Ulisses Rezende Teixeira

REMODELAGEM DO BANCO DE DADOS DE UM SISTEMA EM TEMPO REAL COM GRANDE VOLUME DE INFORMAÇÕES: ESTUDO DE CASO - SMARTMINE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel.

Orientador
Prof. Guilherme Bastos Alvarenga

Co-Orientadora
Profa. Olinda Nogueira Paes Cardoso

Lavras
Minas Gerais - Brasil
2005

Ulisses Rezende Teixeira

REMODELAGEM DO BANCO DE DADOS DE UM SISTEMA EM TEMPO REAL COM GRANDE VOLUME DE INFORMAÇÕES: ESTUDO DE CASO - SMARTMINE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel.

Aprovada em *20 de Janeiro de 2005*

Luiz Thomaz do Nascimento

Prof. Guilherme Bastos Alvarenga
(Orientador)

Profa. Olinda Nogueira Paes Cardoso
(Co-Orientadora)

Lavras
Minas Gerais - Brasil

Dedico este trabalho a meu pai que infelizmente não está mais entre nós, mas que, com certeza, onde quer que ele esteja, estará muito orgulhoso por mais esta conquista.

Agradecimentos

Agradeço em primeiro lugar a minha mãe e minha irmã por estarem sempre à meu lado e terem sido os pilares de mais esta conquista. A toda a minha família pelo apoio e força. Aos meus amigos, responsáveis por inesquecíveis momentos de descontração e apoio nas horas difíceis. A todos os irmãos de Ordem do Capítulo DeMolay de São João del Rei. Aos professores Ronald Zanetti (DEN) e Antonio Maria Resende (DCC) pelo confiança depositada ao longo destes 2 anos e meio de projeto MipFor e ao professor Magno de Souza (DAE) pelo projeto Coopeufla. Ao professor Guilherme Bastos e a todos os amigos de Devex, especialmente ao Modulo Manager pela acolhida. E finalmente a todos os colegas de classe pela convivência nos 4 anos de graduação que estão se encerrando e que com certeza irão deixar saudades.

Resumo

REMODELAGEM DO BANCO DE DADOS DE UM SISTEMA EM TEMPO REAL COM GRANDE VOLUME DE INFORMAÇÕES: ESTUDO DE CASO - SMARTMINE [1]

O SmartMine como um sistema em tempo real, gera um grande volume de informações que cresce com o tempo. O avanço da tecnologia e a conseqüente redução dos custos dos dispositivos de armazenamento de dados eliminaram o problema do armazenamento. No entanto, dado o tamanho de sua coleção de dados e a crescente busca por desempenho, o simples armazenamento de seus dados não necessariamente indica facilidade de acesso. Tratar, avaliar e selecionar o conteúdo relevante desse grande volume de informações continua sendo uma tarefa extremamente difícil. A capacidade de processamento dos computadores atuais, apesar de ter evoluído juntamente com os dispositivos de armazenamento, não é suficiente para garantir um desempenho aceitável na manuseabilidade destes dados. É necessário dar a eles um tratamento adequado, criando condições favoráveis à sua utilização. Algumas soluções foram propostas e implementadas tais como divisão de tabelas muito grandes em outras menores, inclusão de processos de sumarização de dados entre outros, mas o problema ainda não está solucionado. Atualmente, têm surgido novas tecnologias de banco de dados, novos conceitos e novos paradigmas. Este trabalho busca apresentar soluções no sentido de criar um banco de dados prático, consistente e eficiente, testando as novas tecnologias existentes no mercado e tecnologias já consolidadas apontando pontos positivos e negativos na sua adoção. Além disso, busca avaliar impactos de uma remodelagem do banco de dados de um sistema já implantado.

Palavras - Chave: Banco de dados, remodelagem, tempo real.

[1] Orientador: Prof. Guilherme Bastos Alvarenga Co-Orientadora: Profa. Olinda Nogueira Paes Cardoso

Abstract

REMODEL OF THE DATABASE OF A SYSTEM IN REAL TIME WITH GREAT VOLUME OF INFORMATIONS: CASE STUDY - SMARTMINE[2]

The SmartMine as a system in real time, generates a great volume of information that grows with the time. The advance of the technology and the consequent reduction of the costs of the devices of storage of data had eliminated the problem of the storage. However, given to the size of its collection of data and the increasing search for performance, the simple storage of its data not necessarily indicates access easiness. To treat, to evaluate and to select the excellent content of this great volume of information extremely continue being a difficult task. The capacity of processing of the current computers, although to have together evolved with the storage devices, it is not enough to guarantee an acceptable performance in the handly of these data. An adjusted treatment is necessary to give they, creating conditions favorable to its use. Some solutions had been implemented and proposals, but the problem was still not decided. Actually, new technologies of data base, new concepts and new paradigms have appeared. This work searchs to present solutions in the direction to create practical, consistent and efficient a data base, testing the new existing technologies in the market and technologies already consolidated pointing positive and negative points in its adoption. Moreover, it searchs to evaluate impacts of a remodel of the data base of an implanted system already.

Key Words: Database, remodel, real time.

[2]Orientador: Prof. Guilherme Bastos Alvarenga Co-Orientadora: Profa. Olinda Nogueira Paes Cardoso

Sumário

Lista de Figuras

Capítulo 1

Introdução

O sistema SmartMine é um sistema de gerenciamento de mina em tempo real que possibilita o aperfeiçoamento da utilização dos recursos minerais, humanos e equipamentos. Foi projetado enfatizando os temas mais importantes com informações confiáveis e bem apresentadas, em oposição a simples coleta de dados. Isso permite o compartilhamento de informações únicas, em tempo real, entre todos os níveis e departamentos da empresa, permitindo a rápida tomada de decisão de forma confiável, precisa e em tempo hábil. As ferramentas de informações do SmartMine determinam índices de utilização e disponibilidade dos equipamentos e as produções e produtividades da lavra de mina.

A primeira implementação de seu banco de dados foi construída sobre a tecnologia de banco de dados relacional. Criou-se uma tabela única para guardar todos os registros de estados dos equipamentos de transporte e carga. Após algum tempo, verificou-se que esta implementação não permitia a realização de consultas em tempo hábil. A solução encontrada foi duplicar as tabelas em tabelas auxiliares, continuando a usar a tabela única para armazenar um log de operações do sistema. A entrada dos dados era feita nas tabelas auxiliares e através de gatilhos (*triggers*) sumarizava-se na tabela de *log*.

A princípio, esta solução se mostrou satisfatória, mas com o passar do tempo e o acúmulo dos dados, surgiram problemas de consistência de dados entre a tabela de *log* e as tabelas auxiliares devido a condições não previstas no plano inicial além de problemas de manutenção, devido à forma como foi implementada. Por esses problemas já encontrados a revisão do projeto de banco de dados é essencial para o bom funcionamento do sistema.

1.1 Objetivos

Este trabalho, desenvolvido em parceria com a Devex Tecnologia e Sistemas Ltda (empresa de desenvolvimento de *software* e *hardware*, sediada em Belo Horizonte

- MG), analisa a situação atual, aponta gargalos e propõe algumas mudanças que poderiam melhorar ou até mesmo extinguir os problemas encontrados, ou ainda, se for o caso, a remodelagem do banco de dados como um todo.

A idéia inicial é manter a estrutura do banco de dados, minimizando assim o trabalho de adequar o sistema. Mas caso não seja possível alcançar estes objetivos desta maneira, mudanças mais profundas serão realizadas afim de alcançar os objetivos propostos aqui, que são solucionar os problemas ligados ao banco de dados que o sistema está apresentando.

1.2 Escopo do trabalho

O Capítulo 2 deste trabalho apresenta uma contextualização deste trabalho dentro das normas de modelagem e criação de banco de dados, apresentando formas de criação e modelos pré-definidos. Além disso, são descritas algumas das tecnologias candidatas a serem implementadas no processo de remodelagem.

O Capítulo 3 apresenta a metodologia utilizada do trabalho. A forma como ele foi desenvolvido, como era o banco de dados antes e quais as propostas definidas durante a etapa de análise e busca de soluções.

O Capítulo 4 apresenta os resultados e discussões acerca do trabalho realizado. É feita uma breve descrição do sistema Smartmine, comentando cada um dos seus módulos e as suas ligações. Descreve também a solução final apresentada e como foi implementada. No final, uma pequena análise do funcionamento do sistema após a conclusão do trabalho, apontando quais objetivos foram obtidos e propondo futuras alterações para problemas que não foram abordados neste trabalho.

No Capítulo 5, é feita uma breve conclusão.

Capítulo 2

Referencial Teórico

2.1 Contextualização

A evolução de demandas de uso de tecnologia de informação, resultantes de diversos processos de negócio, tem renovado à preocupação das organizações quanto à gestão do seu acervo de informações.

Segundo [LIMA - 2001], a frase que um dia foi definida como máxima da Engenharia de Informação, de que "os dados constituem-se em um dos principais ativos de qualquer instituição", ganhou novos contornos com os requisitos de negócio e tecnológicos das soluções atuais da era do conhecimento, ou era da informação.

A garantia de aspectos como alta disponibilidade, escalabilidade, desempenho, segurança, armazenamento de conteúdo multimídia, suporte a transações analíticas entre outras faz parte do elenco de requisitos a ser atendido. Além disso, os antigos problemas como estruturação e armazenamento, segurança, redundância, integridade, consistência de dados e tempo de resposta, que sempre acompanharam o tratamento de informações corporativas, continuam existindo e ganharam novos contornos [LIMA - 2001].

Um requisito fundamental do ambiente de banco de dados é o atendimento dos requisitos de desempenho necessários para satisfazer os usuários. No projeto do banco de dados, um dos objetivos primordiais é que o banco de dados ofereça "o melhor desempenho para as aplicações".

A monitoração do ambiente quanto ao atendimento destes requisitos de desempenho leva a ações que vão desde análises de gargalos de unidades de entrada/saída (IO), até a avaliação do uso das linguagens de manipulação dos dados (*Data Manipulation Languages* - DML) que podem levar a acessos não otimizados na base de dados.

[KORTH - 1995] define três níveis de ajustes de um sistema de banco de dados:

- O nível mais baixo é o de hardware e as opções de ajuste de sistemas nesse nível incluem: adicionar discos ou usar um sistema RAID se o processo de *Input* e *Output* de disco é um gargalo, adicionar mais memória se o tamanho de buffer de disco é um gargalo ou passar para um processador mais rápido se a utilização da CPU é um gargalo.

- O segundo nível consiste em parâmetros do sistema de banco de dados, como tamanho de buffer de controle de checkpoints (pontos de controle). O conjunto exato de parâmetro do sistema de banco de dados que podem ser ajustados depende do sistema específico de banco de dados, mas os sistemas gerenciadores de banco de dados bem projetados realizam tantos ajustes quantos forem possíveis, automaticamente, liberando o usuário ou administrador dessa carga.

- O terceiro nível é o projeto de mais alto nível, englobando esquemas e transações. Você pode ajustar o projeto do esquema, a criação de índices e a execução de transações de modo a melhorar o desempenho. O ajuste nesse nível é relativamente independente do sistema. Os três níveis de ajuste interagem entre si. Para um projeto de grande porte, deve-se considerar tal combinação quando é feito o ajuste do sistema. Por exemplo, ajustar um nível mais alto pode resultar em uma alteração no gargalo de hardware, como do sistema de discos para CPU, ou vice-versa.

Além desses três níveis, existe ainda o nível da própria aplicação. Um projeto de sistema bem feito pode resultar em excelentes resultados no final. Este nível se aplica somente em casos onde o sistema está ainda em fase de construção [KORTH - 1995].

No estudo de caso abordado por este trabalho, a aplicação já está concluída, portanto é inviável fazer qualquer tipo de alteração em seu projeto. Além disso, não se tem controle direto sobre o hardware que irá executar o sistema. Portanto, iremos nos concentrar principalmente nos níveis dois e três, tentando ajustar ou até mesmo remodelar o banco de dados para alcançar o seu melhor desempenho e resolver os problemas citados anteriormente.

2.2 Projeto de banco de dados

Esta seção é toda baseada na obra de [BATINI - 1992].

A falta de abordagens adequadas para o projeto de um banco de dados pode incorrer em resultados indesejáveis, como ineficiência em atender a demanda de aplicações e problemas com a manutenção do banco de dados. Geralmente a causa disso é a falta de clareza em entender a natureza exata dos dados em um nível conceitual (abstrato).

O projeto de banco de dados pode ser encarado a partir de duas perspectivas: *Top-down* e *bottom-up*.

2.2.1 Projeto com perspectiva Top-Down

Enfatiza os requisitos da aplicação obtidos com o usuário a partir da compreensão dos dados operacionais relevantes para a aplicação. É um processo mais usual e é aplicado com mais freqüência em casos onde não existe sistema informatizado ou banco de dados anterior.

Parte de um alto nível de abstração para o baixo nível. Pode ser decomposto em quatro etapas: levantamento de requisitos, projeto conceitual, projeto lógico e projeto físico.

A **análise de requisitos** coleta informações sobre os dados, suas restrições e seus relacionamentos. O resultado é um documento com a especificação formal dos requisitos.

O **projeto conceitual** usa como base à especificação dos requisitos produzindo como resultado o esquema conceitual do banco de dados. Um **esquema conceitual** é uma descrição em alto nível da estrutura do banco de dados, independente do Sistema de Gerenciamento de Banco de Dados (SGBD) [1] adotado para implementá-lo. Um **modelo conceitual**, por exemplo, o modelo Entidade-Relacionamento é usado para descrever os esquemas conceituais.

O propósito do projeto conceitual é descrever o conteúdo de informação do banco de dados ao invés das estruturas de armazenamento que serão necessárias para gerenciar essa informação.

Suas vantagens são a indicação de dados e seus relacionamentos da forma como percebidos no mundo real, independência de detalhes de representação de SGBDs, fácil compreensão pelo usuário leigo e facilita a manutenção dos dados e a modificação dos requisitos.

O **projeto lógico** tem por objetivo avaliar o esquema conceitual frente às necessidades de uso do banco de dados pelos usuários/aplicações, para realizar possíveis refinamentos visando alcançar maior desempenho das operações sobre o banco de dados. A tarefa final do projeto lógico é a geração do esquema lógico decorrente do esquema conceitual resultante do refinamento.

Um **esquema lógico** é uma descrição da estrutura do banco de dados que pode ser processada por um SGBD. Um **modelo lógico** é usado para especificar esquemas lógicos. Os modelos lógicos mais conhecidos para bancos de dados convencionais, pertencem a três classes: em redes, hierárquico e relacional, sendo o último o mais amplamente usado. O projeto lógico depende da classe do modelo de dados usado pelo SGBD, mas não do SGBD específico usado.

[1] Sistemas gerenciadores de bancos de dados SGBD ou DBMS Database Management System - são sistemas que gerenciam BDs ou são linguagens utilizadas para manter os BDs

O **projeto físico** toma por base o esquema lógico para construir o esquema físico. Um **esquema físico** é uma descrição da implementação do banco de dados em memória secundária; ele descreve as estruturas de armazenamento e métodos de acesso usados para efetivamente realizar o acesso aos dados. O projeto físico é direcionado para um SGBD específico, por exemplo: Oracle, Sybase, OpenIngres, Access, SQL Server dentre outros.

Decisões tomadas durante o projeto físico, para melhorar o desempenho, podem afetar a estrutura do esquema lógico. Uma vez que o projeto físico do banco de dados é completado, os esquemas lógico e físico são expressos usando a linguagem de definição de dados do SGBD adotado. O banco de dados é criado e populado e pode ser testado para se tornar operacional.

O esquema físico do banco de dados é influenciado pelas fases por que passou na construção do banco de dados. A fase de projeto conceitual é tida como uma das mais (senão a mais) delicadas em todo esse processo, pois depende muito da habilidade do projetista do banco de dados e das qualidades do modelo de dados adotado para a elaboração do esquema conceitual. A meta nessa fase é obter um esquema conceitual do banco de dados que seja tão completo e expressivo quanto possível. Esse esquema deve procurar expressar o máximo da semântica envolvida na informação. Mecanismos de representação de alto nível são empregados, tais como representação de hierarquias de subconjunto e de generalização, representação de restrições de cardinalidade e de atributos compostos e multivalorados. Para a representação do esquema conceitual geralmente utiliza-se uma extensão do modelo Entidade-Relacionamento.

O projeto conceitual de um banco de dados não pode ser totalmente auxiliado por ferramentas automáticas; cabe ao projetista entender e transformar os requisitos dos usuários em esquemas conceituais. O projeto conceitual é, assim, a fase mais crítica do projeto do banco de dados. Ele não depende somente da habilidade e experiência do projetista, mas também da cooperação dos usuários que são responsáveis por descrever suas necessidades. O esquema conceitual deve fazer parte da documentação do processo de projeto, sendo utilizado durante a operação e manutenção do banco de dados, pois facilita o entendimento dos esquemas de dados e das aplicações que os utilizam.

Para auxiliar o projetista a elaborar o projeto conceitual de um banco de dados existem as **abstrações de dados**, que apresentam as vantagens:

- Ajudam o projetista a entender, classificar e modelar a realidade,
- Melhoram a eficiência de implementações subsequentes,
- Permitem melhor representar a semântica das novas aplicações de banco de dados, provenientes de áreas não tradicionais.

As abstrações comumente usadas no projeto conceitual são: classificação, agregação e generalização.

Abstração de classificação: é usada para agrupar objetos similares, caracterizados por propriedades comuns, em classes de objetos.

A classificação estabelece um relacionamento É-INSTANCIA-DE entre cada elemento da classe e a classe.

Abstração de agregação: é um conceito de abstração para construir objetos compostos a partir de seus objetos componentes.

- Uma entidade é uma agregação de atributos: PESSOA, composta por Nome, Sexo, Profissão;
- Um relacionamento é uma agregação de entidades e atributos;
- Um atributo composto é uma agregação de atributos;
- Pode-se agregar entidades relacionadas entre si, compondo uma entidade de nível mais alto.

  Essa abstração estabelece um relacionamento É-PARTE-DE entre os componentes e a classe.

  Abstração de generalização: define um relacionamento de subconjunto entre os elementos de duas ou mais classes.

  Essa abstração estabelece um relacionamento É-UM entre a classe pai (chamada superclasse) e cada classe filha (subclasse).

2.2.2 Projeto de banco de dados com perspectiva Bottom-up

Parte do baixo nível para atingir uma representação mais em alto nível. Enfatiza a descrição de dados já existentes. É também chamado de Engenharia Reversa de BD, pois é aplicado em casos onde existem fontes de dados ou sistemas informatizados (legados) sem BD.

É dividido em Cinco etapas:

- 1. coleta de fontes de dados
- 2. representação em uma tabela não-normalizada
- 3. normalização
- 4. integração de esquemas relacionais das fontes
- 5. engenharia reversa do esquema relacional

A Coleta de fontes de dados organiza os dados operacionais que serão representados em tabelas não normalizadas na segunda etapa. A terceira etapa normaliza as tabelas da etapa anterior decompondo sistematicamente em várias outras tabelas, eliminando redundâncias no armazenamento e organização dos dados em

entidades lógicas. Este processo é baseado na aplicação de regras. A quarta etapa unifica as fontes de dados normalizadas integrando as tabelas que mantêm as mesmas entidades e relacionamentos, eliminando tabelas redundantes. Finalizando, a quinta e última etapa obtem o esquema conceitual dos dados, baseando-se em regras de conversão e análise dos dados.

2.3 Modelos de Banco de Dados

2.3.1 O modelo relacional

Esta subseção foi toda baseada na obra de [**?**] O Modelo de Dados relacional representa os dados contidos em um Banco de Dados através de relações. Estas relações contêm informações sobre as entidades representadas e seus relacionamentos. O Modelo Relacional é claramente baseado no conceito de matrizes, onde as chamadas linhas (das matrizes) seriam os registros e as colunas (das matrizes) seriam os campos.

Cada linha é chamada de TUPLA e cada coluna é chamada de ATRIBUTO. O conjunto de valores passíveis de serem assumidos por um atributo, será intitulado de DOMÍNIO. O domínio consiste de um grupo de valores atômicos a partir dos quais um ou mais atributos retiram seus valores reais.

O esquema de uma relação nada mais é que os campos (colunas) existentes em uma tabela. Já a instância da relação consiste no conjunto de valores que cada atributo assume em um determinado instante. Portanto, os dados armazenados no Banco de Dados, são formados pelas instâncias das relações.

As relações não podem ser duplicadas, a ordem de entrada de dados no Banco de Dados não deverá ter qualquer importância para as relações, no que concerne ao seu tratamento. Os atributos deverão ser atômicos, isto é, não são passíveis de novas divisões.

Chave Primária é o Atributo que defini um registro, dentre uma coleção de registros. Chave Secundária (Ternária, etc), serão chaves que possibilitarão pesquisas ou ordenações alternativas, ou seja, diferentes da ordem criada a partir da chave primária ou da ordenação natural (física) da tabela. Chave Composta é aquela chave que contém mais de um atributo. Chave Estrangeira é aquela chave que permite a ligação lógica entre uma tabela (onde ela se encontra) com outra na qual ele é chave primária

Limitações do modelo relacional

O modelo relacional de banco de dados possuíam uma série de limitações onde podemos citar:

- Os banco de dados relacionais não possibilitavam a modelagem de objetos.

- Possuíam um conjunto limitado de tipos de dados e não permitiam a criação de novos tipos.

- A linguagem de consulta utilizada não era compatível com as novas linguagens de programação orientadas a objetos.

Todas essa limitações culminaram com o surgimento de novos paradigmas, onde podemos citar os banco de dados orientados a objetos. Estes por sua vez, devido à incompatibilidade com o modelo relacional não fez muito sucesso, evoluindo então para o modelo objeto relacional.

2.3.2 Banco de dados objeto relacionais

Esta subseção foi baseada na obra de [CHANG - 1996] Banco de dados objeto relacional é um termo usado para descrever um banco de dados que evoluiu do modelo relacional para um banco de dados híbrido que contém tecnologia relacional e tecnologia de objeto.

Esta nova tecnologia é totalmente compatível com os banco de dados relacionais tradicionais. É possível fazer a migração sem precisar reescrevê-los.

A integração de representações orientadas a objetos e relacionais é semanticamente clara e consideravelmente mais poderosa do que as representações relacionais ou orientadas a objeto sozinhas.

Ao contrário da estratégia de banco de dados relacional, que trata com dados no nível mais baixo possível, uma série de linhas e colunas, o modelo orientado a objetos trata com dados em um nível muito mais alto; ele trata com os objetos que circundam os dados. Objetos são representações de software de entidades do mundo real. Para capturar as características e recursos do mundo real, os objetos são compostos de atributos e informações operacionais.

Quando os objetos são semelhantes uns aos outros nos comportamentos e outros atributos, eles podem ser reunidos em uma classe.

Além dessas, existem ainda diversas outras características que fazem da tecnologia de banco de dados objeto relacionais uma ferramenta muito poderosa e superior aos bancos de dados relacionais.

2.3.3 Banco de dados Temporais

Esta subseção foi baseada na obra de [**?**] Bancos de dados em tempo real são um subconjunto dos bancos de dados temporais, ambos tratam internamente a dimensão temporal. Um sistema em tempo real gera uma grande quantidade de informação em poucas unidades de tempo. A idéia de um sistema gerenciador de banco de dados em tempo real retiraria a carga do controle de tempo do projeto de banco de dados.

Nos últimos anos as pesquisas na área de Sistemas de Gerenciamento de Banco de Dados em Tempo-real (SGBD-TR) têm se intensificado. Estas pesquisas são motivadas pelo fato de que determinadas aplicações precisam tratar com grandes volumes de dados, além de dados e transações com restrições temporais. Um possível encaminhamento visando solucionar tal problema seria buscar a integração entre as tecnologias de Sistemas de Gerenciamento de Banco de Dados (SGBD) e Sistemas em Tempo-real (STR). Entretanto, como discutido em, a simples integração destas tecnologias (conceitos, mecanismos, ferramentas) não é suficiente para se desenvolver um SGBD-TR. Questões como modelagem conceitual, controle de concorrência, escalonamento, recuperação, entre outras, estão sendo consideradas e pesquisadas.

Dada a complexidade das aplicações em tempo-real, muitos pesquisadores acreditam que o paradigma da orientação a objetos é o mais apropriado para tratar com tais aplicações. Também se faz necessário o emprego de ferramentas eficazes com base formal para modelar tais aplicações.

A orientação a objetos é uma forma mais natural de definir o mundo real em um ambiente computacional. Como o próprio nome diz, trata de objetos, com atributos e métodos, assim como no mundo real.

Um banco de dados orientado a objetos, ao contrário do relacional, pode adicionar uma dimensão temporal nos objetos instanciados.

Um SGBD-TR pode ser visto como a integração de um SGBD convencional com um STR [1]. Assim, um SGBD-TR além de processar transações e garantir a integridade dos dados, deve também operar em tempo-real, para satisfazer as restrições temporais impostas aos dados e transações. Destacam-se como características principais de um SGBD-TR a noção de dados temporalmente consistentes, e a habilidade para definir restrições temporais às transações.

Deve-se observar que há casos em que as restrições temporais impostas às transações precisam ser satisfeitas com o objetivo de manter a consistência temporal dos dados. Em algumas situações estas restrições só podem ser satisfeitas se a consistência lógica dos dados for sacrificada. Em outras, para manter a consistência lógica dos dados, a consistência temporal deve ser sacrificada. Uma vez que estas situações são bastante freqüentes em SGBD-TR, os resultados produzidos por uma transação não precisam ser totalmente corretos, ou seja, podem ser imprecisos. Portanto as propriedades ACID não precisam ser cumpridas totalmente, e foram redefinidas para SGBD-TR. Geralmente a imprecisão permitida deve ser limitada.

Idealmente os dados gravados em um BDTR devem ser idênticos em valor ao seu correspondente no ambiente. No entanto, geralmente há um atraso de atualização no BDTR, o que pode levar a inconsistências entre estes valores. Portanto é necessário um mecanismo para verificar a consistência temporal do dado face à extensão do atraso na sua atualização. Um tal mecanismo, pode ser baseado na definição de um rótulo de tempo (timestamp) e de um intervalo de validade ab-

soluta (avi) para cada dado ou conjunto de dados a serem gravados no banco de dados.

A consistência temporal dos dados pode ser medida de duas maneiras: Consistência absoluta entre o valor real de um item de dado no ambiente de aplicação e o valor correspondente no banco de dados. Então, a idade de um item de dado deve estar dentro do intervalo de validade absoluta especificado. Esta medida surge da necessidade de manter o banco de dados consistente com o estado real do ambiente. Consistência relativa entre os itens de dados que são usados para calcular outros dados. Os itens de dados que serão usados na computação de outros dados devem ser gravados dentro de um intervalo de tempo relativo especificado. Esta medida surge da necessidade de assegurar que dados que serão usados na computação de outros dados representem aproximadamente o mesmo instante de tempo do ambiente.

De modo geral as transações em um SGBD-TR podem ser classificadas como: transações de escrita (ou sensores), transações de atualização e transações de leitura. As transações de escrita, tipicamente periódicas, obtêm o estado do ambiente e escrevem os dados no banco de dados. As transações de atualização tanto podem ler quanto escrever no banco de dados periodicamente ou aperiodicamente. As transações de leitura lêem dados do banco de dados e também podem ser periódicas ou aperiódicas.

Uma transação em SGBD-TR também pode ser classificada com base no efeito de não cumprir sua restrição temporal, em estrita, suave ou firme1.

Estrita: uma transação é estrita quando qualquer resultado produzido após seu deadline é inútil para o sistema. Isso significa que qualquer transação estrita deve ser abortada quando não puder cumprir seu deadline, independentemente de sua conseqüência.

Suave: uma transação é suave quando o resultado produzido após seu deadline sempre tem algum valor, que vai perdendo sua utilidade à medida que se distancia de seu deadline.

Firme: uma transação é firme quando a perda do deadline não gera nenhum efeito ou valor para o sistema. Geralmente estas transações são recomeçadas quando perdem seu deadline.

As restrições de tempo impostas às transações podem se originar de dois tipos de requisitos: requisitos de consistência temporal dos dados e requisitos impostos pelo sistema em relação ao tempo. O primeiro assume a forma de requisito de periodicidade, por exemplo: a cada 20 segundos verifique a temperatura do ambiente. O segundo geralmente assume a forma de deadlines impostos às transações aperiódicas, por exemplo: se temperatura > 1000 graus, adicione líquido refrigerador ao reator dentro de 5 segundos.

Oracle Time Series Cartridge

Esta seção foi baseada na obra de [OZSOYOGLU - 1995] e [NAVATHE - 1994] A partir da sua versão 8*i*, o SGBD Oracle passou a contar com uma extensão conhecida como Time Series Cartridge ou Cartucho de Séries de Tempo. Possibilita o armazenamento e a recuperação de dados temporais.

Uma série de tempo é o montante de dados de entrada de rótulos temporais (timestamps). Cada atributo ou coluna em uma tabela tem associado um rótulo temporal, ou seja, uma marca que situa o dado ou objeto no tempo.

O Time Series utiliza conceitos próprios para tratar informações temporais, como a utilização ou não de calendários. Aplicações de bancos de dados temporais, basicamente são implementadas de duas formas, com uso de calendários ou sem calendários. Calendários são objetos do Time Series que permitem vincular as informações a datas ou a períodos de tempo definidos. Existem dois modelos para tratar dados de séries de tempo: o modelo de séries de tempo regular e o irregular. No modelo irregular os dados da série não estão necessariamente associados ao uso de calendários, enquanto que no modelo regular obrigatoriamente devem estar associados.

A definição da utilização de um ou outro modelo de série de tempo varia com a especificação da modelagem temporal. Geralmente, calendários não são necessários quando o rótulo temporal não tiver um padrão ou quando tiver um padrão, mas este não estiver correto.

Capítulo 3

Metodologia

Inicialmente será realizado um estudo em cima do sistema SmartMine para se entender o seu funcionamento. Este é um sistema bastante complexo o que demandará muito tempo até que se esteja devidamente inteirado sobre seu funcionamento.

Após concluída esta etapa, segue-se a análise específica do banco de dados da forma como está implementado atualmente. Nesta fase o importante é entender como o banco de dados funciona, quais os problemas encontrados, por que eles ocorrem, o que estes problemas acarretam para o sistema e como os operadores e o próprio sistema tratam estes erros. Verificar também como está a integração do banco de dados com o sistema, se as camadas de aplicação e de acesso aos dados estão bem definidas, se será necessário grandes esforços em uma possível remodelagem geral do banco de dados, entre outros.

Em seguida, partiremos à procura de possíveis soluções. A principio, tentando manter a estrutura original para que não seja necessário fazer grandes alterações no sistema. A busca por novas tecnologias que poderiam ajudar na resolução dos problemas também faz parte desta etapa. A nova tecnologia deverá ser analisada, ponderando sobre pontos positivos e negativos em sua adoção. Possíveis soluções serão enumeradas e apresentadas juntamente com um relatório descrevendo suas características e impactos em sua adoção. Entre as candidatas estão a expansão Time Series Cartridge da Oracle, que usa a tecnologia objeto relacional, o uso de banco de dados em tempo real e banco de dados temporais. Se for necessária a remodelagem de todo o banco de dados, será escolhida uma das perspectivas: top-down ou bottom-up, sendo que a última é mais aplicável ao caso, já que existe uma implementação concluída. Apesar de já existirem candidatas à solução, este trabalho não se limitará somente aos temas citados no referencial. Novas pesquisas serão feitas em busca da solução ideal.

A fase seguinte será a implementação da melhor solução encontrada na fase anterior. Seguindo-se a análise da implementação, verificando se os erros foram realmente corrigidos e se não surgiram novos problemas. Pode ser que seja necessário

implementar mais de uma solução até encontrar a ideal, que garanta a integridade dos dados, desempenho adequado e facilidade na manutenção, que são os requisitos a serem atendidos.

Capítulo 4

Resultados e discussões

4.1 O Sistema SmartMine®

O SmartMine® é software intensivo de gerenciamento de mina em tempo real. É uma solução integrada em software e hardware, que usa técnicas modernas de simulação orientada a eventos e métodos estocásticos de otimização [23].

Os principais módulos do Smartmine são:

- O módulo de Controle da Operação da mina - Dispatch Module;
- O módulo de Planejamento da mina - Manager Module;
- O módulo de consultas a relatórios e acompanhamento da mina em tempo real - Real Time Web Module;
- O módulo do sistema embarcado usando informações de GPS - GPS Tracking Module;
- O módulo de Otimização do Despacho - Optimization Module.

4.1.1 Dispatch Module

O módulo de controle da mina é a interface entre o SmartMine e os movimentos de materiais e equipamentos que ocorrem durante a operação da mina. Este módulo é responsável por capturar estes eventos, armazená-los no banco de dados e indicar ao operador do sistema o que está ocorrendo, incluindo carga/descarga, alteração dos estados dos equipamentos. Ele funciona vinte e quatro horas por dia.

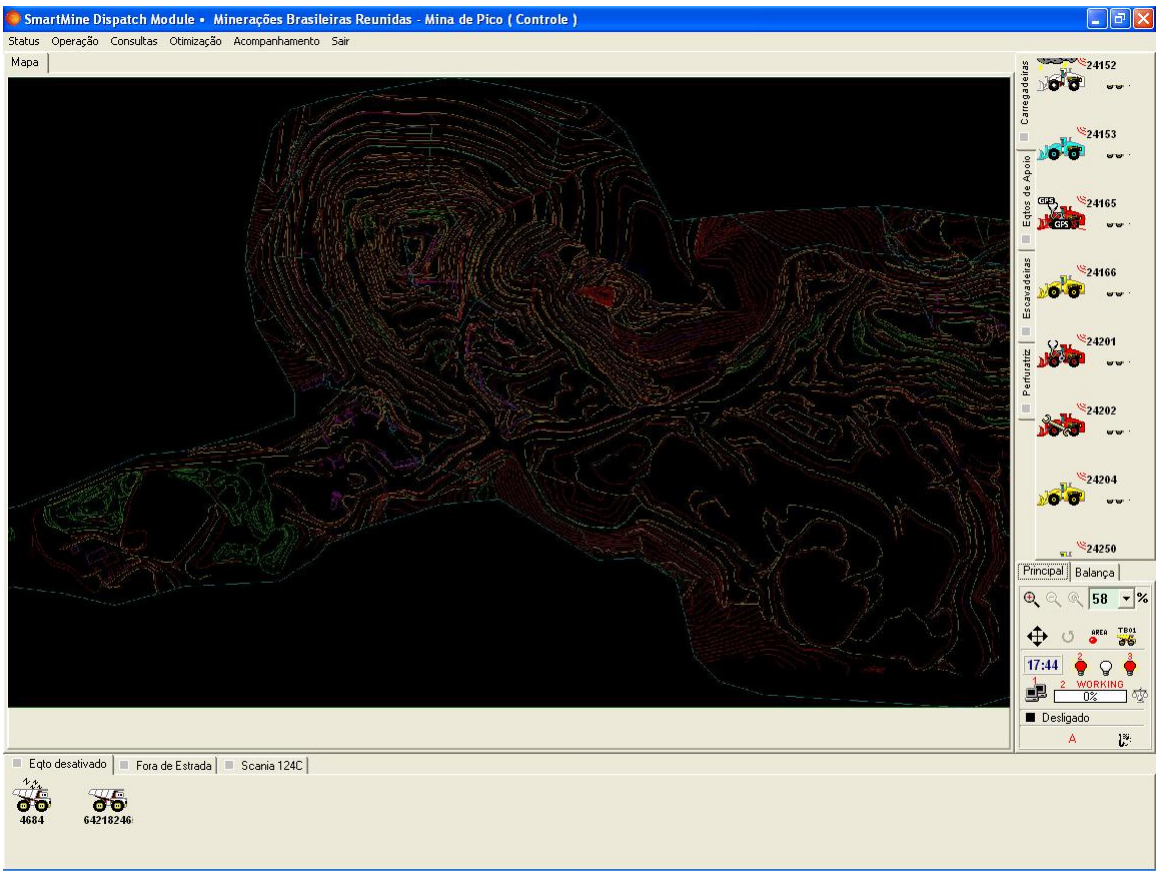

Figura 4.1: Dispatch Module

4.1.2 Manager Module

Representa a interface entre os departamentos de planejamento e operação de mina. É o módulo em que se entra com os dados do plano de mina, onde são definidos todos os caminhões, equipamentos de carga e demais equipamentos de mina com suas propriedades, como capacidade de carga, identificação, autonomia, grupo, modelo entre outros.

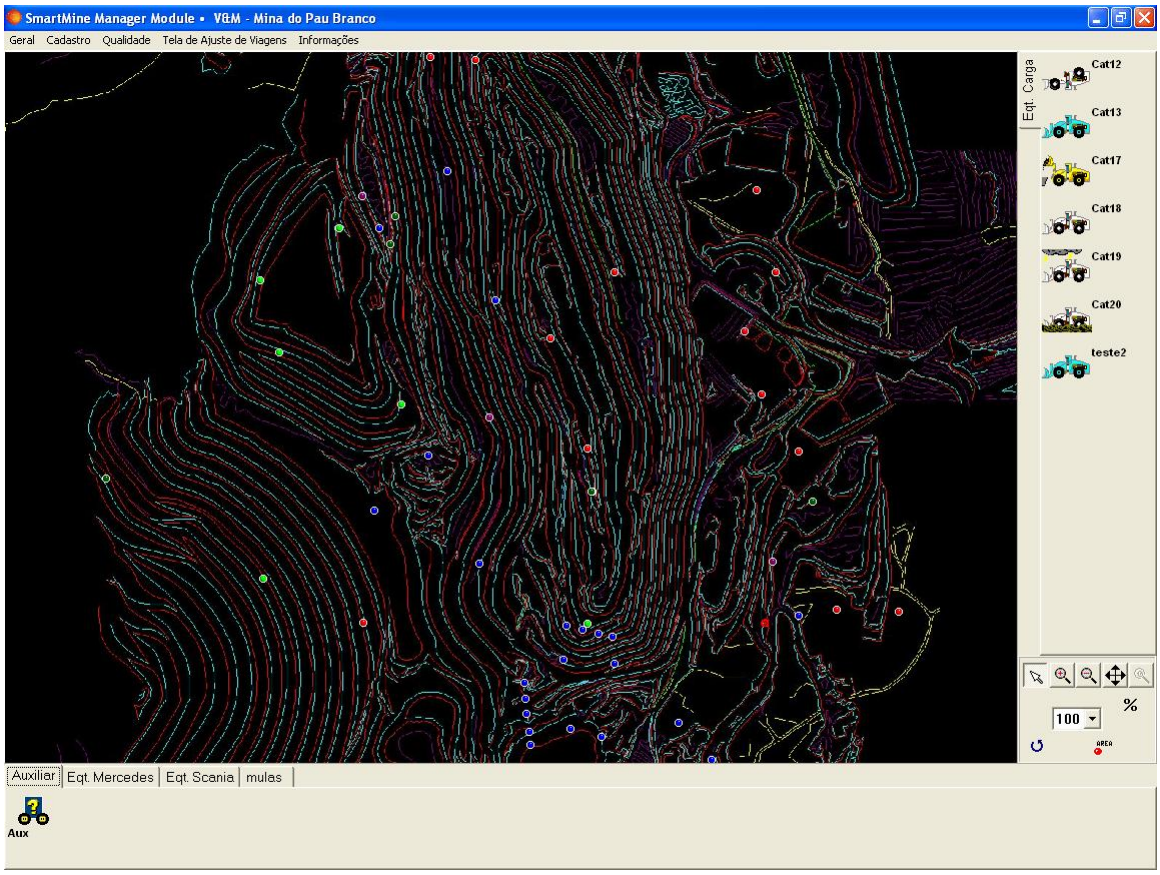

Figura 4.2: Manager Module

4.1.3 Real Time Web Module

É o módulo de consulta aos dados que o Smartmine armazena. É uma aplicação Web (o que possibilita que esses dados sejam consultados de qualquer computador na rede), que permite que todas as variáveis importantes sejam acompanhadas exatamente como são apresentadas na mina, naquele momento.

Figura 4.3: Real time Web Module

4.1.4 GPS Tracking Module

É o módulo que permite a maior parte da automatização da operação do sistema. Basicamente, é constituído por:

- Tracker SMT 100 - um computador móvel embarcado usado para aquisição de posicionamento, aquisição de eventos e status dos caminhões e envio para o Módulo de Despacho;
- Loader SMT 100 - computador embarcado para escavadeiras e carregadeiras;
- Access Point SMT 100 - é um gateway para o software SmartMine, coletando todas as informações geradas pela escavadeira e caminhões através da rede de rádio sem fio;
- Base Station - as estações base são acopladas ao Access Point, fazendo a conexão wireless com os equipamentos móveis. Uma estação base também pode ser usada como estação repetidora.

O Tracker é instalado nos equipamentos, fazendo dos operadores dos usuários do sistema. Dessa forma, a responsabilidade de indicar todas as trocas de estado deixa de ser do operador do Dispatch Module.

O Tracker pega informações do GPS, empacota-as e as envia para os outros módulos, principalmente o Dispatch Module.

4.1.5 Optimization Module

Recebe a todo tempo do Dispatch Module a situação em tempo real da mina. Este módulo realiza então múltiplas simulações e otimizações da operação da mina esperada para os próximos minutos. Estas simulações fazem parte de um processo contínuo de otimização, de modo que qualquer alteração nos parâmetros é levada em consideração. A partir das simulações, são sugeridas as alocações e os despachos dos caminhões de forma a otimizar a operação da mineração. Por se tratar de um problema combinatorial, sem resolução exata em tempo hábil, o Optimization Module utiliza a técnica de Algoritmos Genéticos para gerar as soluções.

As sugestões de despacho geradas podem ser enviadas automaticamente para o Tracker, esperando ou não a confirmação do operador do caminhão.

4.1.6 O banco de dados do SmartMine

O Manager Module realiza os cadastros básicos para o funcionamento do sistema. Estes cadastros não exigem tanto desempenho do banco de dados. Portanto o nosso foco se concentrará principalmente no dispatch module, que recebe todas as informações em tempo real enviada pelo hardware dos equipamentos, trata estas informações e as insere no banco de dados.

O dispatch module recebe os dados na forma de troca de eventos. Uma tabela que contém todos os equipamentos disponíveis na mina recebe a informação do evento atual e armazena esta informação enquanto o evento estiver ocorrendo. Quando o equipamento muda seu estado, o dispatch recebe as informações referentes ao novo estado atual. Então processa este novo estado - através de uma máquina de estados construída dentro do banco de dados através de funções e procedimentos - validando a sua troca, fecha o evento anterior colocando a data final como sendo a data inicio do evento atual - para garantir continuidade na linha de tempo de operação do equipamento - e insere na tabela de log de operação. Dependendo do tipo do estado podem ser feitas outras ações como a sumarização nos casos de eventos que contêm produção. Depois de inserido no log de operação o estado anterior é substituído pelo evento atual reiniciando o ciclo.

O processo de sumarização foi uma das estratégias utilizadas para garantir um desempenho aceitável principalmente nos relatórios do Web Module. Os dados inseridos no log de operação são agrupados em tabelas auxiliares através de campos chaves. Por exemplo, a sumarização por hora e equipamento, onde os eventos de cada equipamento são agrupados de acordo com a hora de chegada. Desta forma, todos os n registros de um determinado equipamento em uma determinada hora no log se tornará apenas um único registro sumarizado.

4.2 Resultados

4.2.1 Uma análise do Banco de Dados antes da remodelagem

A implementação do banco de dados antes da remodelagem tinha algumas deficiências, como por exemplo o próprio desenho da máquina de estados que possuía algumas falhas que não processavam corretamente determinadas seqüências de eventos. Além disso era bastante confusa o que impedia qualquer tentativa de correção.

A máquina de estados era executada no momento da entrada do evento no banco de dados, ou seja, antes mesmo de validar o novo evento, este era processado. Todos os eventos eram tratados de forma equivalente e não era possível desfazer com segurança uma sumarização.

Não havia também qualquer tipo de identificador para os eventos. Quando era preciso buscar um determinado evento na tabela de log de operação era preciso decodificar códigos de estados e buscar por equipamento, estado e data/hora do início e fim do evento. Isto deixava qualquer consulta na tabela de log extremamente demorada.

Mas o problema mais grave era mesmo o da sumarização já citado anteriormente. O sistema realiza vários tipos de sumarização. Entre elas: sumarização de tempo por hora e/ou por turno, sumarização por equipamento, por estado, etc. Como ela era feita no momento da entrada do evento no sistema, se este evento

fosse invalidado mais tarde a sua produção continuaria sumarizada. Isto causava grandes diferenças na comparação dos dados da tabela de log de operação e das tabelas auxiliares que guardavam os dados sumarizados. E essas diferenças se refletiam principalmente nas variáveis de tomada de decisão e nos relatórios diminuindo a confiabilidade do sistema.

Uma outra operação do sistema que era bastante prejudicada pela modelagem do banco de dados era a chamada contingência. Esta operação pode ser definida como sendo o lançamento manual de eventos para determinados períodos de tempo. A contingência pode ocorrer no evento atual, substituindo-se o evento atual por outro a escolha do operador, ou pode ocorrer em um intervalo de tempo no passado. Neste caso, os eventos anteriores serão substituídos inteiramente pelos novos. Para isto, é preciso buscar todos os eventos que estariam dentro do intervalo escolhido, apagá-los e inserir novamente os novos eventos. Como já foi dito anteriormente, consultas na tabela de log de operações eram extremamente lentas, pois não existia um identificador único nesta tabela. Além disso, como o sistema tratava cada evento independentemente e o usuário poderia escolher qualquer intervalo, poderia ocorrer a quebra de eventos. Esta quebra de eventos poderia invalidar eventos anteriores ou posteriores ao período de contingência escolhido. Em alguns casos poderia ocorrer até mesmo a quebra na linha de tempo de operação de um determinado equipamento.

Todos estes pequenos problemas às vezes passavam totalmente despercebidos na operação diária do sistema. Eram notados somente em análises um pouco mais apuradas e também em relatórios detalhados. Com isto a confiabilidade do sistema ficava bastante comprometida. A manutenção do sistema se tornava um pouco penosa e cansativa e os programadores se perdiam nas dezenas de milhares de linhas de código.

4.2.2 A proposta de remodelagem

A mudança proposta na remodelagem do banco de dados passou acima de tudo por uma mudança de postura em relação à arquitetura do sistema. Através de testes foi comprovado que tarefas executadas em procedimentos armazenados dentro do próprio banco além de mais eficientes eram mais confiáveis do que funções e procedimentos criados dentro do sistema.

A etapa seguinte era a de analisar e testar as tecnologias emergentes e disponíveis atualmente no mercado.

O pacote Time Series disponível no produto da Oracle apresentava grandes propostas em relação ao gerenciamento de datas. Prometia um controle interno eficiente da dimensão temporal bastante explorada no sistema SmartMine. Apesar disso, por motivos de incompatibilidade com o produto SQL Server da Microsoft a idéia de se adotar esta tecnologia foi logo abandonada.

A solução deveria ser portável para os dois sistemas gerenciadores de banco de

dados - SGBD, tanto o ORACLE quando o SQL Server, pois o sistema SmartMine estava disponível nas duas plataformas, variando de cliente para cliente. Com esta postura, as novas tecnologias foram praticamente abandonadas já que não existe um certo nivelamento entre os dois SGBD's. Restou a tecnologia dos bancos de dados relacionais já utilizadas. Se não era possível inovar, então a solução era otimizar a forma como estava sendo feito. Talvez uma mudança na lógica, mas neste enfoque esbarrávamos em uma outra barreira que era a do sistema legado. Não podíamos mudar radicalmente, pois o banco de dados precisava manter compatibilidade com o sistema. Era inviável ter que reescrever grandes blocos de código do sistema. Havia um sistema em funcionamento e clientes dependentes deste sistema. A equipe de desenvolvimento não poderia ser muito exigida nesta reestruturação.

Com este pensamento, voltamos nosso enfoque às regras do negócio envolvidas no sistema, analisando e comparando com as dificuldades que enfrentávamos atualmente. E foi partindo dos problemas que tínhamos em mãos que passamos a pensar em soluções. A primeira delas foi que era preciso ter um identificador para cada evento dentro da tabela de log de operações. Este identificador seria muito útil em consultas nesta tabela, otimizando e diminuindo o tempo de busca.

Outro ponto que resolvemos adotar foi a adoção de procedimentos e funções no lugar de gatilhos - triggers. Este ponto merece destaque pois apesar dos gatilhos terem desempenho comprovadamente eficiente, o fato destes serem executados automaticamente sem necessidade de chamadas explícitas dificultam o entendimento e aprendizagem, como também a depuração para busca de possíveis erros.

Mas o conceito por trás de toda a reestruturação do banco de dados se resume basicamente à mudança na lógica de tratamento dos eventos. Foi adotado o conceito de ciclo ou viagem, assim como na operação real de mineração. Um ciclo era um determinado conjunto de eventos consecutivos.

Existem três tipos básicos de eventos:

- Eventos de operação: são estados básicos de movimentação dos equipamentos dentro de uma mina. Como exemplo podemos citar os eventos de carregamento, basculamento, movimentando cheio, dentre outros.

- Eventos Neutros: são eventos que não influem diretamente na operação da mina. Não alteram a rotina e nem alteram rotas. Como por exemplo filas de carregamento, filas de basculamento, etc.

- Atrasos operacionais: são eventos que alteram o fluxo de mineração. Podem ser planejados ou podem surgir também como imprevistos. Como exemplo podemos citar manutenções preventivas, mau tempo, falta de operador, troca de turno, etc.

Com isso surgiram algumas classificações para as seqüências destes eventos:

- Ciclo completo perfeito: são ciclos formados exclusivamente pelos quatro eventos básicos de operação (movimentando vazio - carregando - movimentando cheio - basculando).

- Ciclo Completo: formado pelos estados básicos de operação acrescidos de estados neutros intercalados.

- Ciclo Completo fragmentado: é um ciclo completo onde pode haver alguma repetição de um dos estados de operação.

- Ciclo incompleto com produção: um ciclo onde não ocorrem os quatro estados básicos de operação, mas necessariamente ocorreu um basculamento.

- Ciclo incompleto sem produção: um ciclo que contem estados de operação mas não ocorreu o estado de basculamento

- Alimentação direta: ciclo formado unicamente pelo evento de produção direta, operação realizada somente por equipamentos de carga.

- Operação de carregamento: ciclo formado exclusivamente para equipamentos de carga quando estes realizam eventos de carregamento.

Os atrasos operacionais são eventos que não acarretam a formação de ciclos.

Desta forma pudemos agrupar os eventos impedindo que viagens sejam quebradas ao meio, facilitando assim o processo de sumarização e de-sumarização.

Estudamos também a possibilidade de utilização do conceito de segmentação de tempo, que consiste na quebra do tempo em horas exatas.Por exemplo um evento que se inicia às 11:48:30 e termina às 12:04:03, segmentado se transformaria em 2 registros, um de 11:48:30 ate às 12:00:00 e outro de 12:00:00 ate o final do evento as 12:04:03. Isto num primeiro momento geraria um esforço extra quando da chegada dos eventos, mas posteriormente facilitaria bastante na geração dos relatórios, pois funcionalmente os relatórios são gerados sempre em horas exatas.

Num primeiro momento pensamos que a segmentação proporcionaria um ganho de desempenho consideravel nas consultas realizadas pois guardaria os eventos com horas inteiras e a grande maioria das consultas realizadas são desta forma.

Apesar dos aspectos positivos deste conceito na prática ele não se mostrou tão funcional. Testes práticos demonstraram que seriam necessárias grandes mudanças e que o ganho no desempenho não seria suficiente para justificar sua adoção, visto que ganho no desempenho seria compensado pelo maior esforço na hora de armazenas estes dados.

Na verdade este conceito poderá ser utilizado em partes específicas do sistema com grandes ganhos de desempenho, assim como a sumarização.

4.2.3 A implementação da remodelagem

O primeiro passo foi modelar uma máquina de estados simples que atendesse o fluxo de entrada de eventos que havíamos estudado, prevendo todas as possíveis combinações na entrada de eventos. Esta foi criada da forma como mostrada na Figura 4.4.

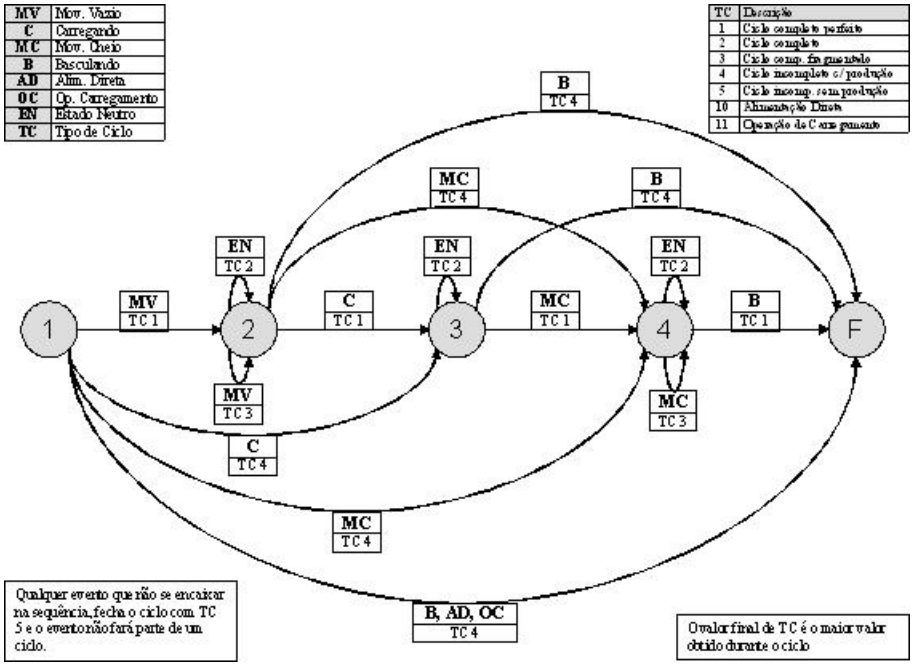

Figura 4.4: Máquina de estados

O passo seguinte foi a implementação da máquina de estados na forma de uma função implementada diretamente no banco de dados. Esta função processa o estado atual de acordo com os estados anteriores armazenados temporariamente na tabela da máquina de estados. Após o processamento e a obtenção do resultado do tipo do ciclo, esta tabela é liberada e os eventos são direcionados para a tabela de log de operações realizando todas as operações necessárias.

Por razões de desempenho a tabela de log de operações continuou sendo dividida de acordo com o tipo de equipamento. Existindo assim três tabelas, uma para equipamentos de transporte, uma para equipamentos de carga e outra para equipamentos auxiliares.

Alguns campos também tiveram de ser adicionados às tabelas de log. Informações como código e tipo do ciclo e identificador do evento passaram a fazer parte do log de operação. Outros campos foram renomeados aumentando assim a legibilidade.

Todos os gatilhos foram substituídos por procedimentos e funções adequados

para cada atividade realizada. Chamadas explícitas foram adicionadas.

Foram incluídas também novas tabelas de logs como o log de ciclos e o log de turnos. Estas tabelas facilitarão o armazenamento e a recuperação de informações relativas à operação da mina, gerando relatórios mais ricos em detalhes e mais confiáveis nas tomadas de decisão.

Além destas foram criadas também novas tabelas de logs de erros para facilitar a depuração e a correção de possíveis erros que possam vir a ocorrer. Estas tabelas não possuem qualquer tipo de checagem de integridade justamente para garantir que se a informação não foi salva corretamente na sua tabela original, com certeza as informações estarão disponíveis neste log de erros.

Em todos os procedimentos e funções foram adicionados tratamentos de erro que seguem o padrão estabelecido internamente na empresa, retornando códigos padrão para as camadas superiores ate chegar ao nível do usuário com mensagens claras e explicativas e registrando qualquer anormalidade do sistema em umas das tabelas de log de erros.

Os scripts de linguagem SQL internos ao sistema foram revisados. Ao longo de todo o desenvolvimento desta remodelagem alguns conceitos foram sendo desenvolvidos e validados em questão de melhora de desempenho. Como por exemplo, a seqüência dos campos nas restrições de consultas. Primeiramente devem ser utilizados os campos que compõem a chave primaria da tabela, em seguida, os campos que contêm índices e a seguir os demais campos.

Outro falor descoberto que influi diretamente no desempenho das consultas é o número de tabelas envolvidas diretamente nas restrições. Quando se faz uma união de várias tabelas na mesma consulta, o SGBD faz um cruzamento de todas as linhas das tabelas envolvidas para em seguida aplicas as regras de restrições. Desta forma há um grande esforço inicial apenas nesta junção diminuindo o desempenho. Nestes casos é preferível realizar várias seleções independentes e no final junta-las a fazer tudo de uma unica vez.

A quantidade de dados a ser selecionada também influi no desempenho na seleção. É preciso levar em consideração a atividade de leitura em disco. Desta forma podemos prever qual a melhor forma para selecionarmos uma determinada quantidade de dados afim de diminuirmos os movimentos da cabeça de leitura do disco.

4.2.4 Análise dos resultados

Após implementada a solução descrita na seção anterior e realizado vários testes de desempenho passou-se para a etapa de análise dos problemas e verificação de sua solução.

Algumas partes do banco de dados foram bastante modificadas, incluindo até mesmo mudanças de conceito, apesar disso, não exigiu grandes esforços para se adequar o sistema.

O nova máquina de estados, em comparação com a anterior, ficou bem mais clara e legível, facilitando assim a manutenção do próprio banco de dados, que era um dos nossos objetivos.

O problema da inconsistência de dados também foi solucionado, tanto através do redesenho da máquina de estados, quanto em relação às mudanças no modo de entrada e armazenamento dos dados, leia-se a adoção do conceito de ciclos.

Outro ponto que merece destaque e que estava incluído nos objetivos deste trabalho era o de melhorar o desempenho principalmente de consultas, já que um sistema em tempo real deve oferecer consultas a dados e resultados também em tempo real. Neste ponto, deve ser ressaltado as otimizações feitas nos scripts das consultas a nível de aplicação. Através de testes exaustivos encontrou-se a melhor maneira de se realizar estas consultas, melhorando o desempenho dos sitema como um todo.

Em geral os objetivos propostos foram alcançados, ficando um ou outro ponto que ainda poderia ser melhorado para trabalhos futuros, visto que em um sistema do porte do SmartMine a preocupação com desempenho e a adequação do banco de dados devem ser constantes.

Capítulo 5

Conclusão

Dentre os objetivos deste trabalho que eram principalmente remodelar o banco de dados do sistema SmartMine, resolvendo problemas de desempenho e inconsistência de dados verificou-se que em sua grande maioria foram alcançados.

Após implementar as soluções encontradas no decorrer deste trabalho pode-se verificar que houve grandes melhorias no funcionamento do sistema. Erros cotidianos foram solucionados e as principais carências foram supridas. Novos conceitos de tratamento de erros e padrões de codificação foram desenvolvidos e implantados durante a remodelagem do banco de dados.

Desta forma para o estagio atual que o sistema se encontra o modelo de banco de dados é um dos mais adequados, mas ele deve estar em constante evolução para acompanhar as mudanças do próprio sistema.

Alem disso, agora que o sistema está se comportando de forma bem mais estável poderemos passar para uma segunda etapa no desenvolvimento do banco de dados, seguindo-se uma análise do volume de informações e do seu tratamento adequado. Já estamos estudando a possibilidade de implantação de técnicas de clusterização para diminuir o fluxo de dados que o sistema precisa tratar com mais freqüência.

A divisão dos dados históricos em blocos ou clusters separados fisicamente em disco é umas das vedetes das novas tecnologias de banco de dados e iremos continuar atentos para manter o sistema SmartMine atualizado com o que há de mais moderno e eficiente no mercado adequando o sistema à nossa realidade disponível.

Referências Bibliográficas

[AARHUS - 1996] University of Aarhus, Aarhus, Denmark. **Design CPN - Overview of CPN ML Syntax**, version 3.0, 1996.

[BASTOS - 2003] BASTOS, Guilherme. **Sistema de gerenciamento de mina: SmartMine.**

[BATINI - 1992] BATINI, C. CERI, S.; NAVATHE, S.B. - **Conceptual Database Design.** The Benjamin Cummings Publishing Company, Inc., 470pp., 1992.

[BESTRAVOS - 1997] BESTRAVOS, A., LIN, K. J. and SON, S.H. **Real Time Database Systems: Issues and Applications**. Kluwer Academic Publishers, 1997.

[BLAHA - 1998] BLAHA, M., PREMERLANI W. **Object-Oriented Modeling and Design for Database Applications**. Prentice Hall, 1998.

[CHANG - 1996] PERKUSICH, A., PERKUSICH, M.L.B. and CHANG, S.K. **Object oriented design, modular analysis, and faulttolerance of real-time control software systems.** International Journal of Software Engineering and Knowledge Engineering, 6(3):447-476, 1996.

[DIPIPPO - 1995] DIPIPPO, L.C. **Semantic Real-Time Object-based Concurrency Control**. PhD thesis, Department of Computer Science and Statistics, University of Rhode Island, 1995.

[ELMASRI - 1999] ELMASRI, R.; NAVATHE, S.B. - **Fundamentals of Database Systems.** Addison- Wesley Publishing Company, 3a. edição, 955 pp., 1999.

[GORDON - 1993] GORDON K. **"diswg":database management systems requirements**. Technical report, Alexandria, Virginia: NGCR SPAWAR 331 2B2, 1993.

[GUERRERO - 1997] GUERRERO, D.D.S., DE FIGUEIREDO, J.C.A. , PERKUSICH A. **Object-based high-level petri nets as a formal**

approach to distributed information systems. In Proc. of IEEE Int. Conf. on Systems Man and Cybernetics, pages 3383-3388, Orlando, USA, October 1997.

[JENSEN - 1992] JENSEN, K. **Coloured Petri Nets: Basic Concepts, Analysis, Methods and Practical Use**. EACTS - Monographs on Theoretical Computer Science. Springer-Verlag, 1992.

[KORTH - 1995] KORTH, H.F. ; SILBERSCHATZ, A. -**Sistema de Bancos de Dados.** Makron Books, 2a. edição revisada, 754 pp., 1995.

[KRISHNA - 1996] KRISHNA, C. ,SHIN, K. G. **Real-Time Systems**. MacGraw Hill Series in Computer Science, 1996.

[LIMA - 2001] LIMA, Eduardo Jorge Lapa e RAMOS,Joel. **Os desafios do Administrador de Bancos de Dados**

[NAVATHE - 1994] NAVATHE, S.B., ELMASRI, R.A. **Fundamentals of Database Systems.** 2nd Edition. Addison-Wesley Pub. Co., New York, 1994.

[OZSOYOGLU - 1995] OZSOYOGLU, G., SNODGRASS, R.T. **Temporal and real-time databases: A survey**. IEEE Transactions on Data and Knowledge Engineering, 7(4):47-56, August 1995.

[PECKHAM - 1996] PECKHAM, J., WOLFE, V.F., PRICHARD, J.J. and DIPIPPO, L.C. **Design of a real-time object-oriented database system.** Technical report, University of Rhode Island, TR94-231, 1996.

[PERKUSICH - 1997] GUERRERO, D.D.S. ,PERKUSICH, A., DE FIGUEIREDO, J.C.A. . **Modeling a cooperative environment based on an object-based modular petri net**. In Proc. of The 9th International Conference on Software Engineering and Knowledge Engineering, pages 240-247, Madrid, Spain, June 1997.

[RAMAMRITHMAN - 1993] RAMAMRITHMAN, K. **Real-time databases.** International Journal of Distributed and Parallel Databases, 1993.

[RAMAKRISHMAN - 1998] RAMAKRISHNAN, R. - **Database Management Systems.** McGraw -Hill Companies, Inc., 741pp., 1998.

[RUMBAUGH - 1991] RUMBAUGH,J., BLAHA, M., PREMERLANI, W., EDDY, F. and LORENSEN, W. **Object-oriented modeling and design.** Englewood Cliffs, N.J.: Prentice Hall, 1991.

[TEODORO - 2003] TEODORO, Julio Cesar. - **Customização De Software Intensivo - Smartmine.** 2003.

[TURNELL - 1998] PERKUSICH, M.L.B., TURNELL, M.F.Q.V. and PERKUSICH A. **Object-oriented real-time database design based on petri nets.** In Proc. of IEEE Int. Conf. on Systems Man and Cybernetics, pages 202-207, San Diego, USA, October 1998.