Um Framework para construção de Sistemas de Banco de Dados Móvel com Regras Ativas

Sérgio da Costa Côrtes
scortes@inf.puc-rio.br

Carlos Jose Pereira de Lucena
lucena@inf.puc-rio.br

PUC-RioInf.MCC35/01 Outubro 2001

Abstract

This study presents an overview of the DBS-M-AR framework, with emphasis on its components and hot spots. This is an object-oriented framework for the construction of Mobile Database Management Systems with Active Rules. A detailed discussion of its instantiation is also included.

Keywords: Mobile system, framework, framelet, database, active rules, distributed systems.

Resumo

Este trabalho apresenta uma visão geral do framework SBD-M-RA, com ênfase em seus componentes e hot spots. Trata-se de um framework orientado a objetos para a construção de sistemas de gerência de bancos de dados móvel com regras ativas. Uma discussão detalhada de sua instanciação também é apresentada.

Palavras-chave: Sistemas móveis, framework, framelet, banco de dados, regras ativas, sistemas distribuídos.

1. Introdução

Recentes avanços na tecnologia da comunicação sem fio (*wireless*) têm transformado os serviços de informações móveis em uma realidade. Um número cada vez maior de novos sistemas móveis, tais como sistemas de navegação, todos os tipos de vendas, segurança pública, entre outros, exemplifica aplicações emergentes que necessitam de acessos a bancos de dados utilizando equipamentos do tipo palmtop, PDA (Personal Digital Assistant) e notebook. A tecnologia de computação móvel não é somente uma melhoria na distribuição e no fluxo de informações, mas é também um incrível aumento na funcionalidade das aplicações de bancos de dados.

Sistemas de Gerência de Banco de Dados (SGBD) provêm confiança, eficiência, acesso, alta disponibilidade e mecanismos eficazes para armazenamento e manutenção de grandes volumes de informações. Pesquisas e trabalhos práticos têm se direcionado para a criação de bancos de dados voltados para aplicações bastante diferenciadas das convencionais, estendendo sua semântica, de forma que o próprio SGBD as suporte. SGBD com Regras Ativas podem ser empregados em ambientes móveis, utilizando seu comportamento ativo no sentido de resolver alguns problemas inerentes ao ambiente móvel.

O uso da técnica de framework para a construção de Sistemas de Bancos de Dados Móveis com Regras Ativas permite o desenvolvimento de diferentes softwares deste tipo, re-aproveitando as funcionalidades que sejam comuns e permitindo implementações específicas para diversos ambientes operacionais, com diferentes semânticas de regra.

Diversos frameworks estão sendo desenvolvidos para diferentes aplicações. Em [14] é proposto um framework e uma arquitetura de sistemas para construção de um sistema de informações móvel utilizando-se um DBMS ativo;.uma proposta de uma arquitetura reusável é proposta em [13]. No entanto, não foi encontrada qualquer descrição do uso da técnica de framework com o intuito de construir DBMS Móveis com Regras Ativas.

Quanto à computação móvel, tendências e o futuro da computação móvel são apresentados em [01], onde se destacam os esforços de padronização, as arquiteturas e protocolos para computação móvel. Um modelo de transação global para um banco de dados móvel é descrito em [04]. Este modelo é implementado usando sub-transações compensáveis. Um modelo de concorrência baseado no esquema de bloqueio em duas fases (*two phase locking*) foi proposto para um protocolo de bloqueio distribuído em tempo-real em [05]. O problema de recovery e rollback de transações móveis é abordado em [06]. Em [15] é feita uma profunda análise e comparação de como as características (*features*) móveis influências nas propriedades ACID das transações de banco de dados. A utilização de agentes móveis para construir aplicações flexíveis e eficientes em plataformas distribuídas é apresentada em [07]. No entanto, não foi encontrado qualquer trabalho que indique o uso da tecnologia de SGBD com Regras Ativas com o intuito de resolver os problemas de ambientes móveis.

O restante deste trabalho está organizado da seguinte forma. Na seção 2, a conceituação necessária para o desenvolvimento do framework SBD-M-RA é descrita. Na seção 3 o framework SBD-M-RA, seus principais componentes e seus *hot spots*. Na seção 4 sua instanciação. Finalmente, na seção 5, o *status* corrente da pesquisa e alguns passos futuros são comentados.

2. Sistemas Móveis, Sistemas de Gerência de Bancos de Dados e Frameworks

Mobilidade tornou-se um novo paradigma em sistemas distribuídos [19], causando efetivamente mudanças significativas na forma de desenvolver sistemas de software. Entre suas principais características se destacam a mobilidade de seus hosts, sua interface de comunicação sem fio e a portabilidade de seus dispositivos. [02]. A capacidade dos sistemas de computação não pára de crescer, existe uma tendência acentuada de desenvolvimento dos novos equipamentos móveis e, cada vez mais, aplicações estão sendo desenvolvidas oferecendo um número cada vez maior de serviços que necessitam da utilização de Sistemas de Bancos de Dados – SBD [01]. Um SBD é composto por seus Bancos de Dados, seu Sistema de Gerência de Banco de Dados – SGBD e todas as Aplicações de seus Usuários [18]. Assim, os SBD para uma plataforma Móvel precisam ser projetados para atender às necessidades e funcionalidades da computação móvel. A utilização de frameworks na construção dos SBD para plataforma móvel, certamente, trará o aumento do reuso do software e a redução do tempo para desenvolvimento de suas aplicações.

2.1. Sistemas Móveis

Mobilidade tornou-se um novo paradigma em sistemas distribuídos. Como conseqüência da mobilidade, destaca-se a complexa e difícil tarefa de manter as bases de dados operacionais atualizadas, em função de sua rápida variação. Em relação a interface de comunicação, há o problema da qualidade da conexão sem fio, que varia abruptamente em função do local , as falhas de comunicação entre os terminais remotos e seus hosts, e a segurança da comunicação entres os clientes e seus servidores. Quanto a portabilidade dos dispositivos, destacam-se a pouca memória disponível nos equipamentos móveis, a lentidão de seus processadores e a sua capacidade em se manter ativo em função da energia fornecida por sua bateria [02]e [03].

2.1.1. Arquitetura de um Sistema Móvel

A arquitetura genérica de uma plataforma móvel consiste em uma arquitetura distribuída [19], na qual diversos computadores, geralmente conhecidos como *Hosts Fixos* (FS - Fixed Hosts) e *Estações de Base* (BS – Base Stations), são interligados através de uma rede cabeada de alta velocidade. Hosts fixos são computadores de finalidade genérica que não são equipados para gerenciar unidades móveis, mas podem ser configurados de forma a fazê-lo. Estações de base são equipadas com interfaces não-cabeadas e podem comunicar-se com *unidades móveis* para suportar o acesso a dados ([18] e [11]).

Unidades Móveis (Mobile Units) são computadores portáteis movidos à bateria, que se movimentam livremente em um domínio geográfico de mobilidade, uma área que é restringida pela limitada amplitude de banda dos canais de comunicação sem fios. Para gerenciar a mobilidade das unidades, o domínio de mobilidade geográfica é dividido em domínios menores chamados de *células*. A disciplina móvel requer que o movimento de unidades seja irrestrito dentro do domínio de mobilidade geográfica (movimento intercelular), ao mesmo tempo em que, possuir contigüidade de acesso durante o movimento, garante que o movimento de uma unidade móvel através dos limites das células não terá nenhum efeito sobre o processo de recuperação de dados [18].

Unidades Móveis e *Estações de Base* se comunicam através de canais não-cabeados (*wireless*) que possuem amplitudes de banda significativamente menores do que aquelas de uma rede cabeada. Um canal *downlink* é utilizado para enviar dados de *Estações de Base* para *Unidades Móveis* e um canal *uplink* é utilizado para enviar dados de *Unidades Móveis* para *Estações de Base* [18].

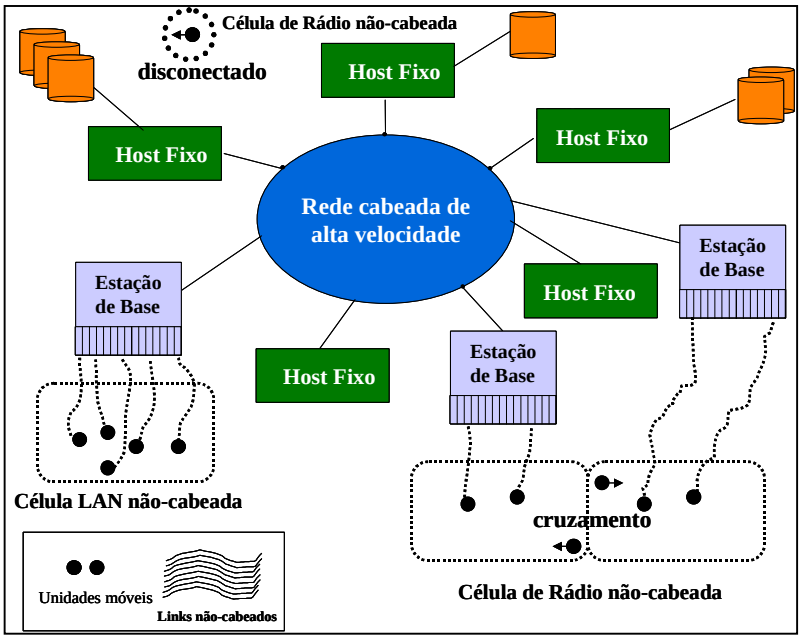

Figura 1: Uma arquitetura genérica para uma plataforma móvel [18].

Entre os principais requisitos impostos pelas características da computação móvel aos sistemas de softwares destacam-se [02]: (1) capacidade de localizar/endereçar elementos móveis; (2) capacidade de perceber mudanças no ambiente de execução; (3) capacidade de se auto-configurar; (4) capacidade de migrar funcionalidade; (5) uso eficiente dos recursos no elemento móvel; (6) uso eficiente do recurso canal de comunicação não-cabeados (*wireless);* (7) alto grau de tolerância à falhas; e (8) mecanismos para autenticação e cifragem de dados.

2.1.2. Gerência de Dados em um Ambiente Móvel

A computação móvel pode ser considerada como uma variação da computação distribuída [19]. Bancos de dados móveis podem ser distribuídos sob dois possíveis aspectos: (1) Todo o banco de dados é distribuído, principalmente, entre os componentes cabeados, possivelmente com replicação total ou parcial. Assim, uma estação de base gerencia seu próprio banco de dados com uma funcionalidade do tipo SGBD, com funcionalidades adicionais para localizar unidades móveis e características adicionais de gerência de consultas e transações, para atender aos requisitos de ambientes móveis; (2) O banco de dados é distribuído entre componentes com fios e componentes sem fios. A responsabilidade sobre a gerência de dados é compartilhada entre estações de base e unidades móveis.

Portanto, as questões relativas à gerência de dados distribuídos podem também ser aplicadas a bancos de dados móveis com os seguintes aspectos sendo analisados de forma especial: (1) distribuição e replicação de dados; (2) modelos de transações; (3) processamento de consultas; (4) recuperação e tolerância a falhas: (5) projeto de bancos de dados móveis ([18], [04], [05] e [06]).

2.2. Sistemas de Gerência de Banco de Dados - SGBD

Um SGBD é um conjunto de softwares altamente complexo e sofisticado que possibilita, basicamente, os serviços de armazenamento, recuperação e gerência de dados. Existe uma grande diversidade de arquiteturas de SGBD, no entanto, é possível identificar um conjunto básico de funcionalidades que normalmente consta das mais diferentes arquiteturas [18]. Entre os principais componentes comuns destacam-se: o compilador de sua linguagem de definição de dados e de manipulação de dados, o processador de consultas, bem como o gerente de banco de dados, de arquivos e de dicionário de dados (metadados). O módulo gerente de banco de dados abrange os módulos de controle de autorização, processador de comando, verificador de integridade, otimizador de consulta, gerente de transação, scheduler (escalonador), gerente de recuperação e gerente de buffer. Além dos módulos citados, existem diversas estruturas de dados que são necessárias como parte da estrutura física para implementação dos bancos de dados, entre elas os arquivos de dados, de índices e dicionários de dados.

2.2.1. SGBD com Regras Ativas - SGBDRA

SGBD convencionais são ditos passivos, isto é, os dados são criados, recuperados, modificados e excluídos somente em resposta a operações feitas por usuários ou programas aplicativos. Já os SGBDRA executam algumas operações automaticamente, em resposta à ocorrência de certos eventos, como a exclusão de uma linha de uma tabela, para satisfazer algumas condições predefinidas, como garantir o conjunto de restrições de integridade, algum evento temporal ou de uma aplicação. SGBDRA possibilitam o desenvolvimento de bancos de dados não convencionais, dando suporte a uma semântica que reflete o comportamento com base em eventos ou acontecimentos em um determinado domínio do banco de dados [20]. Os SGBDRA são semanticamente mais expressivos pois podem: (1) realizar funções que em sistemas passivos precisam ser codificadas em aplicações, como manter a réplica atualizada de um dado em outro *site*; (2) facilitar o desenvolvimento e manutenção de regras do negócio que antes tinham que ser desenvolvidas, as vezes redundantemente, dentro do escopo de vários sistemas passivos; (3) realizar tarefas que requerem subsistemas com "objetivos especiais" em sistemas passivos de banco de dados, como manutenção de integridade de dados entre sistemas.

Os SGBDRA são centrados no conceito de regras, que especificam o comportamento ativo desejado. De forma geral, as regras dos SGBDRA são compostas por três partes: (1) **Evento**: faz a regra ser disparada (*triggered*); (2) **Condição**: é verificada quando a regra é disparada; (3) **Ação**: executada quando a regra é disparada e se a condição for verdadeira. Existe uma grande variedade de alternativas para eventos, condições e ações porém, em geral, estas envolvem operações usuais de acesso e manipulação, internas aos bancos de dados. Outros componentes, específicos do contexto de regra, também podem ser incluídos como, por exemplo, disparos semânticos e temporais, utilização do estado de transição anterior ou posterior do banco de dados [17].

2.3. Framewoks

O reuso de software tem sido um dos principais objetivos da engenharia de software. Reutilizar software não é simples. Com o surgimento do paradigma da orientação a objetos, a tecnologia adequada para reuso de grandes componentes tornou-se disponível e resultou na definição de frameworks orientados a objetos. Frameworks têm atraído a atenção de muitos pesquisadores e engenheiros de software e têm sido definidos para uma grande variedade de

domínios. As principais vantagens de framework são, entre outras, o aumento do reuso e a redução do tempo para desenvolvimento de aplicações [23].

2.3.1. Arquitetura de Framework

Um framework consiste em *frozen spots* e *hot spots*. *Frozen spots* definem a arquitetura global de um sistema de software – seus componentes básicos e os relacionamentos entre eles. Eles permanecem imutáveis em qualquer instanciação do framework. *Hot spots* representam aquelas partes do framework que são específicas para uma determinada implementação do sistema de software. *Hot spots* são projetados para serem genéricos – eles podem ser adaptados para as necessidades específicas da aplicação em desenvolvimento. Quando se cria um sistema de software concreto usando um framework, seus *hot spots* são preenchidos de acordo com as necessidades e requisitos específicos do sistema [23].

Um framework de software é uma mini-arquitetura reutilizável que fornece a estrutura e comportamento, genéricos, para uma família de abstrações de software, com um contexto que especifica suas interações e uso dentro de um determinado domínio. Esta especificação é completada através da codificação do contexto, onde as abstrações têm "fim aberto" (*open-ended*), sendo projetadas como *plug-points* específicos. Estes *plug-points* (tipicamente implementados usando *callback* ou polimorfismo) permitem que o framework seja adaptado ou estendido para atender variadas necessidades, e que possa ser combinado com outros frameworks. Um framework não é uma aplicação completa: falta a necessária funcionalidade específica de uma aplicação. Ao invés disso, uma aplicação pode ser construída a partir de um ou mais frameworks, inserindo-se as funcionalidades ausentes em suas lacunas *plug-and-play* [23].

Enfim, um framework fornece a infra-estrutura e os mecanismos que executam uma política para interação entre componentes abstratos com implementações abertas. Os principais benefícios dos frameworks orientados a objetos decorrem da modularidade, reusabilidade, extensibilidade e inversão de controle que eles oferecem aos desenvolvedores.

2.3.2. Framelet

Framelet são frameworks que possuem as seguintes características: (1) não assumem o controle principal da aplicação; (2) possuem no máximo 10 classes/componentes; e (3) possuem uma interface simples. O termo framelet explicita um pequeno framework, normalmente utilizado na construção de outros frameworks. Conseqüentemente, os princípios para construção de framework podem ser aplicados na construção dos framelets [21].

3. Um Framework para Sistema de Banco de Dados Móvel com Regras Ativas

Um framework para construção de SBD Móvel com Regras Ativas – SBD-M-RA é um conjunto de software que usa as tecnologias de banco de dados convencionais e ativos, bem como de frameworks, aplicados à *computação móvel*. A seguir, são apresentados seus componentes, classificados em *hot* e *frozen spots*, e seu uso é exemplificado através de uma instanciação do framework proposto.

3.1. Objetivo do Framework

O objetivo do framework proposto é possibilitar a construção de SBD-M-RA, em ambiente móvel, que possam diferir entre si em função da forma como controlam e administram a execução das regras, o modo de detectar seus eventos, bem como quanto ao aspecto de distribuição de regras e dados ([2], [3] e [22]).

O framework proposto está contido na arquitetura de um SGBD convencional e o estende com características de banco de dados ativos e distribuídos, bem como do ambiente de computação móvel, que podem ser configuradas conforme o interesse. A integração do framework proposto na arquitetura de um SGBD, bem como sua interação com os demais componentes, estão apresentadas na figura 2.

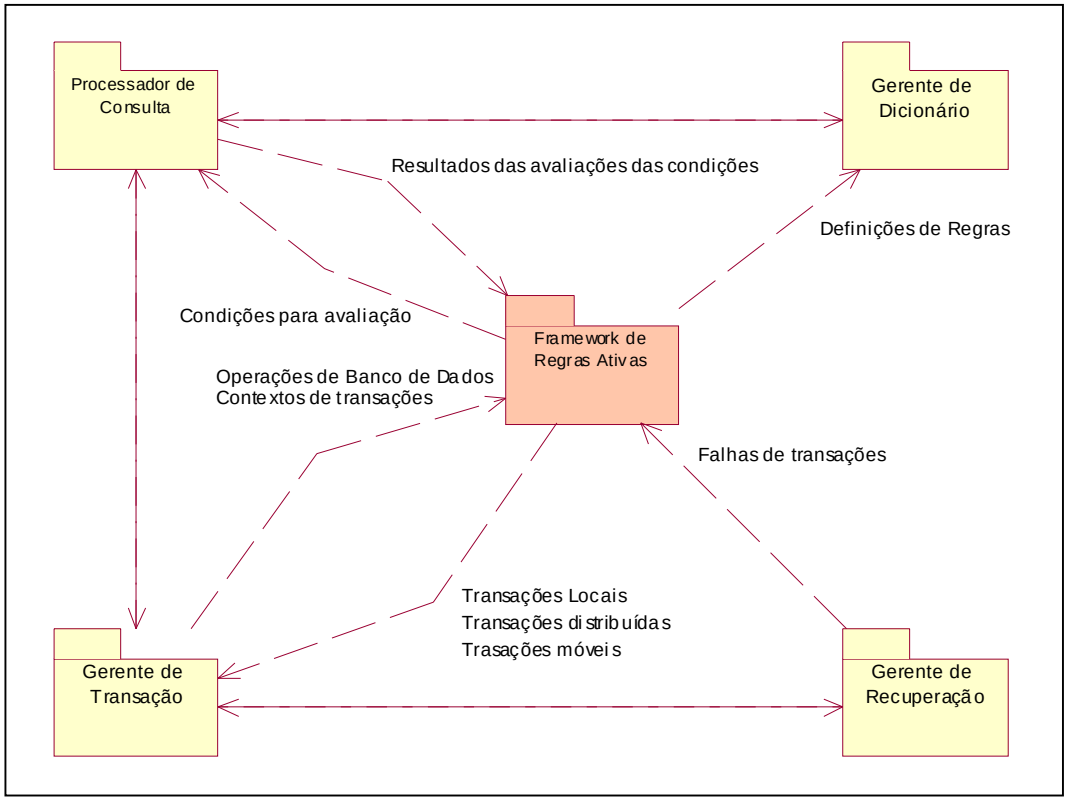

Figura 2: Arquitetura do SBD Móvel com o Framework de Regras Ativas.

3.2. Componentes do Framework

O framework de Regras Ativas para um ambiente móvel é composto por um conjunto de componentes e está apresentado na Figura 3. O componente Gerente de Distribuição é um *framelets.*

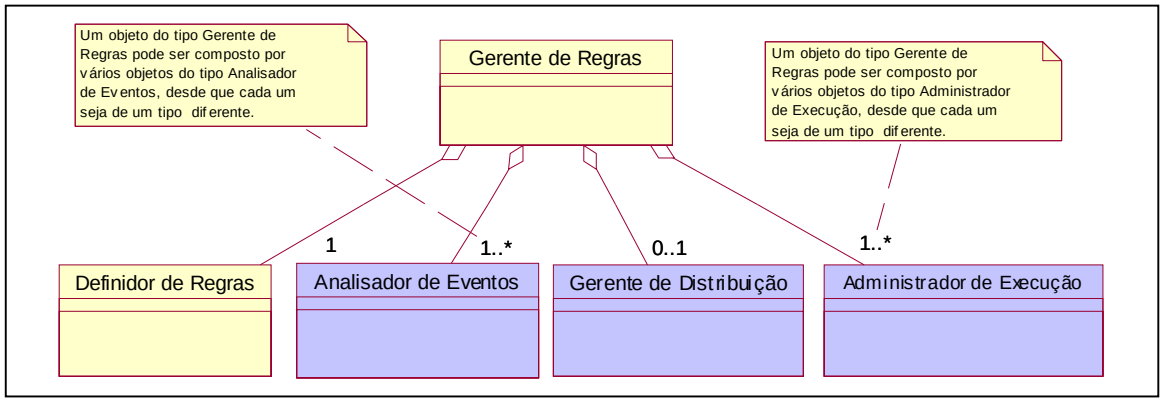

Figura 3: Framework de Regras Ativas para um Ambiente Móvel.

Os componentes *Gerente de Regras* e *Definidor de Regras* correspondem aos *frozen spots* do framework, ou seja, conservam-se inalteráveis em qualquer de suas instanciações. Os componentes *Analisador de Eventos, Gerente de Distribuição e Administrador de Execução* correspondem aos *hot spots* do framework, ou seja, devem ser configurados com as funcionalidades específicas desejadas para uma determinada instanciação do framework.

3.2.1. *Frozen Spot* do Framework

Os *frozen spots* do framework proposto são:

(a) *Gerente de Regras*: responsável por todo o controle do processo de definição, análise, distribuição e execução das regras. É composto pelo frozen spot *Definidor de Regras* e pelos hot spots *Analisador de Eventos, Gerente de Distribuição e Administrador de Execução*;

(b) *Definidor de Regras:* chamado sempre que uma nova regra é definida, alterada ou excluída. É responsável pela atualização das informações apropriadas no catálogo do sistema. É quem interage com o gerente de dicionário para escrever as regras.

3.2.2. *Hot Spots* do Framework

Os *hot spots* do framework proposto, detalhados a seguir são os framelets: *Analisador de Eventos, Gerente de Distribuição e Administrador de Execução*.

3.2.2.1. Framelet Analisador de Eventos

Este componente é responsável por detectar e avaliar os eventos responsáveis pela ativação das regras, no ambiente de banco de dados e de computação móvel, e por avaliar as condições associadas a elas. O framework pode ser configurado para analisar diferentes tipos de eventos,

como eventos de banco de dados (decorrentes de modificação ou recuperação de dados), eventos típicos do ambiente móvel, eventos temporais e eventos definidos por aplicações.

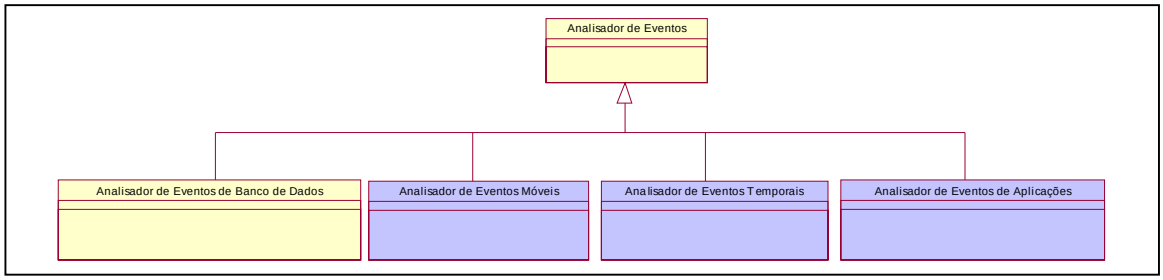

Figura 4: *Framelet* Analisador de Eventos

- **Analisador de Eventos de Banco de Dados:** detecta e avalia eventos de banco de dados. Um evento de banco de dados pode ser especificado como uma operação de inserção (*insert*), exclusão (*delete*), modificação (*update*) ou recuperação (*select*) sobre sua base de dados.

- **Analisador de Eventos Móveis**: detecta e avalia eventos do tipo móvel. Um evento móvel pode especificar regras para acompanhamento de unidades móveis, tais como mudança de célula, interrupção na comunicação unidade móvel-estação de base ou estação de base-host fixo, resposta à unidade móvel por outra estação de base, dentre outros.

- **Analisador de Eventos Temporais**: detecta e avalia eventos do tipo temporal. Um evento temporal pode especificar que uma regra seja disparada (*triggered*) ou em um determinado momento, ou repetidamente, ou em intervalos periódicos de tempo como, por exemplo, a cada 10 minutos.

- **Analisador de Eventos de Aplicação**: detecta e avalia eventos definidos por uma aplicação (programa). Eventos são definidos e as aplicações de tempo em tempo notificam ao framework a ocorrência do evento. Assim, as regras que especificam essa condição são disparadas.

Para integrar um novo tipo de *analisador de eventos* à arquitetura do framework, é necessário apenas definir uma nova classe especializada da classe *Analisador de Eventos*, conforme apresentado na figura 4.

3.2.2.2. Framelet Gerente de Distribuição

Este componente é responsável pela execução de regras em um ambiente distribuído e/ou móvel. Ele também é um framework, logo possui seus próprios *hot* e *frozen spots*. Seus componentes são apresentados na Figura 5.

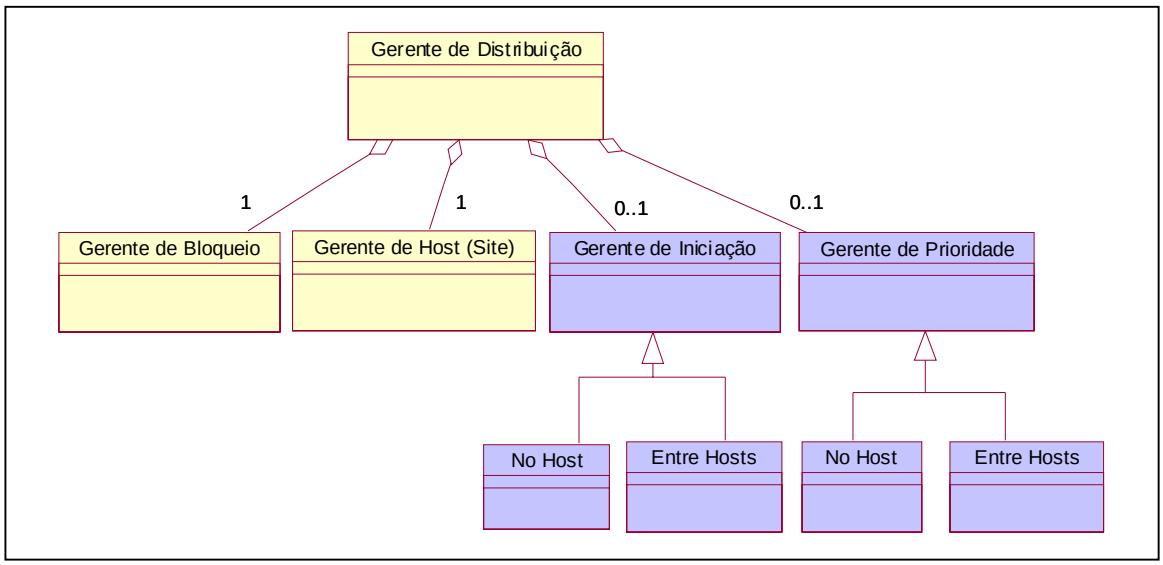

Figura 5: Framelet *Gerente de Distribuição*

Os componentes *Gerente de Distribuição*, *Gerente de Bloqueio*, *Gerente de Host (Site)* correspondem aos *frozen spots* do framelet, ou seja, conservam-se inalteráveis em qualquer de suas instanciações. Os componentes *Gerente de Iniciação e Gerente de Prioridade* correspondem aos *hot spots* do framelet. Eles só deverão ser integrados ao framework caso o processamento das regras possa ser executado em cada *Host* (*site)* independentemente dos demais, o que acarretará na necessidade de controles extras de início das transações das regras e do estabelecimento de prioridades de execução das regras entre os diversos *sites*. Caso tais componentes não sejam integrados ao framework, o processamento das regras só poderá acontecer no mesmo host *(site*) onde ela for disparada.

- **Gerente de Bloqueio:** Determina e controla os protocolos de bloqueio utilizados para garantir a consistência e integridade dos dados nos bancos de dados distribuídos e/ou móveis.

- **Gerente de Host (Site):** Determina e controla a possibilidade que as sub-transações geradas pelas regras possam ler e atualizar dados em outros hosts (sites) remotos e móveis.

- **Gerente de Iniciação**: Determina e controla a possibilidade de que as sub-transações geradas pelas regras sejam iniciadas, mesmo quando a transação de disparo da regra puder ler subseqüentemente ou modificar dados naquele host (site). Pode ser configurado para tratar os seguintes tipos de iniciação:

  - **No Host**: controla o início das transações apenas no host (site) onde a regra deverá ser executada.

- **Entre Host**: controla o início da execução das regras entre os hosts (sites), considerando outras transações em estado de espera para execução nos diferentes hosts (sites).

- **Gerente de Prioridade:** Determina e controla a aplicação de prioridades nas sub-transações geradas pelas regras em hosts (sites) distintos. Pode ser configurado para tratar os seguintes tipos de prioridade:

  - **No Host**: controla a prioridade de execução das transações apenas no host (site) onde a regra deverá ser executada.

  - **Entre Host**: controla a prioridade de execução das regras entre os hosts (sites), considerando outras transações em estado de espera ou execução nos diferentes hosts (sites).

3.2.2.3. Framelet Administrador de Execução

Este componente é responsável pela identificação da forma como são geraradas as sub-transações do banco de dados, da geração do plano para a execução das regras e de sua execução, bem como das ações externas ao banco de dados solicitadas pelas regras e eventos externos ao banco de dados. Ele é composto pelos componentes *Executor de Regras de Banco de Dados e Executor de Ações Externas ao Banco de Dados.*

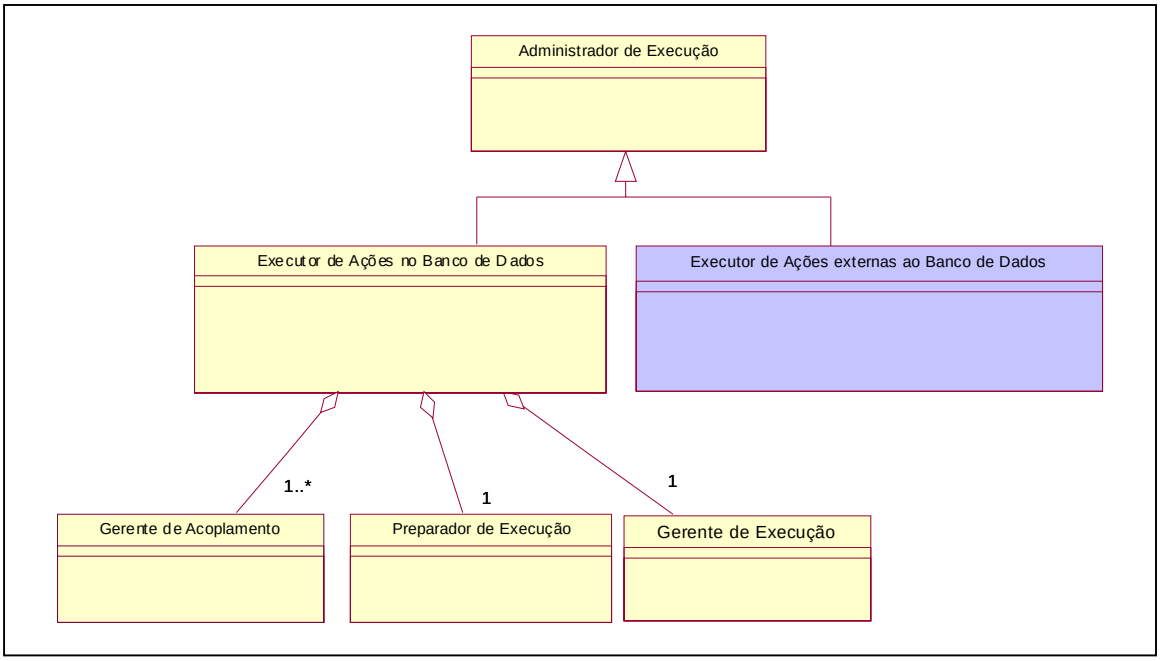

Figura 06: *Framelet* Administrador de Execução.

- **Executor de Regras de Banco de Dados:** é o responsável pela execução de todas as regras que envolve o banco de dados. Seus componentes são:

- **Gerente de Acoplamento**: É o responsável pela identificação de como as sub-transações ou transações das regras devem ser executadas. É quem interage com o Gerente de Transação do SGBD para solicitar a execução das ações relativas às regras, quando as mesmas são de banco de dados. O framework pode ser configurado para tratar diferentes tipos de acoplamento, quanto ao momento de execução de suas ações, como execução de forma imediata, postergada ou desacoplada do corpo da transação que disparou a regra.

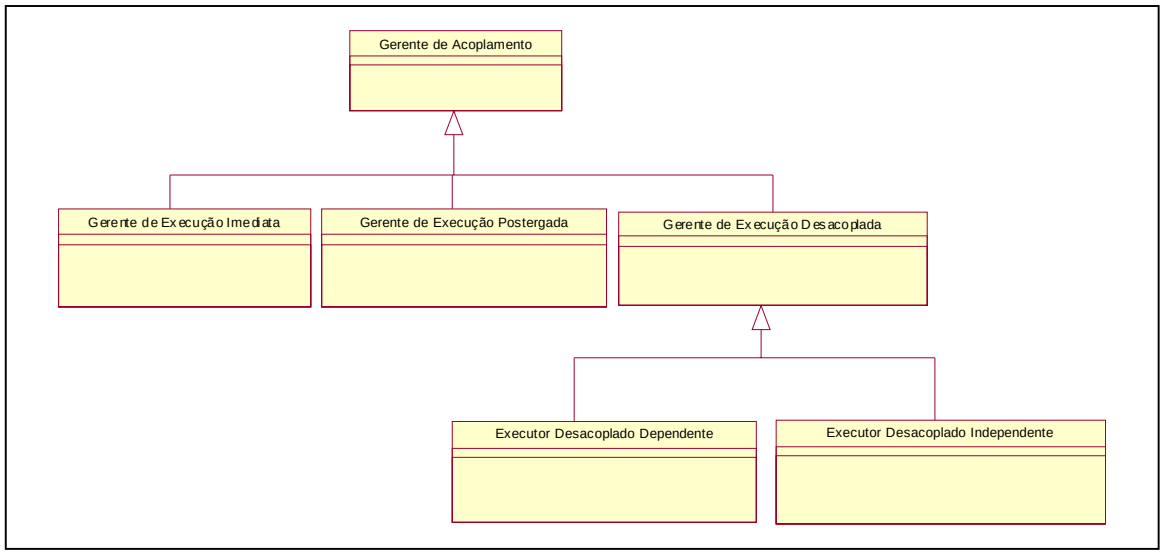

Figura 7: *Componente* Gerente de Acoplamento

  - **Gerente de Execução Imediata**: solicita a execução das ações da regra imediatamente (*immediate*) a seguir à avaliação ou disparo da regra, dentro da mesma transação que a disparou;

  - **Gerente de Execução Postergada**: solicita a execução das ações da regra no ponto de commit da transação que a disparou (deferred);

  - **Gerente de Execução Desacoplada**: solicita a execução das ações da regra em uma transação em separado (decoupled) da que a disparou. A transação para execução da regra pode ser criada logo após o commit da transação que a disparou, situação tratada pelo Executor Desacoplado Dependente, ou pode ser criada independentemente do commit da transação que a disparou, situação tratada pelo Executor Desacoplado Independente.

Para integrar um novo tipo de modo de execução das ações da regra à arquitetura do framework, é necessário apenas definir uma classe especializada da classe *Gerente de Acoplamento*, conforme a figura 7.

- **Preparador de Execução**: é o responsável por preparar os códigos para execução das regras. Cria uma árvore gramatical, descrevendo o comando do usuário; e manuseia as regras implementadas, combinando a árvore gramatical gerada pela consulta do usuário com a regra apropriada, gerando uma nova árvore gramatical para execução.

- **Gerente de Execução**: É o responsável pela geração do plano de execução da transação no banco de dados e seu controle e execução propriamente dita. É

quem interage com o Gerente de Transação do SGBD para solicitar a execução das transações de banco de dados. Possui os componentes Gerador do Plano de Execução e Executor de Regras.

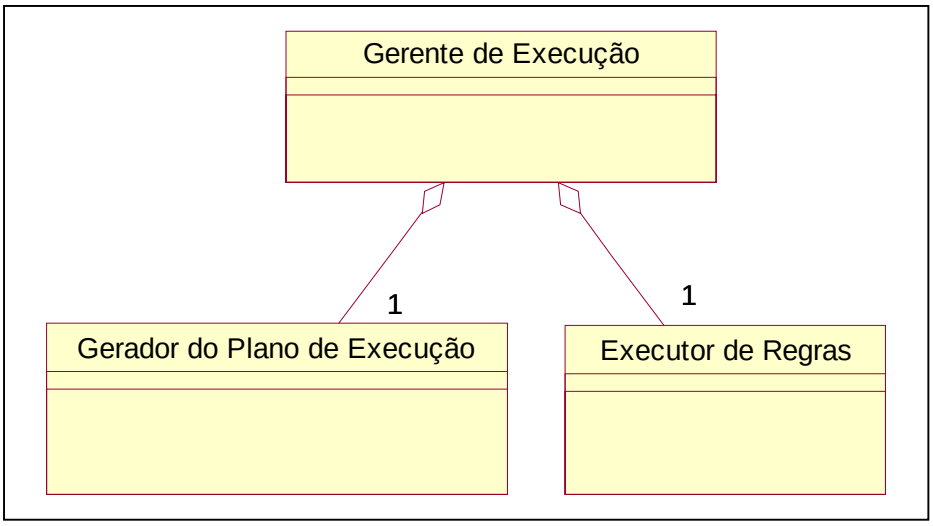

Figura 8: *Componente* Gerente de Execução

- **Gerador do Plano de Execução**: é o responsável pela geração do plano de execução das regras no banco de dados, inclusive para o ambiente móvel e distribuído.

- **Executor de Regras**: é o responsável pela execução dos planos de execução das regras no banco de dados, inclusive para o ambiente móvel, garantindo as propriedades de consistência e integridade do ambiente de banco de dados móvel e distribuído.

- **Executor de Ações Externas ao Banco de Dados:** é o responsável pela execução de todas as ações que não envolvem a utilização de transações de banco de dados, tais como enviar mensagens para uma unidade móvel, reconectar uma unidade móvel a uma estação de base, manter o serviço quando de uma mudança de célula, entre outras ações.

4. Instanciação do Framework

Para exemplificar o uso do Framework, será apresentada sua instanciação para a construção de um ambiente específico denominado SBD-M-RA-1 (figuras 9-a, 9-b e 9-c).

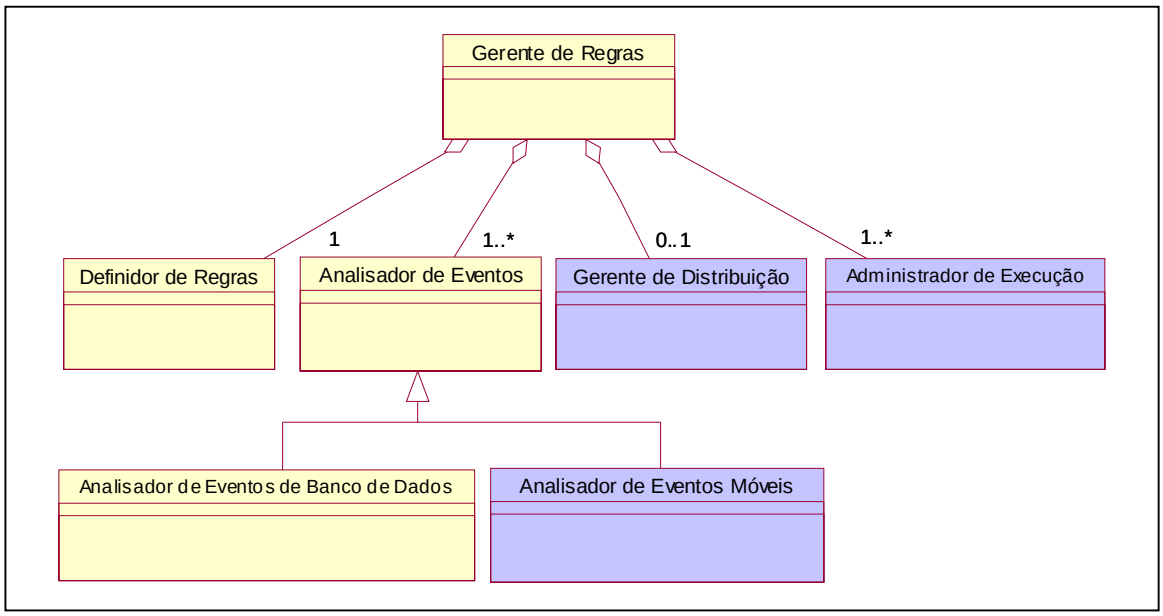

Figura 9-a: Instanciação do *Hot Spot* Analisador de Eventos

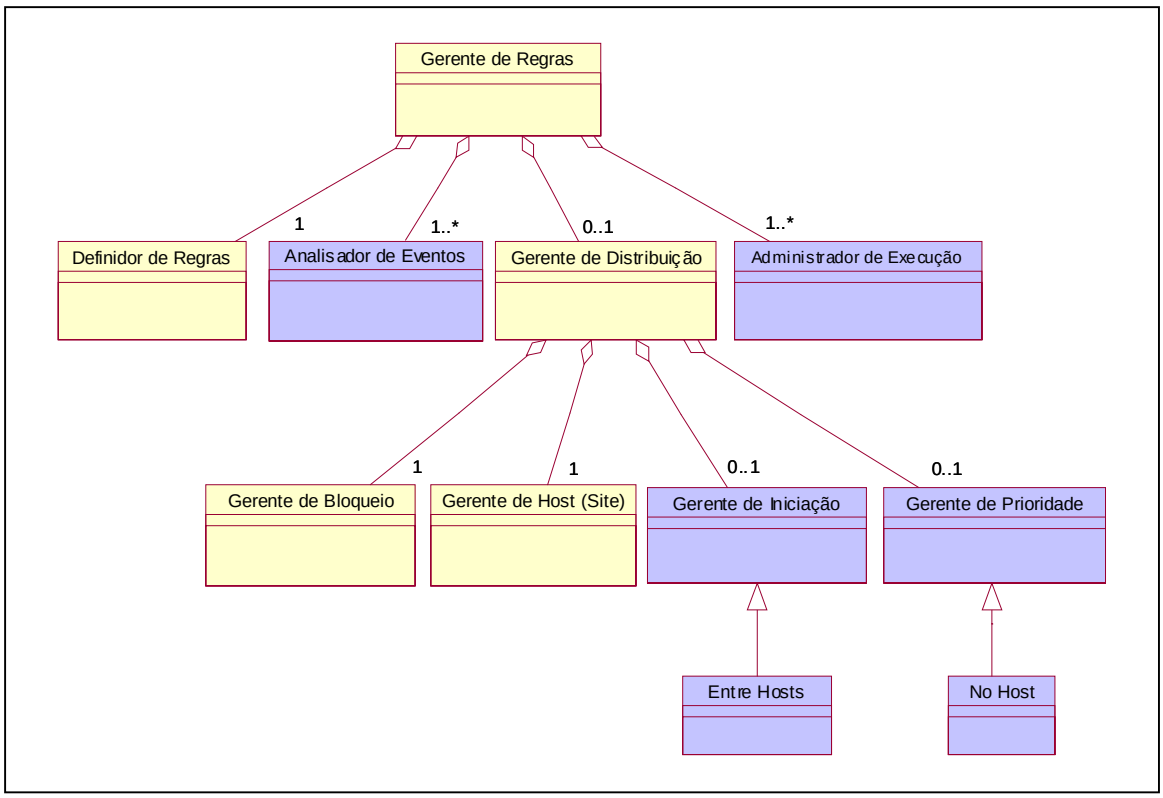

Figura 9-b: Instanciação do *Hot Spot* Gerente de Distribuição

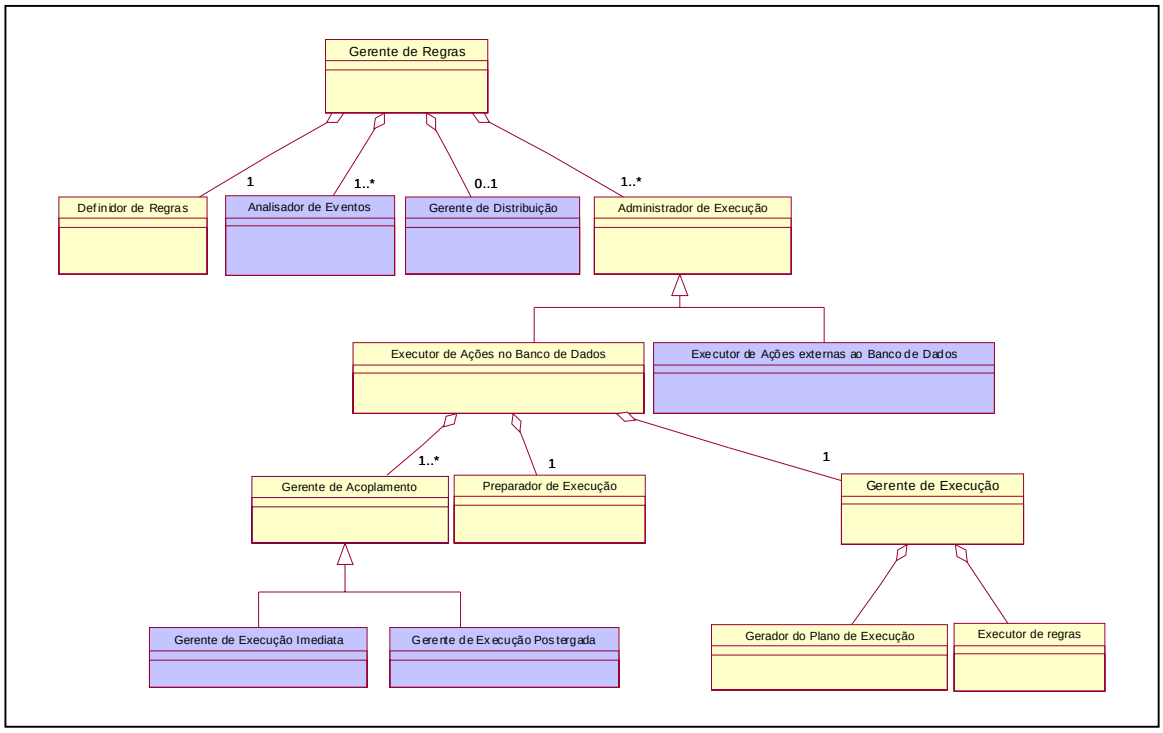

Figura 9-c: Instanciação do *Hot Spot* Administrador de Execução

Na instanciação do Framework apresentada nas figura 9-a, 9-b e 9-c, os *Hot Spots* foram instanciados para satisfazer a um SBD-M-RA 1com as seguintes características:

- **Analisador de Eventos**: Somente eventos de transição de banco de dados e do ambiente móvel serão detectados.

- **Gerente de Distribuição/Gerente de Iniciação**: Um gerente coordenará o momento da iniciação da execução das sub-transações geradas pelas regras entre os diferentes *host (site)*.

- **Gerente de Distribuição/Gerente de Prioridade:** Cada *host* criará seu próprio *scheduler* (escalonador) para determinar a prioridade de execução das sub-transações geradas pelas regras.

- **Administrador de Execução/Executor de Ações externas ao Banco de Dados**: Serão executadas toas as ações externas ao Banco de dados, conforme detecção e avaliação do componente *Analisador de Eventos*.

- **Administrador de Execução/Executor de Ações no Banco de Dados/Gerente de Acoplamento**: As regras serão executadas imediatamente (*immediate*) a seguir à avaliação ou disparo da regra, dentro da mesma transação que a disparou ou no ponto de commit da transação que a disparou (deferred).

5. Conclusões e Trabalhos Futuros

A utilização da técnica de framework na construção de SBD-M-RA se apresenta como uma alternativa muito eficiente, principalmente em função do reaproveitamento do código das funcionalidades que sejam comuns, enquanto possibilita implementações específicas para diversos ambientes operacionais, com diferentes semânticas de regra.

O objetivo desse trabalho foi propor um framework de banco de dados com regras ativas para um ambiente móvel que possibilite a construção de ambientes de dados que possam diferir entre si em função da forma como controlam e administram a execução das regras, o modo de detectar seus eventos, bem como quanto ao aspecto da mobilidade das unidades móveis, suas necessidades, controlando-as diretamente no banco de dados.

O framework proposto está contido na arquitetura de um SGBD convencional e o estende com características ativas, distribuídas e móveis, que podem ser configuradas conforme o interesse.

Diversos outros trabalhos podem dar continuidade a esse estudo, tais como o desenvolvimento de um framework para um SBD-M-RA com a tecnologia de agentes de softwares em cada componente, de um framework Paralelo com Regras Ativas para o ambiente móvel, o desenvolvimento dos componentes de terminação e confluência das regras, além da própria implementação do framework proposto utilizando-se de agentes de softwares.

Referências Bibliográficas

[01] Andry Rakotonirainy; “Trends and Future of Mobile Computing”; Proceedings of the DEXA, 1999, pp 136-140.

[02] Markus Endler; Francisco José da Silva e Silva, “Requisitos e Arquiteturas de Software para Computação Móvel”, USP, 2000

[03] Ana Paula Afonso; Francisco S. Regateiro, Mário J. Silva “Dynamic Data Delivery to Mobile Users”; Proceedings of the DEXA, 1999, pp 121-126.

[04] Lars Frank, “Atomicity Implementation in Mobile”; Proceedings of the DEXA, 1999, pp 105-113.

[05] KamYiu Lam, Tei-Wei Kuo, Wai-Hung Tsang, Gary C.K.Law “Concorrency Control in Mobile Distributed Real-Time Database Systems”; Information Systems Vol. 25, 2000, Nº 4, pp 261-286.

[06] M. M. Gore, R. K. Ghosh, “Recovery of Mobile Transaction”; Proceedings of the DEXA, 2000, pp 23-27.

[07] Steffen Lipperts, “On the Efficiente Deployment of Mobility in Distributed System Management”; Proceedings of the DEXA, 2000, pp 193-197.

[08] Subhasish Mazumdar, Panos K. Chrysanthis, “Achieving Consistency in Mobile Databases through Localization in PRO-MOTION”; Proceedings of the DEXA, 1999, pp 82-89.

[09] David Ratner, Peter Reiher, Gerald J. Popek “Roam: A Sacalable Reprlication System for Mobile Computing”; Proceedings of the DEXA, 1999, pp 96-104.

[10] Paulo Jorge Marques, Luís Moura Silva, João Gabriel Silva, “A Flexibilite Mobile-Agent Gramework for Accessing Information Systems in Disconnected Computing Environments”; Proceedings of the DEXA, 2000, pp 173-177.

[11] Jeong-Joon at all, “Casting Mobile Agents to Workflow Systems: On Performance and Scalability Issues”; DEXA, 2001, Springer-Verlag Berlin Heidelberg 2001, pp 217-263.

[12] Mikhail N. Mikhail, Baher A. esmat, AMR A. El-Kadi, “A New Architecture for Mobile Computing”; SCI2001.

[13] Hans Fritsch, Stella Gatziu, “A Reusable Architecture to Construct Active Database”; Technical Report, 1999, Institut für Informatk,.Universitat Zürich

[14] Shiow-yang, Chun-Shun Chang, “An Active Database Framework for Adaptive Mobile Data Access”; *Workshop on Mobile Data Access, 17th International Conference on Conceptual Modeling, Singapore, 1998. Also in Yahiko Kambayashi, Dik Lun Lee, Ee-Peng Lim, Mukesh Kumar Mohania, and Yoshifumi Masunaga (Eds.) Advances in Database Technologies, LNCS 1552, pages 335-346, Springer-Verlag, 1999.*

[15] Patricia Serrano-Alvarado, Claudia L. Roncancio, Michel Adiba “Analyzing Móbile Transaction Suport for DBMS”; DEXA, 2001, Berlin Heidelberg 2001, pp 595-600

[16] Richard Vlach, “Mobile Database Procedures in MDBAS”; DEXA, 2001, Berlin Heidelberg 2001, pp 559-563

[17] Stefano Ceri e Jennifer Widom; “Active Database Systems Triggers and Rules for Advanced Database Processing”; Morgan Kaufmann, 1996.

[18] Ramez Elmasri e Shamkant Navathe; “Fundamentals of Database Systems”; Addison-Wesley, 1999, 3rd Edition.

[19] M.T. Özsu e P. Valduriez; “Principles of Distributed Database Systems”; Prentice Hall, 1999.

[20] Norman W. Paton; “Active Rules in Database System”; Springer, 1998.

[21] W. Pree; “Framelets as Basis of Lean Component Architectures”; Handouts da palestra proferida na PUC-Rio, 1999.

[22] Sérgio da Costa Côrtes; “Regras Ativas em Sistemas de Banco de Dados: Sintaxe, Semântica e Processamento Distribuído”. Dissertação de Mestrado, Departamento de Informática, PUC-Rio, Brasil, 1999.

[23] E.M.A. Uchôa; “Framework para Integração de Sistemas de Bancos de Dados Heterogêneos”. Tese de Doutorado, Departamento de Informática, PUC-Rio, Brasil, 1999.