uma equipe de arquitetos em um escritório de arquitetura e um funcionário administrativo deste escritório. Sem os novos modos de bloqueios que são CIS e CIX, não seria possível por parte dessas transações junto com suas transações superiores o bloqueio simultâneo dos objetos componentes de uma mesma instância do objeto complexo Projeto da Casa, ao passo que, se uma transação externa Te tentar atualizar alguma instância pertencente a uma classe componente da hierarquia do objeto complexo, ela terá que esperar a liberação dos bloqueios por parte dessas transações.

A matriz de compatibilidade da Figura 4.7, define completamente a semântica associada aos novos modos de bloqueios definidos. Na figura, s é "sim", significando a compatibilidade entre os modos de bloqueios e n é "não", significando a incompatibilidade entre os modos de bloqueios.

	IS	IX	S	SIX	X	CIS	CIX
IS	S	s	s	s	n	S	n
IX	S	s	n	n	n	n	n
S	S	n	s	n	n	s	n
SIX	S	n	n	n	n	n	n
X	N	n	n	n	n	n	n
CIS	S	n	s	n	n	s	s
CIX	N	n	n	n	n	s	s

Figura 4.7 Matriz de Compatibilidade - Protocolo Proposto.

Deve ser observado que o modo de bloqueio CSIX (por analogia com o modo SIX) será desnecessário, por que é idêntico ao modo de bloqueio CIX, já definido.

O novo protocolo proposto para as aplicações cooperativas visa aumentar, como dito na seção 4.1, o nível de concorrência no sistema para estes tipos de aplicações, sendo apresentado a seguir.

4.2.2 - O PROTOCOLO DE CONTROLE DE CONCORRÊNCIA

O protocolo de controle de concorrência proposto é dividido em duas "grandes" regras, uma para definir como adquirir um bloqueio em uma hierarquia "IS-A" ou É-PARTE-DE, e a outra para definir herança e transferência de bloqueios para os casos de cooperação fraca ou forte entre as transações no interior de uma unidade cooperativa. Este protocolo não trabalha bem se for permitido que transações acessem a hierarquia aleatoriamente.

R1 - Adquirindo um bloqueio em uma hierarquia "IS-A" ou É-PARTE-DE

Transação não-cooperativa:

Adquirir os bloqueios da raiz para as folhas

Liberar os bloqueios das folhas para a raiz ao término da transação

Transação cooperativa

Transação_em_atividade = verdade

Enquanto transação_em_atividade

Adquirir bloqueios da raiz para as folhas

Liberar os bloqueios das folhas para a raiz no ato da validação

Se a transação termina, então transação_em_atividade = falso

HIERARQUIA "IS-A"

Para adquirir um bloqueio explícito do tipo S ou IS em um objeto, primeiro adquirir um bloqueio do tipo IS nos ancestrais ao longo da cadeia de ancestrais na hierarquia.

Para adquirir um bloqueio explícito do tipo X ou IX em um objeto, primeiro adquirir um bloqueio do tipo IX ou SIX nos ancestrais ao longo da cadeia de ancestrais na hierarquia.

HIERARQUIA É-PARTE-DE

Para adquirir um bloqueio explícito do tipo S ou IS em um objeto, primeiro adquirir um bloqueio do tipo CIS nos ancestrais ao longo da cadeia de ancestrais na hierarquia.

Para adquirir um bloqueio explícito do tipo X, IX, ou SIX em um objeto, primeiro adquirir um bloqueio do tipo CIX nos ancestrais ao longo da cadeia de ancestrais na hierarquia.

R2A - HERANÇA E TRANSFERÊNCIA DE BLOQUEIOS (COOPERAÇÃO FORTE)

Uma transação T apenas poderá adquirir um bloqueio do tipo exclusivo sobre um objeto, se todas as outras transações que adquiriram bloqueios do tipo exclusivo ou compartilhado são ancestrais de T.

Uma transação T poderá adquirir um bloqueio do tipo compartilhado sobre um objeto, se todas as outras transações que adquiriram bloqueios do tipo exclusivo são ancestrais de T.

Quando uma subtransação valida hipoteticamente, todos os seus bloqueios são liberados e herdados por sua transação-mãe.

Quando uma subtransação valida definitivamente, todos os seus bloqueios são liberados junto com as demais transações da unidade cooperativa.

Quando uma transação-raiz termina, todos os seus bloqueios são liberados.

R2B - HERANÇA E TRANSFERÊNCIA DE BLOQUEIOS (COOPERAÇÃO FRACA)

Uma transação T apenas poderá adquirir um bloqueio do tipo exclusivo sobre um objeto, se todas as outras transações que adquiriram bloqueios do tipo exclusivo ou compartilhado são ancestrais de T.

Uma transação T poderá adquirir um bloqueio do tipo compartilhado sobre um objeto, se todas as outras transações que adquiriram bloqueios do tipo exclusivo são ancestrais de T.

Quando uma subtransação valida definitivamente, todos os seus bloqueios são liberados e herdados por sua transação-mãe em modo compartilhado (de leitura).

Quando uma transação-raiz termina, todos os seus bloqueios são liberados.

Pelo fato de que, para transações não-cooperativas, o protocolo é do tipo bloqueio em duas fases, situações de interblocagem são potencialmente possíveis (execução serializável de transações não-cooperativas).

A fim de uma melhor compreensão desse nosso protocolo, serão dados alguns exemplos.

EXEMPLOS

EXEMPLO 1: Seja T1 e T2 duas transações planas que executam concorrentemente no sistema. Considere a Figura 4.2, que mostra um esquema de banco de dados.

A Transação T1 executará a seguinte consulta Q1 :

SELECT EACH projeto.planta
FROM Cliente
WHERE nome = "cliente_A" AND
projeto.contrato.orçamento > 50.000

Para esta consulta ela deverá adquirir os seguintes bloqueios :

IS na classe Pessoa
IS na classe Cliente
IS na instância cliente_A
CIS na classe Projeto
IS na classe Planta
S nas instâncias de Planta satisfazendo o predicado.

A Transação T2 por sua vez, tentará atualizar o banco de dados executando a seguinte consulta sobre o esquema mostrado na Figura 4.2 :

UPDATE cliente.endereço
SET endereço = endereço2
WHERE nome = "cliente_A"

A Transação T2 deverá adquirir os seguintes bloqueios :

IX na classe Pessoa
IX na classe Cliente
IX na instância de cliente_A
X no atributo endereço de cliente_A

Como a menor granulagem de bloqueio é o atributo de uma instância, este protocolo possibilita que duas transações possam possuir bloqueios sobre atributos diferentes pertencentes a uma mesma instância, simultaneamente.

EXEMPLO 2: Seja T1 e T2' duas transações planas que executam concorrentemente no sistema. T1 irá executar a consulta Q1 que foi descrita no exemplo anterior, já T2' irá executar a consulta Q2' descrita a seguir:

Q2'

SELECT projeto.contrato
FROM Cliente
WHERE nome = "cliente_A" AND
projeto.contrato.orçamento >= 50.000

Recupera todas as instâncias da classe Contrato correspondentes a cliente_A, cujo orçamento é maior ou igual a R$ 50.000,00.

A transação T2' deverá adquirir os seguintes bloqueios :

IS na classe Pessoa

IS na classe Cliente

IS na instância de cliente_A

CIS na classe Projeto da Casa

IS na classe Contrato

S nas instâncias de Contrato satisfazendo o predicado.

Vemos neste exemplo, que este protocolo irá permitir que duas transações executando independentemente possam acessar simultaneamente diferentes objetos componentes de uma mesma instância de um objeto complexo.

Nós iremos basear o exemplo a seguir, no caso mostrado na Figura 4.1, de uma equipe de arquitetos de um escritório de arquitetura que são representados pelas transações cooperativas Ta1 e Ta2, junto com um funcionário administrativo do escritório, representado pela transação Tfa, dividindo a atualização de uma instância pj do objeto Projeto da Casa.

EXEMPLO 3: Seja a unidade cooperativa hierárquica U(U(Ta1,Ta2),Tfa), e uma transação Te, externa a esta unidade cooperativa. No início, Ta1, Ta2 e Tfa representando respectivamente o arquiteto de fachada, o arquiteto de interior e o funcionário do setor administrativo, bloqueiam para atualização as instâncias de Fachada, f, de Interior, i, e de Contrato, c, do projeto pj de cliente_A. Concorrentemente, cliente_A, representado por Te, tenta acessar seu projeto pj.

Os bloqueios que devem ser colocados por parte de cada transação para atualização das respectivas instâncias, são mostrados seguir:

Ta1

IX na classe Pessoa.
IX na classe Cliente.
IX na instância cliente_A.
CIX na classe Projeto.
CIX na classe Planta.
IX na classe Fachada.
X na instância f de Fachada.

Ta2

IX na classe Pessoa.
IX na classe Cliente.
IX na instância cliente_A
CIX na classe Projeto.
CIX na classe Planta.
IX na classe Interior.
X na instância i de Interior.

Tfa

IX na classe Pessoa.
IX na classe Cliente.
IX na instância cliente_A.
CIX na classe Projeto.
IX na classe Contrato.
X na instância c de Contrato.

Te

IS na classe Pessoa.
IS na classe Cliente.
IS na instância cliente_A
IS na classe Projeto :
Esperando por Ta1,Ta2 e Tfa.

Pode-se verificar que, segundo as regras de herança e transferência de bloqueios do protocolo, (1) quando Ta1 (resp. Ta2) *valida hipoteticamente*, apenas Ta2 (Ta1) poderá ver seus efeitos; ao passo que quando U(Ta1,Ta2) *valida definitivamente* tanto Tfa quanto Te podem ter acesso aos seus efeitos finais. No entanto, para ter acesso a todo projeto pj, Te terá de esperar pela terminação definitiva de U(U(Ta1,Ta2),Tfa).

EXEMPLO 4: Como um outro exemplo, suponha o objeto complexo Projeto de Veículo, mostrado na Figura 4.8, junto com a unidade cooperativa hierárquica U(Tc1,U(Tc3,Tc4),Tc2), manipulando uma instância do objeto.

Dentro desta unidade, temos a subunidade cooperativa, U(Tc3,Tc4), cujas transações em seu interior, Tc3 e Tc4, cooperam fracamente, enquanto que as transações Tc1, Tc2 e U(Tc3,Tc4) também cooperam fracamente. Suponhamos

também que uma transação externa, Te, representando o dono da fábrica deseje examinar a instância do objeto Projeto de Veículo que esta sendo atualizada pela unidade cooperativa hierárquica U(Tc1,U(Tc3,Tc4),Tc2).

Os bloqueios colocados por cada transação são os seguintes :

Tc1

CIX na classe Proj. de Veículo.

IX na classe Subprj. da Carroceria.

X na Instância de Subprj. da Carroceria.

Tc2

CIX na classe Proj. de Veículo

IX na classe Subprj. do Interior.

X na Instância Subprj. do Interior.

Tc3

CIX na classe Projeto de Veículo.

CIX na classe Subprj. do Motor.

IX na classe Subprj. do Carburador.

X na Instância de Subprj. do Carburador

Tc4

CIX na classe Proj. de Veículo.

CIX na classe Subprj. do Motor.

IX na classe Subprj. do Bloco do Motor.

X na instância de Subprj. do Bloco do Motor.

Te

IX na classe Proj. de Veículo.

Espera pela terminação definitiva da unidade cooperativa hierárquica para poder atualizar os seus efeitos.

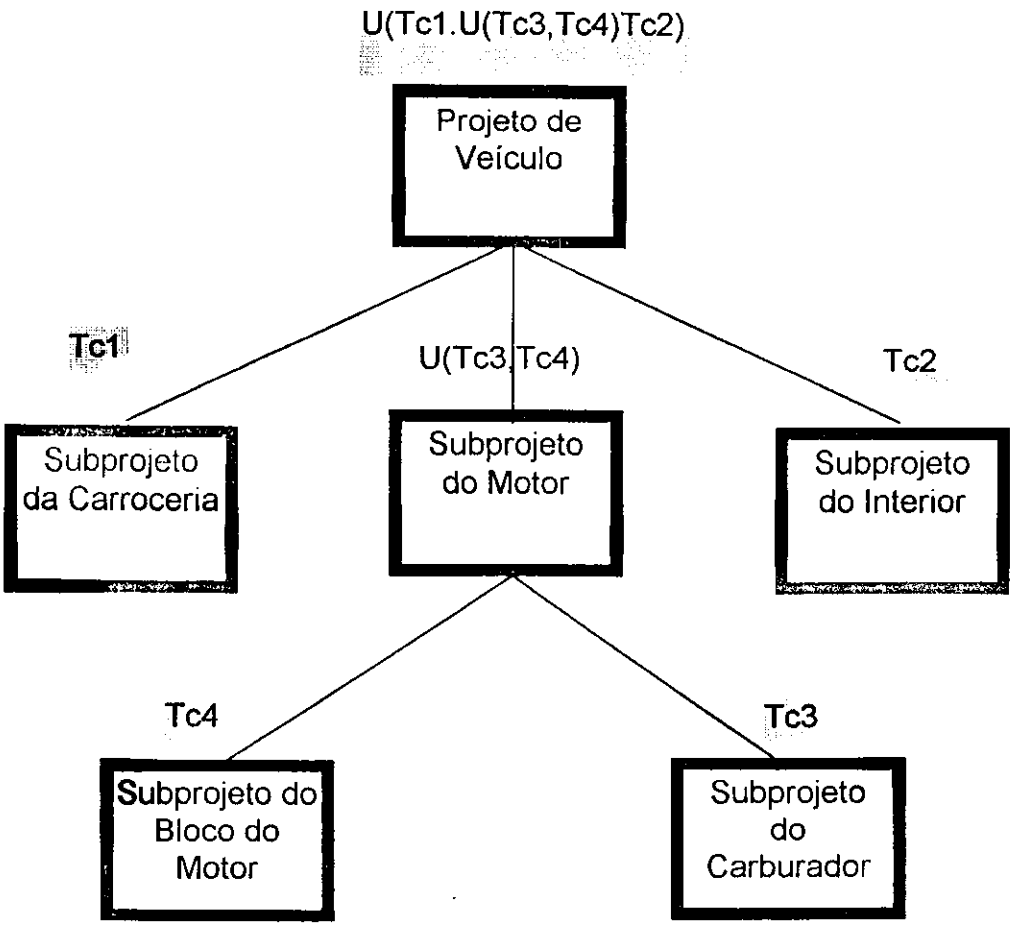


Figura 4.8 Objeto Complexo Projeto de Veículo

EXEMPLO 5: Suponha a unidade cooperativa hierárquica U(U(Ta1,Ta2),Tfa), mostrados na Figura 4.1, atualizando a instância pj do objeto complexo Projeto da Casa.

Sejam as transações Ta1, Ta2 e U(Ta1,Ta2) constituindo uma subunidade cooperativa dentro da unidade cooperativa hierárquica, supondo ainda que as transações Ta1 e Ta2 cooperam fracamente a fim de atualizarem o objeto de Planta, p, correspondente a pj.

Suponha que a transação Ta1 validou os seus efeitos seletivamente para a transação Ta2, e que uma transação externa a esta unidade cooperativa hierárquica tenta acessar os seus efeitos. Os bloqueios que serão colocados por parte de cada uma dessas transações nesta situação são mostrados a seguir:

Ta1 Liberou os seus bloqueios que serão herdados por U(Ta1,Ta2) em modo compartilhado.

U(Ta1,Ta2)

CIS na classe Projeto

CIS na classe Planta

IS na classe Fachada

S na instância, f, de Fachada

Ta2

CIX na classe Projeto

CIX na classe Planta

IX na classe Interior

X na instância, i, de Interior

IS na classe Fachada

S na instância, f, de Fachada

Te (Transação Externa)

CIS na classe Projeto

CIS na classe Planta

IS na classe Fachada

S na instância, f, de Fachada

Vemos que a transação externa, Te, não irá precisar esperar pela validação definitiva da subunidade cooperativa U(Ta1,Ta2) para poder examinar uma parte dos seus efeitos.

4.3 DISCUSSÃO

O Nosso protocolo traz como grande novidade a possibilidade de que duas transações concorrentes atualizem simultaneamente objetos componentes de uma mesma instância de um objeto complexo.

Por outro lado, quando uma transação T estiver acessando para atualização um objeto componente, ela bloqueará implicitamente todos os seus subobjetos e todos os objetos baseados nas classes desses subobjetos, além de também bloquear implicitamente todos os objetos baseados nas classes ancestrais deste.

Como um exemplo, seja o objeto complexo "Projeto da Casa". Considere a instância P_02, cujo objeto componente planta_02 é acessado em modo de atualização por uma transação T. Desta forma, se uma outra transação T' que seja externa à árvore de transações em T, quiser acessar em qualquer modo alguma instância da classe "Projeto da Casa" ou alguma instância das classes Fachada e Interior isto não será possível devido ao bloqueio implícito sobre as instâncias dessas classes pela transação T.

Como na prática apenas os objetos que estão no último nível de uma hierarquia É-PARTE-DE serão atualizados por transações que irão cooperar com o objetivo de preservar a consistência do objeto-raiz da hierarquia, faremos a suposição simplificadora de que apenas as transações que estão no último nível de uma árvore de transações (transações que não possuem subtransações ou transações filhas) adquirem bloqueios para a manipulação de objetos componentes. Por sua vez, as transações superiores a estas não irão manipular os objetos diretamente, ficando sua função restrita ao controle e transferência de bloqueios entre suas transações inferiores.

Disto vemos que na prática o nosso protocolo não será tão restritivo quanto possa parecer à primeira vista, ao contrário, o nosso protocolo irá permitir que mais de uma transação acesse simultaneamente e em modo exclusivo instâncias das classes do último nível da hierarquia, ainda possibilitando que objetos componentes de uma instância de um objeto complexo sejam acessados simultaneamente por mais de uma transação, tornando o nosso protocolo menos restritivo.

Também, além de prevê a herança e transferência de bloqueios entre transações pertencentes a uma árvore de transações que atuam sobre um objeto complexo compartilhando a sua atualização, este nosso protocolo permite ainda que para o

caso de um grau de cooperação fraco entre transações no interior de uma unidade cooperativa, uma transação externa a esta não terá de esperar pelo seu término para ter acesso a uma parte de seus efeitos, tornando o nosso protocolo ainda menos restritivo.

Em contrapartida, o nosso protocolo irá acarretar um aumento do número de bloqueios explícitos por parte de uma transação sobre os objetos devido ao fato de que a menor granulagem de bloqueio existente é a nível de atributos de objetos. Além disso, como o nosso protocolo se baseia nas intenções de bloqueios para prevenir que duas requisições de bloqueios em modos incompatíveis sejam colocados sobre o mesmo objeto, quando se desejar fazer uma mudança na definição de uma classe ou quando se for fazer uma consulta junto a uma classe quando a classe ou os domínios de seus atributos estiverem próximos ao último nível de uma profunda hierarquia "IS-A" ou de uma profunda hierarquia É-PARTE-DE irá acarretar em um grande número de bloqueios explícitos causando um maior *overhead* ao "gerenciador de bloqueios". No entanto, esta abordagem escolhida para o nosso protocolo nos foi fundamental a fim de possibilitar um maior nível de concorrência para o caso de trabalhos cooperativos onde as transações cooperativas atuam diretamente sobre objetos componentes que estão no último nível de uma hierarquia É-PARTE-DE.

Como o enfoque deste protocolo é de trabalhos cooperativos, ele irá ter um bom desempenho, salvo em alguns casos em que uma transação deseja consultar uma instância de uma classe em um nível intermediário de uma hierarquia É-PARTE-DE.

5 CONCLUSÕES E PERSPECTIVAS

5.1 - CONCLUSÕES

As propriedades ACID de transações convencionais satisfazem as características destas transações, como por exemplo, tempo de duração pequeno, nenhuma interatividade entre transações etc.

Entretanto, para o caso de transações cooperativas, que apresentam em suas execuções características bem diferentes das características de transações convencionais, as propriedades ACID deverão ser flexibilizadas para dar suporte a essas novas características.

Como conseqüência, os protocolos de controle de concorrência para transações convencionais não se mostram adequados às transações cooperativas.

O objetivo desta dissertação de mestrado foi o de propor um protocolo de controle de concorrência para aplicações cooperativas cujas transações cooperam segundo o modelo proposto em [Sampaio-95]. Inicialmente, foram apresentadas as principais características de alguns protocolos de controle de concorrência encontrados na literatura para transações não-convencionais, como o protocolo de [Gray 81], baseado em granulagem de bloqueio, para hierarquias de classes em bancos de dados orientados a objetos e o protocolo de [Garza-Kim 88], que inclui também bloqueio de objetos complexos. Estes protocolos se caracterizam principalmente por serem aplicáveis somente a transações de um só nível, não-cooperativas.

Um outro protocolo também analisado nesta dissertação foi o protocolo de [Moss-85], voltado para transações de vários níveis, mas ainda não-cooperativas.

Finalmente, a análise detalhada desses protocolos levou-nos a propor um novo protocolo de controle de concorrência, baseado em granulagem de bloqueios, que poderá ser aplicado às transações cooperativas segundo o modelo MUC, proposto em [Sampaio-95]. Neste modelo, um grupo de transações cooperam a fim de preservar a consistência do banco de dados. As transações controladas por este protocolo nem sempre irão adotar a serializabilidade como critério de correção, dependendo do grau de cooperação existente entre as transações cooperativas. Mais especificamente, para um grupo de transações que coopera no interior de uma unidade cooperativa, se o grau de cooperação é fraco, então as transações ainda são isoladas umas das outras (transações serializáveis). Se o grau de cooperação é forte então as transações no interior da unidade cooperativa não são serializáveis, podendo então interagirem para mostrar seus efeitos intermediários umas às outras.

5.2 - TRABALHOS FUTUROS

O protocolo de controle de concorrência proposto nesta dissertação de mestrado, poderá ainda evoluir em algumas direções a fim de que seja melhor detalhado

- Implementação e análise do desempenho do algorítmo.

REFERÊNCIAS BIBLIOGRÁFICAS

[Agrawal 92] Agrawal, D., El Abbadi, A. A Non Restrictive Concurrency Control for Object Oriented Databases, EDBT, 1992, Vienne, 469-482.

[Baptista 91] Baptista, C. Controle de Concorrência com Transações Multiníveis, Dissertação de Mestrado, UFPB, Campina Grande, 1991.

[Bernstein 87] Bernstein, P., Hadzilacos, V., Goodman, N. Concurrency Control and Recovery in Database Systems. Addison - Wesley, 1987.

[Date 85] Date, J. An Introduction to Database Systems. Addison - Wesley, **V II**, 1985.

[Garza 88] Garza, J. F., Kim, W. Transactions Management in an Object Oriented Database System. Proc. ACM SIGMOD, 1988, Chicago, 37-45.

[Gray 78] Gray, J. Notes on Database Operating Systems. IBM Research Laboratory, San Jose, California, 1978.

[Gray 81] Gray, J. The Transaction Concept: Virtues and Limitations. Proc. Intl. Conference on VERY LARGE DATABASE SYSTEMS, 1981, Cannes, 144-154.

[Kent 79] Kent, W. Limitations of Record-Based Information Models. ACM Transactions on Database Systems, Vl. 4, No. 1, March 1979, 107-131.

[Kim 88] Kim, W., Chow, H., Banerjee, J. Operation and Implementation of Complex Objects, IEEE Transactions on Software Engeneering,V. 14, No. 7, July/1988, 985-996.

[Kim 90a] Kim, W. An Introduction to Object-Oriented Databases. The MIT Press, 1990.

[Kim 90b] Kim, W. Object-Oriented Database Definition and Research Directions, IEEE Transaction on Knowledge and Data Engineering, V. 2, No 3, 1990, 327-339.

[Kim 91] Kim, W., Ballou, N., Garza, J. F., Woelk, D. A Distributed Object-Oriented Database System Supporting Share and Private Databases. ACM Transactions on Information Systems. V. 9, No. 1, January, 1991, 31-51.

[Kortz 89] Kortz, H., Silberschatz. Sistemas de Bancos de Dados, Mc Graw Hill, 189.

[Lorie 83] Lorie, R., Plouffe, W. Complex Objects and their use in Design Transactions. Proc. Databases for Engineering Applications, 1983, 115-121.

[Moss 82] Moss, J. Nested Transactions and Reliable Distributed Computing. Proc. IEEE Sysmposium on Reliability in Distributed Software and Database System, July/1982, Pittsburgh, 33-39.

[Moss 85] Moss, J. Nested Transactions: An Approach to reliable Distributed Computing. MIT Press, 1985 (Series on Information System).

[Sampaio 95] Sampaio, M., Cooperative Transactions: a taxonomy and a new model. X Simpósio Brasileiro de Bancos de Dados, Recife, Outubro/1995, 83-99.

[Sampaio 96] Sampaio, M., Turc, S. Cooperative Transactions: A Data Driven Approach. Proc. of the 29th Int. Conference on System ....., Honolulu-USA, 1996.

[Sampaio 97] Sampaio, M., Nascimento, C. Rule Support in an Object-Oriented Data Manager for Cooperative Design. XII Simpósio Brasileiro de Banco de Dados, Fortaleza, Outubro/1997, 287-301.

[Souto 95] Souto, M. A., Iochpe, C., Transações de Banco de Dados: Novos Requisitos de Processamento de Aplicações Não Convencionais. Revista de Informática Teórica e Aplicada, V. 2, No. 1, 77-121, Janeiro/1995.

[Santos 92] Santos, M. Um Modelo para o Gerenciamento de Transações

Referências Bibliográficas

[Santos 92] Santos, M. Um Modelo para o Gerenciamento de Transações Longas em ambientes de Banco de Dados, Dissertação de Mestrado, UFPE, Recife, 1992.