# Sistemas de Gerência de Banco de Dados baseados em Agentes para um Ambiente de Computação Móvel

Sérgio da Costa Côrtes[1]
scortes@inf.puc-rio.br

Sérgio Lifschitz[2]
sergio@inf.puc-rio.br

PUC-RioInf.MCC11/02 Junho, 2002

## Abstract

Agent-based databases enable the development of Database Management Systems (DBMSs) that may be configured and extended to give support to new data storage and retrieval requirements. For database management systems to be used in a mobile computing environment, the properties of reusability, autonomy, interaction and mobility can be obtained by the use of a software agent. This paper presents a proposal for a layered architecture in a mobile computing environment. The focus of the architecture is on the use of software agents in the server layer of the database, using a framework for the construction of the management system that will include the functionalities necessary for the mobile computing environment. A detailed discussion of DBMS architecture is also presented.

**Keywords**: mobile, agents, framework, framelet, database, active rule, distributed database system

## Resumo

Bancos de dados baseados em agentes possibilitam o desenvolvimento de Sistemas de Gerência de Banco de Dados – SGBDs que podem ser configurados e estendidos para dar suporte a novas necessidades de armazenamento e recuperação de dados. Para sistemas de gerência de banco de dados que serão utilizados em um ambiente de computação móvel, as propriedades de reusabilidade, autonomia, interação e mobilidade, podem ser obtidas através do uso de agente de software. Este trabalho apresenta uma proposta de uma arquitetura em camadas para um ambiente de computação móvel. A foco da arquitetura está no uso de agentes de softwares na camada servidora do banco de dados, utilizando um framework para construção do sistema de gerência de banco de dados que contemplará as funcionalidades necessárias para o ambiente de computação móvel. Uma discussão detalhada de arquiteturas de SGBDs também é apresentada.

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

---

[1] *Doutorando, parcialmente apoiado Fundação IBGE e bolsista PUC-Rio*
[2] *Parcialmente apoiado por bolsa de pesquisa do CNPq 300048/94-7*

# 1. Introdução

Uso da tecnologia de agentes de software no desenvolvimento de Sistemas de Gerência de Banco de Dados (SGBD) permite que estes sejam configurados e estendidos para suportar o desenvolvimento de novas funcionalidades necessárias às aplicações atuais [38]. Para os SGBDs utilizados em um ambiente de computação móvel (*Mobile Database*), propriedades como autonomia, interação e mobilidade são fundamentais e podem ser mais bem desenvolvidas quando se empregam agentes de software em seu desenvolvimento, uma vez que as principais características de um agente é a sua habilidade de agir de forma autônoma, sua mobilidade e capacidade de reação.

A computação móvel pode ser considerada como uma variação da computação distribuída [19]. Assim sendo, questões relativas à gerência de dados distribuídos podem também ser aplicadas a bancos de dados móveis, tais como distribuição e replicação de dados, modelos de transações, processamento de consultas, recuperação e tolerância a falhas, bem como seu projeto de bancos de dados, todos analisados de forma especial [18, 04, 05 e 06].

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 Ativos (SGBDA) podem ser empregados em ambientes móveis, utilizando seu comportamento ativo no sentido de resolver alguns problemas inerentes a este ambiente.

O uso da técnica de framework para a construção de Sistemas de Gerência de Bancos de Dados para um ambiente de computação móvel permite o desenvolvimento de diferentes instâncias do SGBD, re-aproveitando as funcionalidades que sejam comuns e permitindo implementações específicas para diversos ambientes operacionais.

A área de pesquisas de agentes de software visa construir sistemas para ambientes heterogêneos, distribuídos e dinâmicos [41]. A tecnologia de agentes tem sido usada em diversos ambiente complexos [28, 36]. Existem muitos domínios de aplicações para Sistemas de Gerência de Banco de Dados – SGBDs, os quais podem ser estendidos e configurados.

Este trabalho estuda um conjunto de arquiteturas para diferentes tipos e modelos de SGBDs e propõe uma arquitetura em camadas para um sistema de gerência de banco de dados que poderá ser utilizado em um ambiente de computação móvel ambiente, onde seu foco principal está no uso de um framework para desenvolvimento do SGBD e a utilização de agentes de software para sua construção, atendendo aos requisitos da computação móvel.

## *1.1. Trabalhos Relacionados*

Administração de dados para computação móvel é uma área de muito interesse atualmente. Diversas arquiteturas e frameworks estão sendo propostos e desenvolvidos [03, 10, 12, 14, 30] para diferentes aplicações. A tecnologia de agentes também já é muito utilizada [07, 10, 11, 29, 30]. Entretanto, a combinação das tecnologias de framework, agentes de software e dos SGBD com comportamento ativo/reativo ainda é pouco explorado no ambiente da computação móvel.

Em [14] é proposto um framework e uma arquitetura de sistemas para construção de um sistema de informações móvel utilizando-se um SGBDA, porém, o módulo de detecção de eventos é estático, previamente especificado por variáveis do ambiente computacional, limitando sua capacidade de perceber mudanças no ambiente móvel, reagir e controlar a entrega dos dados ao usuário móvel solicitante.

Idéias similares a este trabalho foram discutidas no projeto DEGAS [29], quando diversas vantagens foram mostradas na utilização de agentes em SGBDA, entretanto o conceito de frameworks e a utilização em um ambiente de computação móvel não foram considerados.

O sistema ACADEMIA [30] é utilizado para ilustrar e descrever uma arquitetura para combinar banco de dados e agentes, visando o descobrimento e extração de informações acessíveis via Web. Embora muito interessante, diferentemente do nosso trabalho, não contempla características de um framework e de um ambiente de computação móvel.

A flexible Mobile-Agent Framework é proposto em [10] para acessar dados em um ambiente de computação desconectado, porém, não explora a utilização do SGBD com capacidade de controle e reação automática, limitando-se apenas ao controle de envio e recebimento de mensagens, através de um Mobility Component. Diferentemente do nosso trabalho, explora características inerentes ao ambiente móvel somente para acesso aos dados via aplicação, não utilizando a capacidade de detecção e reação do SGBD, passível de generalidade para todo o ambiente.

Quanto à computação móvel, tendências e o futuro da computação móvel são apresentados em [01] reforçando e motivando o desenvolvimento desse trabalho, 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] e está sendo incorporado no projeto de implementação. 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 influenciam 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]. Todos esses trabalhos estão relacionados com o projeto que será apresentado e contribuem diretamente na sua implementação.

Não foi encontrado na literatura qualquer trabalho que indique o uso das tecnologias de framework, sistemas de agentes, SGBD com comportamento ativo/reativo, combinados com o intuito de resolver os problemas de ambientes da computação móveis, conforme proposto neste trabalho.

## *1.2. Organização desta monografia*

Este trabalho está organizado da seguinte forma. Na seção 2, é apresentado um estudo detalhado sobre arquiteturas para SGBDs, contemplando as arquiteturas convencionais, distribuídas e com regras ativas. Na seção3 são apresentadas três abordagens para construção de um SGBD baseado em agentes de software. Na seção 4 são apresentados os sistemas de computação móveis e nas seções 5 e 6 a conceituação necessária para o desenvolvimento do framework e sua implementação com agentes de software é descrita. Na seção 7 apresentamos a arquitetura proposta para construção de um SGBD para o ambiente de computação móvel. Na seção 8 o framework para desenvolvimento do SGBD com seus principais componentes e seus *hot spots*. Na seção 9 descrevemos a arquitetura de agentes de software para o componente de regras ativas do SGBD. Finalmente, na seção 10, apresentamos nossos comentários finais sobre este trabalho.

# 2. Arquiteturas de Sistemas de Gerência de Banco de Dados – SGBD convencional, distribuído e ativo.

A arquitetura de um SGBD determina suas funcionalidades e que componentes são necessários para realizá-las. Um Sistema de Gerência de Banco de Dados - SGBD convencional é um conjunto de softwares altamente complexo e sofisticado que possibilita, basicamente, os serviços de armazenamento, recuperação e gerência de dados. [08,18]. Um Sistema de Gerência de Banco Distribuído – SGBDD é um sistema de software que permite o gerenciamento do banco de dados distribuído, tornando esta distribuição transparente para seu usuário [19]. Os Sistemas de Gerência de Banco de Dados Ativos - SGBDA 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]. Existe uma grande diversidade de arquiteturas de SGBD, SGBDD e SGBDA , no entanto, é possível identificar um conjunto básico de funcionalidades que normalmente consta das mais diferentes arquiteturas.

Nesta seção, será apresentada uma arquitetura genérica de um SGBD convencional, de um SGBDD, identificando seus componentes e os relacionamentos existentes entre estes. Também é discutida a questão do comportamento ativo em SGBD.

## *2.1. Arquitetura de um SGBD convencional*

Um SGBD é composto por diversos módulos (*componentes*) de softwares, cada qual com uma função operacional específica. Muitas das funcionalidades dos SGBD são suportadas por funções dos sistemas operacionais, tais como exclusividade de acesso a arquivos, prioridades de execução de processos, entre outras. Assim sendo, os SGBD devem ser construídos como camadas acima dos sistemas operacionais, visando utilizar esses serviços [08,18]. Os principais componentes da arquitetura de software de um SGBD convencional estão apresentados na Figura 1, extraída de [08]. Esta arquitetura também mostra quais são as interfaces entre tais componentes e entre estes e alguns componentes externos ao SGBD, tais como os módulos de Consultas dos Usuários e Método de Acesso (este pertencente ao sistema operacional).

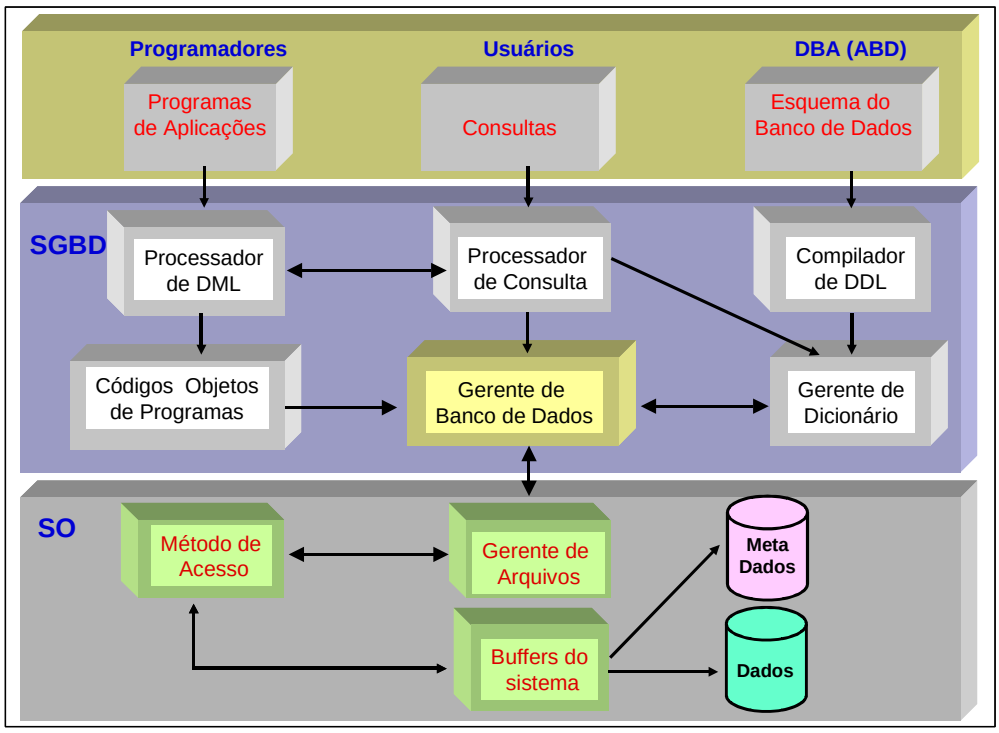


**Figura 1:** Uma arquitetura genérica para um SGBD.

Definição dos componentes de um SGDB convenciona:

- ***Processador de Consulta (query processor)***: Este é o principal componente do SGBD. É responsável por transformar as consultas submetidas ao SGBD em uma série de instruções de baixo nível e direcioná-las ao Gerente de Banco de Dados.

- ***Gerente de Banco de Dados (database manager)***: É a interface do SGBD com os programas de aplicações e consultas dos usuários. Recebe solicitações relativas à submissão de consultas e programas de aplicações e examina os esquemas externo e conceitual para determinar que registros conceituais são necessários para satisfazê-las. Este módulo chama o Gerente de Arquivos para atender e executar as suas solicitações.

- ***Gerente de Arquivos (file manager)***: Manipula os arquivos para armazenamento e gerencia a alocação dos espaços em disco. Estabelece e mantém uma lista com as estruturas e índices definidos no esquema interno do banco de dados. Se o sistema utiliza arquivos *hashed*, ele chama funções de *hashing* para gerar os endereços dos arquivos. Entretanto, o Gerente de Arquivos não gerencia diretamente os *inputs* e *outputs* físicos dos dados. Ele necessita de um método de acesso apropriado, o qual tanto lê quanto grava dados no *buffer* do sistema.

- ***Compilador de DML*** – *Data Manipulation Language* (***DDL compiler***): Converte os comandos da linguagem de consulta embutidas em programas de aplicações em funções padrões, através de chamadas ao host da linguagem. Interage com o processador de consulta para gerar o código apropriado para a consulta.

- ***Compilador de DDL*** – *Data Definition Language* (***DDL compiler***): Converte os comandos de definição de dados num conjunto de tabelas contidas no catálogo do sistema, ou seja, no seu metadados. Essa tabelas são armazenadas no dicionário de dados do sistema.

- ***Gerente de Dicionário*** (***dictionary manager***): É responsável pelo acesso e manutenção do dicionário de dados. É freqüentemente acessado por muitos componentes do SGBD.

Conforme visto anteriormente, o gerente de banco de dados faz a interface do SGBD com os programas de aplicações e consultas dos usuários. A figura 2 apresenta seus componentes, todos descritos logo a seguir.

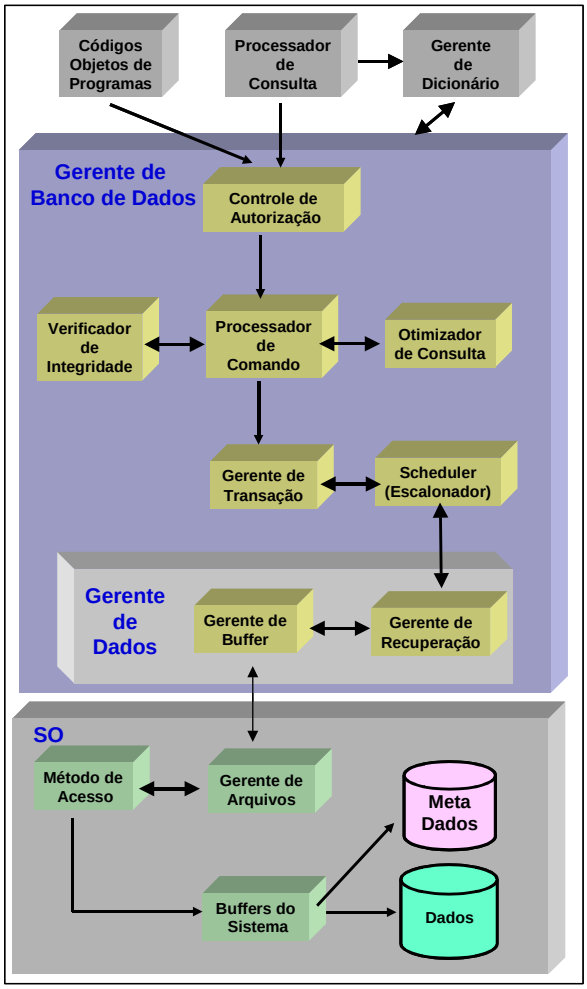


**Figura 2:** Sub-componentes do componente Gerente de Banco de Dados

- ***Controle de Autorização***: Este módulo verifica se o usuário possui a necessária autorização sobre o recurso solicitado para executar a operação desejada.

- ***Processador de Comando***: Após a verificação e certificação de que o usuário possui a autorização necessária para utilizar o recurso, o controle é passado para este módulo que coordenará a execução da instrução.

- ***Verificador de Integridade***: Para as operações que alteram o banco de dados, este módulo verifica se a operação desejada irá satisfazer todas as restrições do banco de dados, tais como integridade referencial, chaves primárias e domínios de atributos (check constraints), entre outras.

- ***Otimizador de Consulta***: Este módulo determina a estratégia ótima para execução da consulta.

- ***Gerente de Transação***: Este módulo executa todo o processamento solicitado pelas operações recebidas das transações.

- ***Scheduler (Escalonado)***: Este módulo é responsável por assegurar que as operações processadas no banco de dados ocorram sem conflitar com outras operações. Este controle é relativo à ordem em que as operações das transações são executadas, garantindo a seriação das mesmas.

- ***Gerente de Recuperação***: Este módulo é responsável por assegurar que o banco de dados permanece em um estado consistente em casos de falhas lógicas ou físicas no sistema. É responsável por efetuar o commit (confirmação) ou rollback (cancelamento) da transação.

- ***Gerente de Buffer***: Este módulo é responsável por transferir os dados entre a memória principal e o armazenamento secundário, tal como disco ou fita. Os módulos Gerente de Recuperação e Gerente de Buffer são freqüentemente referenciados, coletivamente, como Gerente de Dados (Data Manager).

Além dos módulos citados, existem muitas estruturas de dados que são necessárias como parte da estrutura física para implementação dos bancos de dados. Essas estruturas incluem arquivos de dados e índices e dicionários de dados.

## *2.2. SGBD Distribuídos - SGBDD*

Um BDD é uma coleção de dados que pertencem logicamente ao mesmo sistema, mas está fisicamente espalhado (distribuído) pelos *sites* de uma rede de computador. Logo, um BDD é uma coleção de vários bancos de dados distribuídos em vários *sites*, interligados por uma rede de comunicação de dados. Um Sistema Gerenciador de Banco de Dados Distribuídos (SGBDD) é definido como um sistema que permite a administração e acesso eficiente de um BDD, tornando esta distribuição transparente para o usuário [19].

Considerando que o conjunto de *sites* não é vazio e que cada *site* onde reside um BDD possui capacidade de processamento independente de outro *site*, caso aconteça uma ruptura na comunicação do *site* com a rede de tele-processamento, o banco de dados continua operando e suprindo as necessidades locais. A figura 3 a seguir exemplifica um esquema de um banco de dados distribuído.

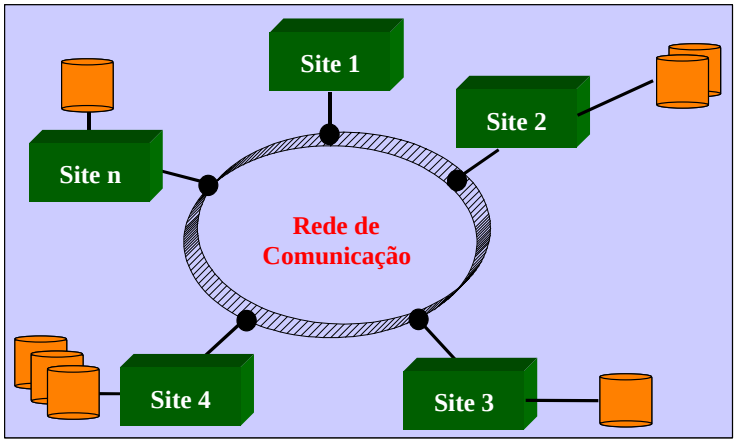


**Figura 3**: Ambiente de um Banco de Dados Distribuído

Os dados em um BDD são fisicamente distribuídos nos vários *sites* e podem ser fragmentados e replicados. A fragmentação divide o banco de dados em unidades lógicas por diversos *sites* enquanto que a replicação permite que os mesmos dados sejam armazenados em mais de um *site*. Estas técnicas são muito utilizadas durante o projeto do BDD. A informação sobre a fragmentação, alocação dos dados nos *sites* específicos e a replicação são armazenadas em um catálogo do sistema global, fundamental para localização dos dados em um ambiente distribuído.

### 2.2.1. Arquiteturas para um SGBDD

Existem muitas possibilidades de arquiteturas para que um SGBDD implemente as soluções distribuídas. A arquitetura Cliente/Servidor é uma delas, muito utilizada atualmente em sistemas comerciais, onde muitos clientes acessam um único servidor de banco de dados. A arquitetura de um banco de dados distribuído (*peer-to-peer)* permite termos um único banco de dados, com um único esquema global, distribuído por vários sites. Já a arquitetura com vários SGBDs (*multi databases*), em que muitos clientes acessam muitos servidores de banco de dados é mais flexíveis, desde que os bancos de dados estejam distribuídos através dos múltiplos servidores. Um estudo mais detalhado sobre as arquiteturas para um SGBDD pode ser encontrado em [19].

#### 2.2.1.1. Arquitetura Cliente/Servidor

A idéia da arquitetura cliente/servidor é dividir as funcionalidades que precisam ser fornecidas aos usuários do SGBD em duas camadas, a camada *servidor* e a camada *cliente*, ambas com funcionalidades distintas. Essa arquitetura facilita o gerenciamento da complexidade dos SGBDs atuais. Conforme mostra a figura 4, a camada servidor faz a maior parte do trabalho de gerenciamento de dados, enquanto que a camada cliente cuida da interface do usuário e da aplicação, além de administrar uma "*memória*" local para solicitação e armazenamento do resultado das consultas.

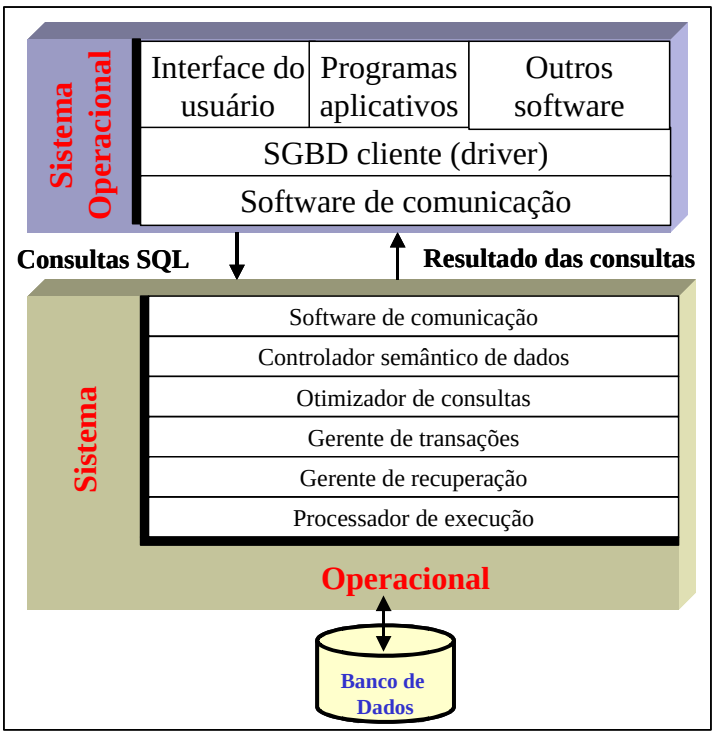


**Figura 4:** Componentes da arquitetura Cliente/servidor extraída de [19]

Diferentes tipos de arquiteturas cliente/servidor podem ser implementadas. A arquitetura mais simples é aquela em que vários clientes acessam um único servidor, denominada *múltiplos clientes-servidor único*. A arquitetura mais complexa é aquela em que o ambiente possui *múltiplos clientes-multiplos servidores.*

### 2.2.1.2. Arquitetura de um Banco de Dados Distribuído

Conforme visto anteriormente, os dados em um banco de dados distribuído podem ser fragmentados e replicados. Assim, em cada site teremos um esquema local do seu banco de dados. Para garantir as propriedades de transparência de localização e independência de dados, cria-se um esquema global do banco de dados que também descreverá a estrutura lógica integrada dos dados em todos os sites, possibilitando visões externas para cada aplicação dos usuários. A figura 5 a seguir descreve os componentes dessa arquitetura.

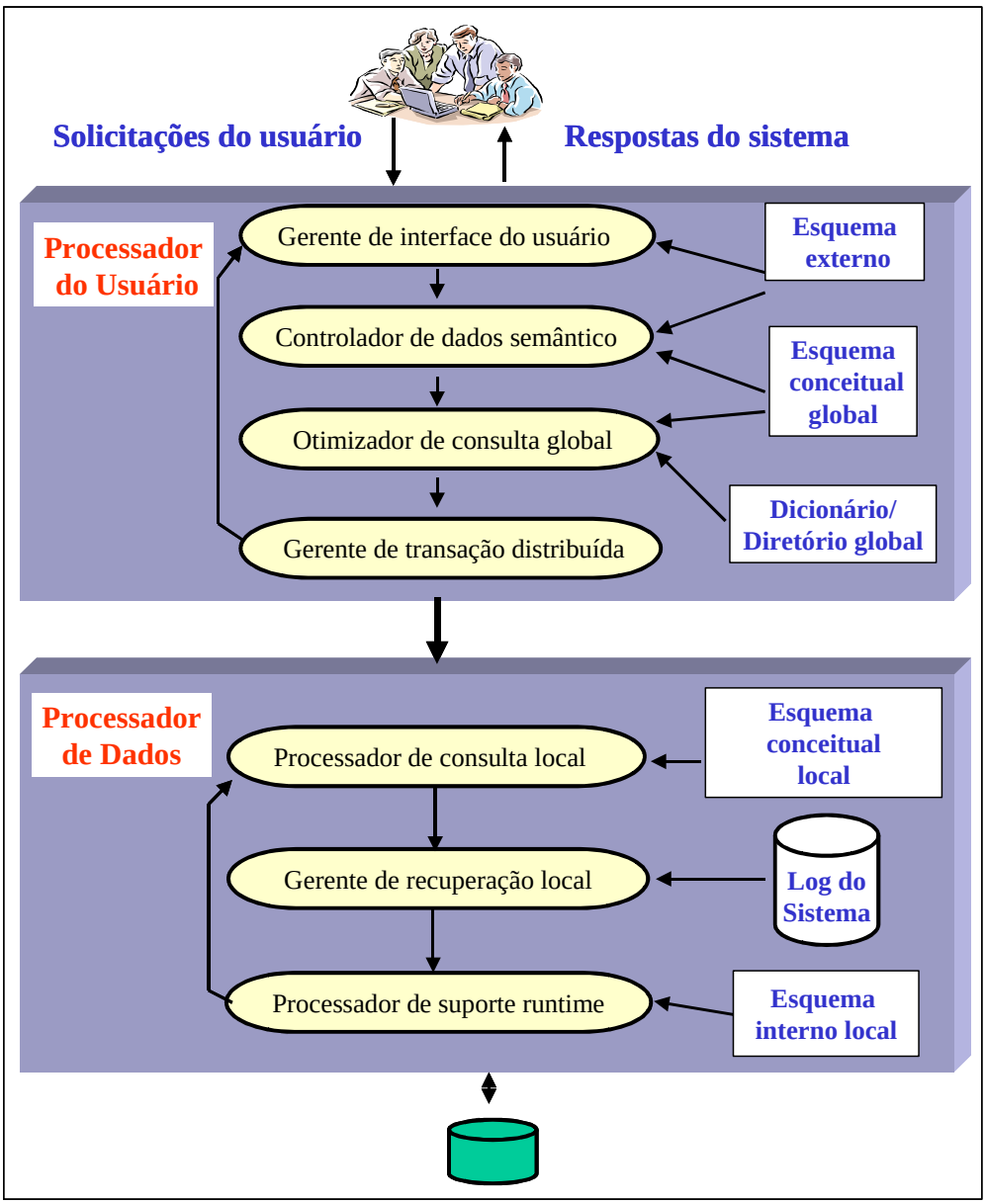


**Figura 5:** Componentes da arquitetura de um SGBDD peer-to-peer extraída de [19]

Como mostra a figura 5, a arquitetura está dividida em dois componentes principais, o *Processador do Usuário* que trata da interação com o usuário e o *Processador de Dados* que lida com o armazenamento dos dados. A seguir apresentamos a descrição de seus componentes.

- ***Componentes do Processador do Usuário***

  - ***Gerente da interface do usuário***: responsável por toda interação com o usuário, interpretação dos comandos submetidos ao SGBD, bem como pela formatação dos resultados que serão fornecidos aos usuários.

  - ***Controlador de dados semânticos***: responsável pela autorização ou não pelo processamento da consulta solicitada pelo usuário. Utiliza as restrições de integridade e autorizações definidas no esquema conceitual global.

- ***Otimizador de consultas globais***: responsável pela definição da melhor estratégia para execução das consultas (*entre elas junções distribuídas*) e conversão das consultas globais em consultas locais, utilizando os esquemas conceitual locais e global.

- ***Gerente de transação distribuída***: responsável pela coordenação da execução distribuída da solicitação do usuário e pela comunicação com outros sites.

- ***Componentes do Processador de Dados***

  - ***Otimizador de consulta local***: responsável pela escolha da melhor estratégia de acesso aos dados do banco de dados.

  - ***Gerente de recuperação local***: responsável pela consistência do banco de dados, independente de qualquer tipo de falha.

  - ***Processador de suporte runtime***: responsável pelo acesso físico ao banco de dados, em função das instruções geradas pelo otimizador de consultas e pela gerência dos *buffers* do banco de dados. É a interface para o sistema operacional.

É importante observar que essa arquitetura, diferentemente da arquitetura *cliente/servidor*, não implica em uma obrigatoriedade de implementação em máquinas diferentes. Assim, sites com a funcionalidade de atualização do banco de dados podem ter as duas camadas, enquanto que sites somente de consulta só precisam da camada *processador do usuário*.

#### 2.2.1.3. Arquitetura com vários SGBDs (multi databases)

As diferenças no nível de autonomia entre os *vários SGBDs distribuídos* e os *SGBDs distribuídos* também se refletem em suas arquiteturas. Sua principal diferença está na definição do esquema conceitual global. No caso de *SGBDs distribuídos logicamente integrados* o esquema conceitual global define a visão conceitual de *todo* o banco de dados. Já no caso de vários SGBDs distribuídos, apenas *alguns* bancos de dados locais podem estar compartilhados. A figura 6 apresenta a arquitetura para vários SGBDs distribuídos.

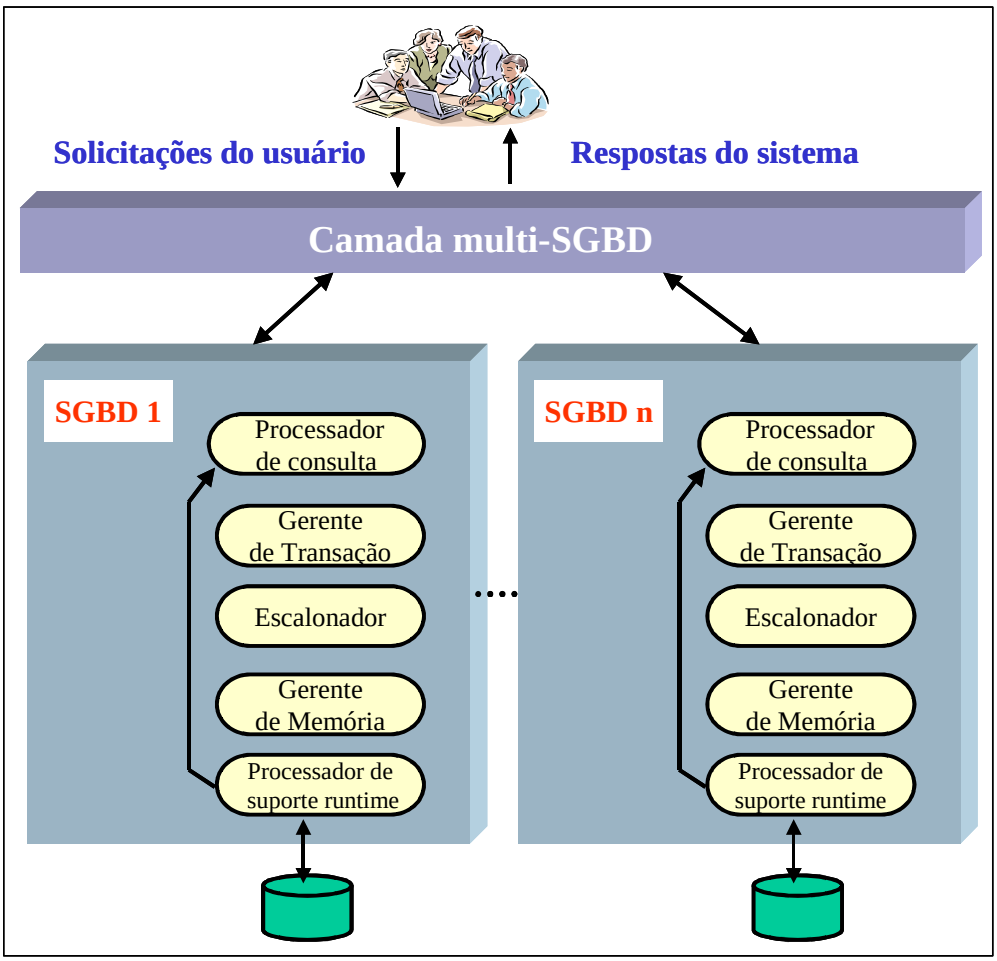


**Figura 6:** Componentes da arquitetura de um SGBDD com vários SGBDs extraída de [19]

Esta arquitetura é significantemente diferente das arquiteturas anteriores, pois nela existem SGBDs completos, cada um dos quais responsável pela gerência de um banco de dados distinto. Esta arquitetura fornece uma camada de software denominada de *multi-SGBD* que funciona coordenando a integração entre os diferentes SGBDs, fornecendo todos os recursos necessários para aceso aos diversos bancos de dados.

## *2.3. SGBD Ativos - SGBDA*

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 SGBDA 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. SGBDA 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 SGBDA 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 redundante, 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 SGBDA são centrados no conceito de regras, que especificam o comportamento ativo desejado. De forma geral, as regras dos SGBDA 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 [17,20].

### 2.3.1. Arquitetura dos SGBDA

A arquitetura de um SGBDA determina suas funcionalidades e que componentes são necessários para realizá-las. Desde que um SGBDA, por definição, deve prover capacidade ativa adicionais a todas as funcionalidades de um SGBD convencional, o mesmo pode ser visto como uma extensão de um SGBD convencional. Existem várias propriedades que sustentam e fundamentam um SGBD e a estratégia da arquitetura usada para implementação da extensão ativa terá um impacto na funcionalidade e na performance do SGBDA. As principais dimensões que necessitam ser consideradas são [20]:

- O grau de integração entre base do SGBD e as capacidades (funcionalidades) ativas;
- A arquitetura do sistema que sustenta o SGBD;
- O modelo de dados do SGBD a linguagem de programação usada para extensão ativa.

Embora existam várias arquiteturas para SGBDA, somente as mais referenciadas na literatura serão citadas a seguir.

- **Arquitetura em Camadas (*Layered*) -** Nessa arquitetura, todos os componentes ativos residem em um módulo construído no topo e independente do SGBD, se este for um SGBD passivo convencional. Utiliza um *run-time*, ou seja, os efeitos da utilização da regra são inseridos no momento da execução da mesma.

- **Arquitetura Integrada (*Built-In or Integrated arqchitecture)* -** Todos os componentes do banco de dados ativo se tornam parte do próprio SGBD. Isto decorre da modificação de um SGBD passivo existente, adicionando funcionalidades ativas ou construindo um SGBD por inteiro. Também utiliza *run-time*.

- **Arquitetura Compilada -** Nessa arquitetura não é necessário uma atividade "*run-time*". Ao contrário, na hora em que procedimentos de uma aplicação ou operações de banco de dados são compilados, eles são modificados para incluir os efeitos de regras de banco de dados ativos.

A figura 7 apresenta uma arquitetura de um SGBDA.

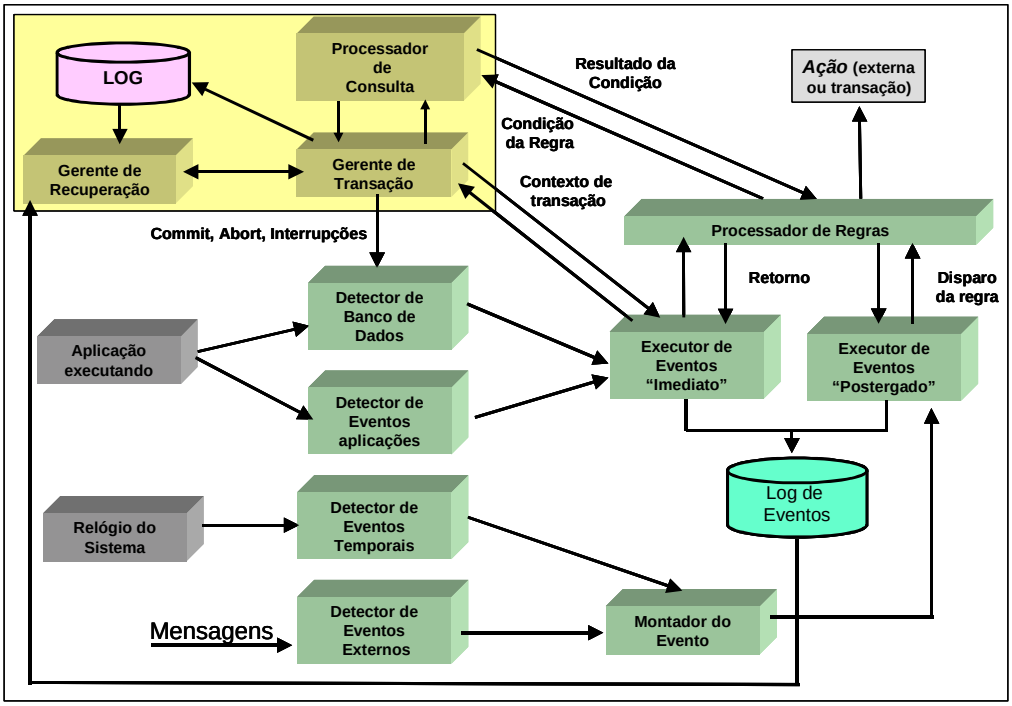


**Figura 7:** Componentes da arquitetura de um SGBDA de [20]

## 3. Sistema de Gerencia de Banco de Dados baseados em agentes de software

Conforme vimos no item anterior, diversas arquiteturas podem ser utilizadas para o desenvolvimento de um SGBD, seja ele convencional, distribuído ou ativo. Entretanto, nos SGBDs atuais ainda não é utilizada uma arquitetura que integre totalmente a tecnologia de agentes de software na construção de todos os componentes do SGBD. Em [38,42] são propostas três diferentes abordagens de arquiteturas para a construção de um SGBD baseado em agentes, as quais são:

- Em camadas (layered) – esta abordagem permite a implementação de um sistema de agentes sobre o SGBD convencional com pequena ou nenhuma modificação do mesmo. Nesta abordagem, o agente funciona como intermediário entre o usuário/aplicação e o SGBD. O Agente recebe, manipula e traduz as solicitações para o SGBD.

- Integrada (Integrated) – esta abordagem é baseada na substituição dos componentes tradicionais de um SGBD por subsistemas de agentes. O conjunto subsistemas de agentes deve apresentar todas as funcionalidades de um SGBD, substituindo todos os componentes de um SGBD tradicional.

- Embutida (Built-in) – esta abordagem prevê o uso de um SGBD existente para fazer uso de sua implementação. Nesta abordagem, os agentes são embutidos dentro do SGBD e utilizam os serviços dos componentes existentes para realizarem suas operações, ao mesmo tempo em que incorporam novas funcionalidades ao SGBD. Os

Agentes agem sobre os componentes podendo interferir em seu comportamento e obter informações sobre seu funcionamento.

Um estudo completo sobre essa três abordagens pode ser encontrado em [42].

# 4. Sistemas Móveis

Mobilidade tornou-se um novo paradigma em sistemas distribuídos, causando efetivamente mudanças significativas na forma de desenvolver sistemas de software. Entre suas principais características se destacam a mobilidade de seus hosts (sites), 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, bem como a utilização de *agentes* proporcionará mais autonomia em sua ação no sentido de melhor atingir seus objetivos.

## *4.1. 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 à 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 à 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].

### 4.1.1. 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]. A figura 8 a seguir apresenta uma arquitetura genérica para um ambiente de computação móvel.

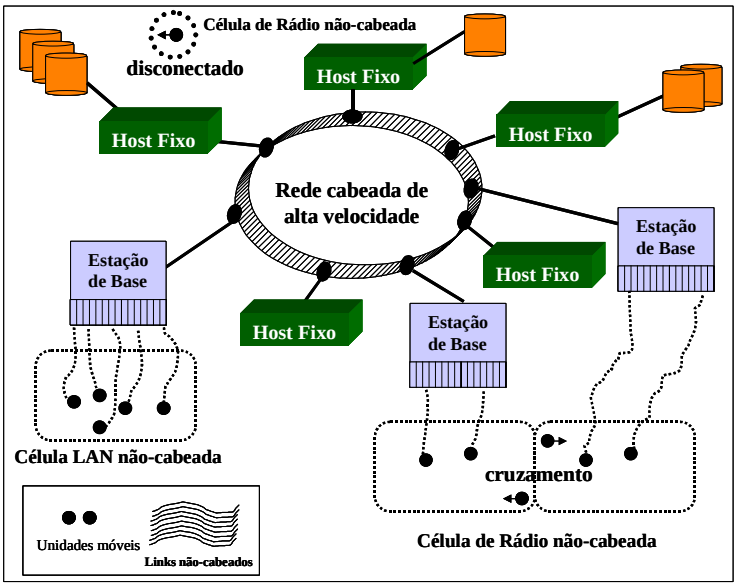


**Figura 8:** Uma arquitetura genérica para uma plataforma móvel [18].

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 autoconfigurar; (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 encriptografia de dados.

### 4.1.2. 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 à falhas: (5) projeto de bancos de dados móveis ([18], [04], [05] e [06]).

# 5. Framework

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 [16].

## *5.1. 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 [16].

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* [16].

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.

## *5.2. 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].

# 6. Agentes

Um agente é um sistema de computação situado em algum ambiente computacional, sistema este que é capaz de executar ações autônomas neste ambiente, visando atingir os objetivos do projeto para o qual foi projetado e desenvolvido. Um sistema de agentes deve poder agir sem a intervenção direta dos seres humanos ou de outros agentes, como também deve possuir todo o controle de suas próprias ações e estado interno. Um agente inteligente é um sistema computacional que é capaz de executar ações autônomas *flexíveis* a fim se atingir os objetivos do projeto para o qual foi projetado e desenvolvido. Por flexível, entendemos que o sistema deve ser *reativo, proativo* e *social* [23].

A principal característica de um agente é a sua habilidade de agir de forma autônoma. Isto implica que um agente recebe estímulos de seu ambiente, pode executar um conjunto de ações que alteram este mesmo ambiente, decidindo pelas ações que devem executadas baseadas em seus próprios objetivos [25]. Os agentes de software podem ser classificados por sua (1) *mobilidade*, (2) *capacidade de reação*, (3) *através dos diversos atributos que idealmente devem exibir*, (4), *por seus papéis, e* (6) *pelos agentes hybrid que combinam dois ou mais filosofias do agente em um único agente* [24].

## *6.1. Agentes Móveis*

Agentes móveis são agentes que são transmitidos de um ambiente computacional para outro, executam dentro de um ambiente virtual seguro no ambiente remoto para obter a informação procurada, e então, são transmitidos de volta para seu ambiente de origem. Agentes estáticos são agentes que emitem mensagens declarativas para outros agentes, freqüentemente utilizando interfaces de comunicação padrões.

A diferença entre os agentes móveis e os agentes estáticos não é, entretanto, demasiadamente grandes. Sistema que empregam agentes móveis, também empregam agentes estáticos associados com seus ambientes computacionais.

A forma de enviar os agentes móveis para um outro ambiente para recolher as informações necessárias é análoga à forma como um agente estático emite um pedido declarativo a um outro agente, o qual contem a informação desejada. Um sistema que usa agentes móveis requer um ambiente comum para execução do agente durante toda a execução do sistema ou o mínimo possível de compatibilidade para sua execução [25].

# 7. Arquitetura do Sistema

Usuários da computação móvel, inúmera vezes, necessitam recuperar (*retriever*) dados em bancos de dados fora da área de controle de sua estação de base (*base station*). Agentes, além de outras propriedades (*agency properties*), possuem a funcionalidade de migração entre ambientes [28], possibilitando uma maior agilidade na busca e recuperação dos dados para a aplicação, embora, em alguns casos, essa migração possa causar uma degradação no ambiente onde residem as fontes de dados [11].

A arquitetura proposta visa incorporar a um SGBD as funcionalidades de um sistema com comportamento ativo/reativo, disponibilizando-o para utilização em um ambiente de computação móvel. Diversos padrões de arquiteturas poderiam ser utilizados, destacando-se as arquiteturas em camadas (*layered*), embutida ou integrada (*built-in* ou *integrated*) e compilada (*compiled*) [17, 22].

Utilizar sistemas de agentes (*agents system*) na construção de um SGBD na terceira camada da arquitetura, disponibilizará diversas funcionalidades do ambiente móvel até então não

disponíveis nos SGBD convencionais, tais como controle da perda de conexão, controle sobre as transações que necessitam várias interações entre os clientes móveis e o SGBD [10], controle sobre o término de transações longas, capacidade de detecção dos eventos móveis em função de mudanças no ambiente do SGBD, atomicidade e concorrência das transações [04,05], entre outras funcionalidades.

Todas essas funcionalidades poderão estar presentes no sistema de agentes, dependendo do nível de implementação desejado. A principal contribuição dessa arquitetura, além da combinação das tecnologias de frameworks, agentes e SGBD com comportamento ativo/reativo para um ambiente de computação móvel, é passar o controle da utilização dos dados do SGBD para o próprio ambiente do SGBD, tornando-o genérico para todas as aplicações o acesso e atualização aos dados, deixando para o ambiente móvel, somente tarefas inerentes ao ambiente, tais como o controle da conexão entre o cliente móvel e o sistema, a localização espacial dos clientes etc [02, 03, 10].

A escolha da arquitetura em *camadas* trás a vantagem de poder estender SGBDs passivos em sistemas com comportamento ativo/reativo e integrá-los através de interfaces comuns a um ambiente de computação móvel com alto grau de integração de seus dados.

A escolha da arquitetura *embutida* ou *integrada* trás a vantagem de desenvolver um novo SGBD com comportamento ativo/reativo, com todos os seus componentes desenvolvidos como agentes de softwares, já incorporando todas as funcionalidades necessárias para o ambiente de computação móvel.

A arquitetura proposta neste trabalho é a arquitetura *embutida* ou *integrada*. A construção dessa arquitetura se dará através da criação de um framework de software. Para melhor implementarmos esse framework utilizaremos a tecnologia de agentes de software, explorando, principalmente, a sua capacidade de autonomia, percepção e adaptação ao ambiente, reação e mobilidade.

## *7.1. Camadas e Funcionalidades da Arquitetura*

A arquitetura proposta está dividida em três camadas: (1) aplicações móveis; (2) estações de base; e (3) SGBD para ambiente de computação móvel com comportamento ativo/reativo. A figura 9 a seguir exibe a arquitetura proposta.

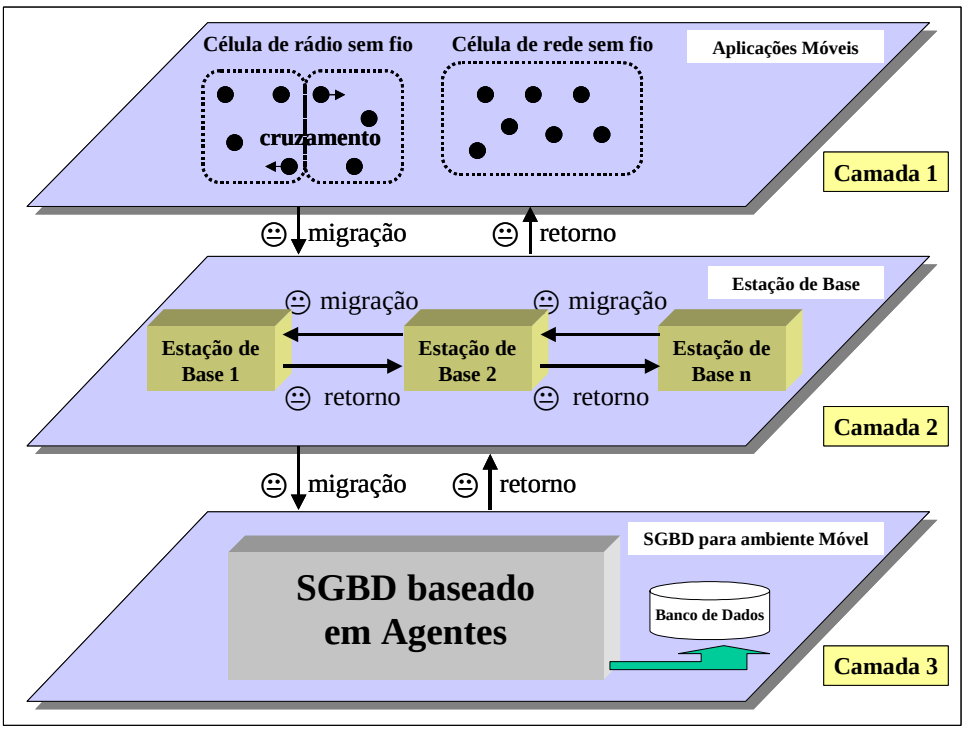


**Figura 9:** Arquitetura proposta para o SGBD baseado em agentes

### 7.1.1. Nível 1 – Aplicações Móveis (Mobile applications)

A primeira camada da arquitetura proposta contem suas aplicações desenvolvidas como agentes de software, visando garantir que a mobilidade dos agentes empregados no ambiente da computação móvel, possibilitem uma maior eficiência na gerência da localização dos usuários, identificados pelos agentes das aplicações, de modo que esses agentes possam se comunicar com os agentes da segunda camada e se re-configurarem dinamicamente, segundo a necessidade do ambiente em que estejam.

### 7.1.2. Nível 2 – Estações de Bases (Base Stations)

A segunda camada da arquitetura proposta contem suas aplicações desenvolvidas como agentes de software, visando garantir a comunicação entre as aplicações móveis e os SGBDs. Inúmeras vezes, os dados necessários para a aplicação móvel não residem na mesma rede local (*local network*) da estação de base a qual o usuário móvel está conectado, dificultando o acesso à fonte de dados e, principalmente, não garantindo que as requisições de dados das aplicações serão atendidas, em função de inúmeras falhas que podem ocorrer [02]. Na arquitetura proposta o sistema de agentes (*agents systems*) das estações de base, terá as seguintes funcionalidades:

1. Garantir a comunicação com os agentes das aplicações móveis;
2. Perceber a mudança de localização do usuário para outra estação de base;
3. Detectar uma possível ruptura na comunicação com o usuário;
4. Entregar os dados solicitados às aplicações móveis;
5. Trocar mensagens com outros agentes na estação de base e entre estações de base;

6. Se movimentar para outra estação de base;

7. Solicitar dados ao sistema de agentes do SGBD;

8. Detectar e controlar uma possível ruptura na comunicação com o ambiente do SGBD;

9. Migrar para o ambiente do SGBD;

10. Garantir a entregar dos dados as aplicações móveis, independente de qualquer tipo de falha.

Ou seja, controlar o ambiente da computação móvel, trocando mensagens como o sistema de agentes do SGBD móvel, visando garantir a qualidade do serviço do ambiente móvel.

### 7.1.3. Nível 3 – Sistemas de agentes para um SGBD

A terceira camada da arquitetura proposta propõe uma arquitetura integrada para um SGBD com comportamento ativo/reativo para um ambiente de computação móvel. Este "SGBD móvel" tem como principais características, além do seu comportamento ativo/reativo, o gerenciamento do banco de dados distribuído, tornando a distribuição dos dados transparente para os usuários da computação móvel. Para melhor desenvolvimento dessa arquitetura, propomos, também, a construção de um framework que integrará os componentes do SGBD e adicionará as funcionalidades necessárias ao ambiente da computação móvel. Os frozen e hot spots do framework proposto serão desenvolvidos como um sistema de agentes e estão descritos a seguir.

## 8. Um Framework para desenvolvimento da arquitetura proposta

Um framework para construção de um SGBD com comportamento ativo/reativo para um ambiente de computação móvel é um conjunto de software que usa as tecnologias de banco de dados convencionais, ativo e distribuído, bem como de frameworks, aplicados à *computação móvel*. A seguir, são apresentados seus componentes, classificados em *hot* e *frozen spots*.

## *8.1. Objetivo do Framework*

O objetivo do framework proposto é possibilitar a construção da terceira camada da arquitetura proposta, desenvolvendo um SGBD através de um sistema de agentes de softwares. Neste SGBD estarão incluídas as funcionalidades de ação e reação, distribuição de dados, detecção de eventos, distribuição de regras e dados entre os hosts (sites) e o controle dos requisitos necessários ao ambiente da computação móvel [02, 03 e 22]. O framework proposto está apresentado na figura 10 no item a seguir, bem como a descrição de seus frozen e hot spots.

## *8.2. Componentes do Framework*

O framework proposto, apresentado na figura 10, é composto pelos componentes *Gerente de Esquema, Gerente de Consulta, Gerente de Banco de Dados, Gerente de Armazenamento, Gerente de Distribuição, Gerente de Regras Ativas* e *Gerente de Ambiente Móvel*.

Os componentes *Gerente de Esquema, Gerente de Consulta, Gerente de Banco de Dados* e *Gerente de Ambiente Móvel* são seus frozen spots, ou seja, conservam-se inalteráveis em qualquer de suas instanciações, enquanto que os componentes *Gerente de Armazenamento, Gerente de Distribuição e Gerente de Regras Ativas* são seus hot spots, ou seja, devem ser configurados com as funcionalidades específicas desejadas para uma determinada instanciação do framework. Estes componentes estão descritos a seguir.

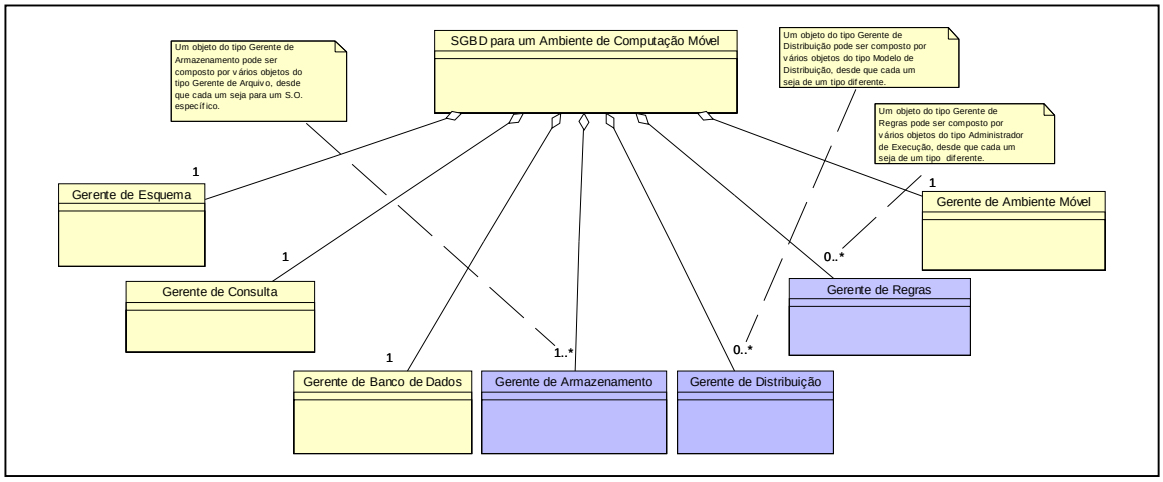


**Figura 10**: Framework para construção do SGBD da terceira camada da arquitetura

## *8.3. Frozen Spots do Framework*

A seguir é apresentada a definição dos frozen spots do framework proposto:

1. *Gerente de Esquema*: responsável pela conversão dos comandos de definição de dados num conjunto de tabelas que serão armazenadas no catálogo do sistema, gerando seus metadados. É responsável, também, pela manutenção do dicionário do SGBD e todo o acesso ao mesmo para validação ou recuperação de dados do SGBD.

2. *Gerente de Consulta*: é responsável pela transformação das consultas submetidas ao SGBD em uma série de instruções e direcioná-las ao Gerente de Banco de Dados. É responsável, também, pela conversão dos comandos da linguagem de consulta embutidas em programas de aplicações em código padrão para processamento no SGBD.

3. *Gerente de banco de Dados*: é responsável pelo processamento propriamente dito no SGBD. Recebe solicitações de consultas e programas de aplicações e examina os requisitos necessários para satisfazê-las. Verifica as autorizações, otimiza as consultas, executa as transações, determina as prioridades de execução, bem como administra a comunicação com o sistema operacional para gravação e recuperação de dados.

4. *Gerente de Ambiente Móvel*: é responsável pela implementação das funcionalidades para o ambiente móvel, bem como pela comunicação dos agentes do SGBD com os agentes móveis das estações de base, visando garantir a entrega dos resultados das consultas, bem como das atualizações.

### 8.3.1. Hot Spots do Framework

A seguir é apresentada a definição dos hot spots do framework proposto:

1. *Gerente de Armazenamento*: é responsável pela manipulação dos arquivos para armazenamento e gerencia a alocação dos espaços em disco. Trabalha em sintonia com o sistema operacional, utilizando um método de acesso apropriado.

2. *Gerente de Distribuição*: é responsável pelos controles e requisitos da distribuição dos dados pela ótica dos SGBD distribuídos, garantindo a execução correta das transações e controlando a concorrência das mesmas.

3. *Gerente de Regras Ativas*: é responsável pelas funcionalidades de ação e reação do SGBD, incluindo a definição de regras, detecção de eventos móveis, temporais, de banco de dados e aplicações, bem como pelo controle e execução das regras no ambiente móvel (distribuído).

## *8.4. Componentes do Framelet Gerente de Regras Ativas*

Para exemplificar a utilização dos agentes de softwares, desenvolveremos o framelet Gerente de Regras Ativas e alguns de seus agentes. O framelet Gerente de Regras Ativas para um ambiente de computação móvel é composto por um conjunto de componentes e está apresentado na Figura 11.

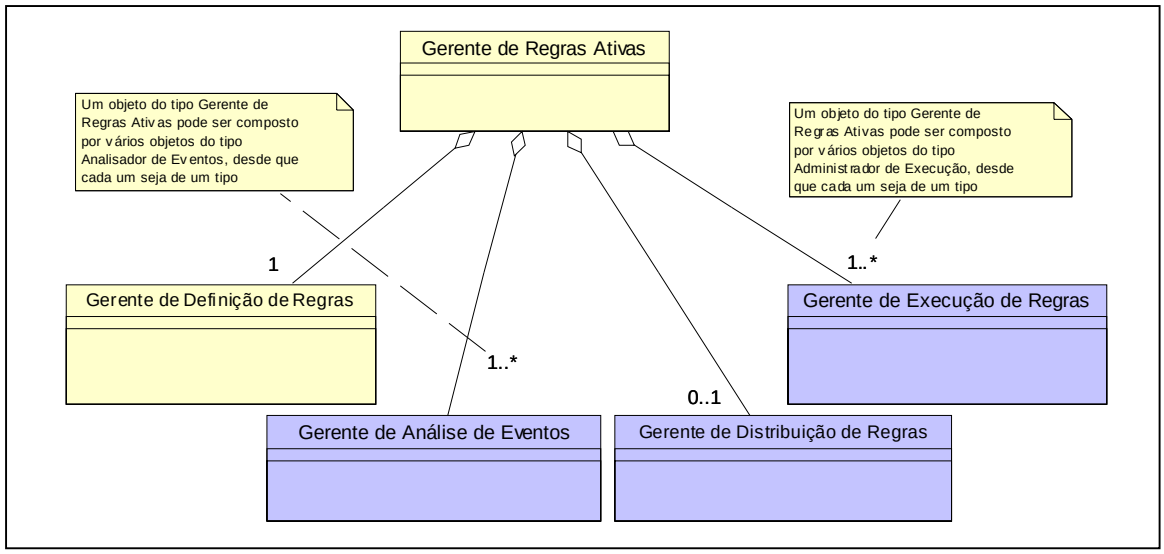


**Figura 11**: Framelet de Regras Ativas para um Ambiente de Computação Móvel.

Os componentes *Gerente de Regras Ativas* e *Gerente de Definição de Regras* correspondem aos *frozen spots* do framework, ou seja, conservam-se inalteráveis em qualquer de suas instanciações. Os componentes *Gerente de Analise de Eventos, Gerente de Distribuição de Regras e Gerente de Execução de Regras* correspondem aos *hot spots* do framework, ou seja,

devem ser configurados com as funcionalidades específicas desejadas para uma determinada instanciação do framework.

### 8.4.1. *Frozen Spot* do *Framelet*

Os *frozen spots* do *Framelet* são:

1. *Gerente de Regras Ativas*: este componente é responsável por todo o controle do processo de definição, análise, distribuição e execução das regras. É composto pelo frozen spot e *Gerente de Definição de Regras* e pelos hot spots *Gerente de Analise de Eventos, Gerente de Distribuição de Regras e Gerente de Execução de Regras*;

2. *Gerente de Definição de Regras:* este componente é responsável pela atualização das informações apropriadas no catálogo do sistema, sendo sempre chamado quando uma nova regra é definida, alterada ou excluída. É quem interage com o *gerente de esquema* para escrever as regras.

### 8.4.2. *Hot Spots* do *Framelet Gerente de Regras Ativas*

Os *hot spots* do *Framelet* são:

1. *Gerente de Analise 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. Ele também é um framework, logo possui seus próprios hot e frozen spots

2. *Gerente de Distribuição de Regras* - 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.

3. *Gerente de Execução de Regras* - este componente é responsável pela identificação da forma como são geradas 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 também é um framework, logo possui seus próprios hot e frozen spots.

## *8.5. Framelet Gerente de Análise de Eventos*

Os componentes deste *framelet* estão especificados na figura 12 e definidos a seguir.

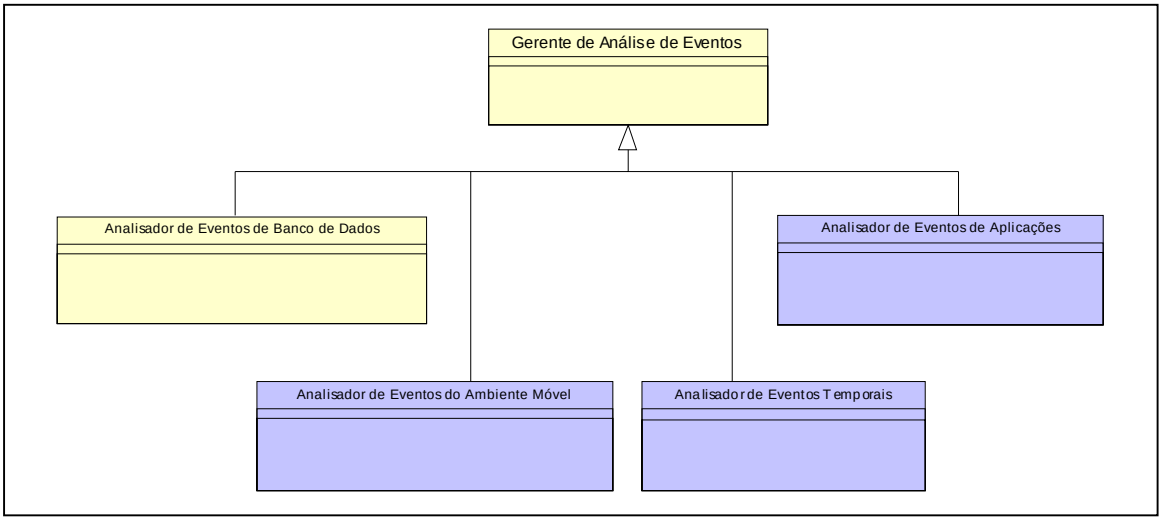


**Figura 12**: Componentes de *Framelet* Gerente de Análise de Eventos

- **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 do Ambiente Móvel**: detecta e avalia eventos oriundos do ambiente de computação 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 *Gerente de Analise de Eventos*, conforme apresentado na figura 12.

# 9. Arquitetura dos *Agentes de Software* para o Framework Gerente de Analise de Eventos

A seguir é apresentada as arquiteturas dos agentes de software para os componentes Gerente de Análise de Eventos e Analisador de Eventos de Banco de Dados.

## 9.1. Arquitetura do *agente de software* para o componente Gerente de Análise de Eventos

O projeto da arquitetura do *agente de software* do componente Gerente de Análise de Eventos é uma abordagem top-down modular, com os módulos *Sensor, Estímulo, Conhecimento* e *Tradução/Comunicação* conforme descrito na figura 13. O módulo *Sensor* é específico de um ambiente e capta mudanças no mesmo. O módulo *Estímulo* modela as intenções estratégicas do agente, segundo os estímulos que o mesmo possa receber.

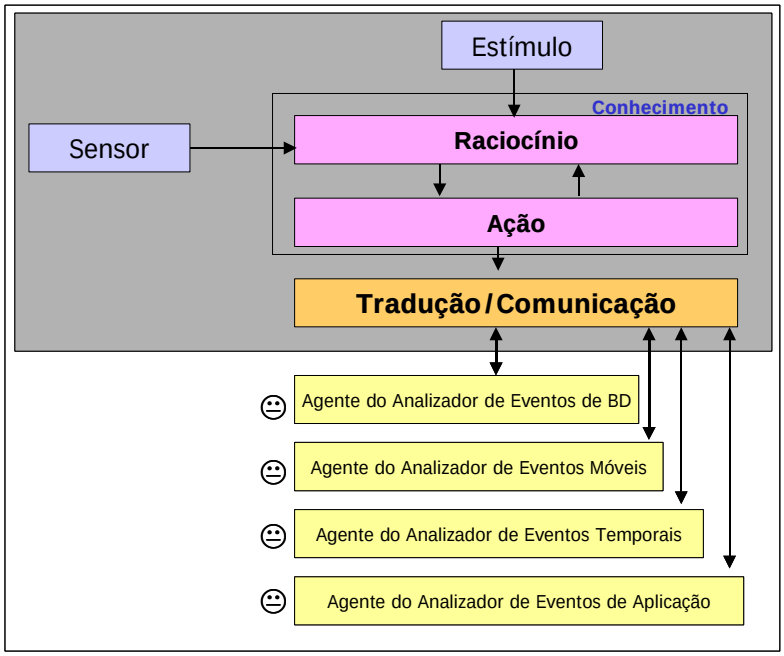


**Figura 13**: Arquitetura do Sistema de Agentes do Componente Gerente de Análise de EventosO modulo *Conhecimento* (*experiência*) avalia a situação corrente e seleciona/controla e monitora as ações que o agente deve executar em uma particular situação. Utilizando-se o módulo *Tradução/Comunicação*, projetado para um ambiente específico, aciona um outro agente, em função do tipo de evento percebido pelo módulo *Sensor* no ambiente ou invocado por um evento específico através do módulo *Estímulo*.

## 9.2. Arquitetura do *agente de software* para o componente Analisador de Eventos de Banco de Dados

O projeto da arquitetura do agente de software Analisador de Eventos de Banco de Dados é uma abordagem top-down modular, com os módulos Tradução/Comunicação, Conhecimento, Mobilidade e ativador, conforme descrito na figura 14 a seguir.

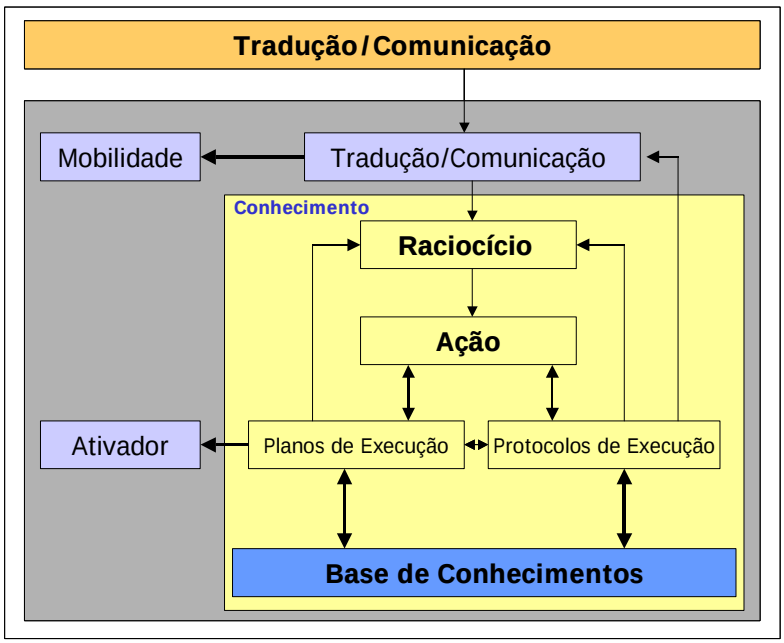


**Figura 14**: Arquitetura do Sistema de Agentes do Componente Analisador de Eventos de Banco de Dados

O módulo *Tradução/Comunicação* é específico de um ambiente e recebe as mensagens de ativação do agente Gerente de Análise de Eventos.

O módulo *Mobilidade* se movimenta entre banco de dados, mudando de ambiente pela rede, quando a requisição de dados exige dados de um outro SGBD.

O módulo *Ativador* ativa todas as ações que devem ser executadas no ambiente do agente ou fora do mesmo.

O módulo *Conhecimento* avalia a situação corrente e seleciona e controla as ações do agente em cada situação específica, seus planos de execução e protocolos necessários. Este módulo é concebido e executa como um sistema baseado em conhecimento. Pode adaptar, rever, modificar ou abandonar metas existentes, bem como decidir por novas reações ou metas em função de mudanças percebidas no ambiente. Consiste de uma *Base de Conhecimento* (*Knowledge Base*) e dos módulos Raciocínio, Ação, Planos de Execução e protocolos de Execução.

O módulo *Raciocínio* analisa a chamada do agente, modela o agente e decide que ações ou reações devem ser executadas.

O módulo *Ação* seleciona a melhor ação a ser executada, examinando a base de conhecimento.

O módulo *Planos de Execução* seleciona e administra os planos de execução que melhor cumprirão as metas selecionadas pelo agente, identificando se esses planos deverão ser executados no próprio agente, enviados ao módulo *Ativador* ou necessitam de protocolos específicos.

O módulo *Protocolos de Execução* representa alguns padrões de comunicação e diálogos com outros agentes e ambientes. Possui características de protocolos e lógicas para construção de diálogos.

## 10. Comentários Finais

A utilização da técnica de framework e agentes de software na construção de SGBDs se apresenta como uma alternativa muito eficiente, principalmente em função do reaproveitamento do código das funcionalidades que sejam comuns e das características de agência que o software desenvolvido passa a ter.

O objetivo deste trabalho foi propor e detalhar uma arquitetura para construção de um Sistema de Gerência de Banco de Dados para um ambiente de Computação Móvel, combinando as tecnologias de framework, agentes e SGBD, incorporando a funcionalidade de ação e reação ao SGBD. Essa combinação de tecnologias deu ao projeto maior facilidades de construção de seu software através do framework, possibilidade de configurações em função do ambiente operacional, bem como incorporou ao banco de dados uma série de facilidades para controle, administração, distribuição e mobilidade através de sistemas agentes, conforme descrito neste trabalho.

A principal contribuição deste trabalho esta baseada na idéia de que as aplicações em um ambiente de computação móvel devem executar apenas atividades específicas desse ambiente, deixando para os Hosts e seus SGBD todas as atividades de controle, segurança, distribuição e armazenamento e recuperação de dados, comuns a todas as aplicações. Além disso, incorporar ao ambiente móvel, um banco de dados com funcionalidades para o ambiente de computação móvel com características e funcionalidades hoje dispersas em muitas aplicações e ambiente, aumentando o potencial para utilização de dados nesse ambiente.

## 11. 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] Thomas Connolly, Carolyn Begg and Anne Strachan, “Database System – A pratical approach to Design, Implementation and Management”; Addison-Wesley, 1996.

[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 254-263.

[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 SGBD”; DEXA, 2001, Berlin Heidelberg 2001, pp 595-600

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

[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] Nicholas R. Jennings and Michael J. Wooldridge; “Agent Technology – Foundations, Applications, and Markets – Applications of Intelligent Agents”; part 1, pages 3-28, Springer, 1998.

[24] H.S Nwana and D.T. Ndumu; “Agent Technology – Foundations, Applications, and Markets – A Brief Introduction to Software Agent Technology”; part 1, pages 29-47, Springer, 1998.

[25] P. Kearney; “Agent Technology – Foundations, Applications, and Markets – Personal Agents: A Walk on The Client Side”; part 1, pages 125-136, Springer, 1998.

[26] M. Georgeff and A. Rao; “Agent Technology – Foundations, Applications, and Markets – Rational Software Agents – From Theory to Practice”; part 3, pages 139-160, Springer, 1998.

[27] B. Burmeister, S. Bussuman, A. Haddadi, and K. Sundermeyer; “Agent Technology – Foundations, Applications, and Markets – Agent-Orient Techniques for Traffic and manufacturing Applications: Progress Report”; part 3, pages 161-174, Springer, 1998.

[28] J. Bailey; M Georgeff, D. Kemp and D. Kinny; “Active Databases and Agent Systems - A comparison”, Proceedings of the Second International Workshop on Rules in Database Systems (RIDS’95), . LNCS 985, pp.342-356, Springer; 1995.

[29] J. Akker and A. Siebes, “Enriching Active Databases with Agent Technology”; Proceedings of the First International Workshop on Cooperative Information Agents (CIA-97), LNCS 1202, pp.116-125, Springer; 1997.

[30] M. Magnanelli and M. Norrie, “Database for Agents and Agents for Databases”, AOIS; 2000.

[31] S. Côrtes e C.J.P de Lucena “Um Framework para construção de Sistemas de Banco de dados Móvel com Regras Ativas”. Monografias em Ciência da Computação – 35/01, PUC-Rio, Departamento de Informática, Brasil, 2001.

[32] J-J. Yoo, Y-H. Suh, D-I. Lee, S-W. Jung, C-S. Jang and J-B. Kim, "Casting Mobile Agents to Workflow Systems: On Performance and Scalability Issues", Procs DEXA Conference, 2001, pp 254-263.

[33] R.L.C. Costa e S. Lifschitz, “Uma Proposta de Arquitetura de SGBD baseado em Agentes para Auto-Sintonia em Índices", Monografia em Ciência da Computação, MCC 28/01, Departamento de Informática - PUC-Rio, Brasil, 2001.

[34] Fayad, M., Schmidt, D.C. and Johnson, R.: “Implementing Applications Frameworks: Object Oriented Frameworks”, Wiley & Sons, 1999.

[35] Kendall, E. Krishna, P. Murali, P. Chirag and Suresh, C.B. “Implementing Application Frameworks.” Wiley. 1999.

[36] Kendall, E.A.; Krishna, P.V.M.; Pathak, C.V. and Suresh, C.B., “An Application Framework for Intelligent and Mobile Agent Systems”, chapter in 8.

[37] L. Li and S. Chakravarthy, "An Agent-Based Approach to Extending the Native Active Capability of Relational Database Systems", Procs (ICDE) IEEE Intl Conf Data Engineering, 1999, pp. 384--291.

[38] S. Lifschitz e J.A.F. Macêdo “Agent-based Databases and Parallel Join Load Balancing” , Procs XXVII Latin-American Conference on Informatics (CLEI), Mérida, Venezuela, Setembro 2001, 15 pp Anais em CD-ROM (Anais de Resumos pp 47)

[39] Nagi, K. and Lockemann, P.: “An Implementation Model for Agents with Layered Architecture in a Transactional Database Environment”. Proceedings of the First International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS) 1999.

[40] Türker, C. Saake, G. and Conrad, S.: “Modeling Database Federations in Terms of Evolving Agents”. Proceedings of the 10th International Symposium on Methodologies for Intelligent Systems (ISMIS), 1997, pp.197-208.

[41] Wooldridge, M., “Intelligent Agents, chapter in Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence”, ed. Gerhard Weiss, The MIT Press, 1999.

[42] José Antônio Fernandes de Macêdo; “Um Estudo de SGBDs Baseados em Agentes”. Dissertação de Mestrado, Departamento de Informática, PUC-Rio, Brasil, 2000.