ESTUDO DO PARADIGMA ORIENTADO A OBJETO EM SISTEMAS DE DECISÃO E MINERAÇÃO DE DADOS EM AMBIENTES DISTRIBUÍDOS ATRAVÉS DE FERRAMENTAS COMPUTACIONAIS INTELIGENTES

RELATÓRIO FINAL DE PROJETO DE INICIAÇÃO CIENTÍFICA
(PIBIC/CNPq/INPE)

Celso André Rodrigues de Sousa (UNIFESP, Bolsista PIBIC/CNPq)
E-mail: celso.andre@unifesp.br

Dr. José Ernesto de Araújo Filho (LIT/INPE, Orientador)
E-mail: ernesto@lit.inpe.br

Julho de 2008

ESTUDO DO PARADIGMA ORIENTADO A OBJETO EM SISTEMAS DE DECISÃO E MINERAÇÃO DE DADOS EM AMBIENTES DISTRIBUÍDOS ATRAVÉS DE FERRAMENTAS COMPUTACIONAIS INTELIGENTES

RELATÓRIO FINAL DE PROJETO DE INICIAÇÃO CIENTÍFICA
(PIBIC/CNPq/INPE)

Celso André Rodrigues de Sousa (UNIFESP, Bolsista PIBIC/CNPq)
E-mail: celso.andre@unifesp.br

Dr. José Ernesto de Araújo Filho (LIT/INPE, Orientador)
E-mail: ernesto@lit.inpe.br

Julho de 2008

00.000.00(000.0)

Instituto Nacional de Pesquisas Espaciais (INPE). Laboratório de Integração de Teses (LIT).
Estudo de sistemas de decisão e mineração de dados em ambientes distribuídos através de ferramentas computacionais inteligentes.
Programa Institucional de Bolsas de Iniciação Científica PIBIC/ CNPq/ INPE – São José dos Campos: INPE, 2007. p.; (INPE-0000-TDI/00)

1. Sistemas de suporte a decisão. 2. Mineração de dados. 3. Sistema distribuído colaborativo. I. Título

RESUMO

A construção de um banco de dados autônomo, modelado a partir do padrão Entidade-Relacionamento, bem como sua interação com uma interface PHP (Hypertext Preprocessor), modelada a partir do padrão UML (Unified Modeling Language), para a integração de diversos grupos de trabalho relacionados à pesquisa, desenvolvimento e inovação em ciência e tecnologia é proposto neste trabalho. Diferente do anterior, este trabalho emprega o paradigma de orientação a objetos para o desenvolvimento do sistema. A escolha por uma modelagem orientada a objetos visa o controle do fluxo de informação, as restrições de acesso, o tratamento de exceções, a integridade das informações do banco de dados, bem como o auxilio ao reaproveitamento e manutenção dos códigos. Outra linguagem de programação utilizada em paralelo ao PHP, chamada Javascript, foi utilizada para auxiliar na comunicação do computador cliente com o servidor, visto que Javascript atua na máquina do cliente e PHP atua no servidor. Com a linguagem Javascript em conjunto ao PHP, foi possível criar um editor de texto, com as funcionalidades de formatação de texto, para a inserção de informações textuais no banco de dados. A arquitetura cliente-servidor utiliza comunicação por meio de objetos remotos para que haja integridade nos dados enviados e recebidos, fazendo com que haja a necessidade de uma linguagem de programação atuando no servidor, nesse caso o PHP, e outra atuando no cliente, nesse caso Javascript. Essa arquitetura utiliza requisições sincronizadas entre as linguagens. O sistema permite a busca de informações no banco de dados do Currículo Lattes sobre os usuários para a atualização automática e armazenamento dos dados. A técnica Ajax (Asynchronous Javascript And XML) é voltada para comunicação com as demais ferramentas por meio de solicitações assíncronas, tornando assim a interface PHP mais dinâmica e as requisições ao banco de dados mais rápidas. O Sistema de Gerenciamento de Banco de Dados (SGBD) escolhido foi o PostgreSQL, pois, além de sua facilidade e estabilidade na comunicação com as demais ferramentas, também garante maior segurança e suporta maior volume de manipulação de dados. O sistema conta com um mecanismo de segurança a fim de manter a integridade contra ataques de rede. Ferramentas computacionais inteligentes visam a mineração de dados e suporte a decisão.

SUMÁRIO

LISTA DE FIGURAS

1. INTRODUÇÃO

Um sistema íntegro e modelado a partir de padrões de Engenharia de Software é a proposta desse trabalho. A modelagem de software é a principal ferramenta para se criar sistemas mais robustos e com menos propensão a erros. Na modelagem do sistema em questão, feito em PHP (Hypertext Preprocessor), usou-se o padrão UML (Unified Modeling Language). UML dispõe de vários tipos de diagramas para representação do funcionamento do sistema. Dentre tais diagramas, utiliza-se neste trabalho os diagramas de classes, de casos de uso e de navegabilidade. Uma boa modelagem de software, além de dar uma visão geral do sistema, reduz o número de erros que o mesmo poderá gerar, o tempo ocioso de programação, aumentando a qualidade do software.

Para a confecção do sistema, optou-se por uma modelagem descentralizada. Essa técnica de modelagem chama-se independência modular do sistema. Ao dividir o sistema conforme sua funcionalidade e criar ligações entre esses módulos, os programadores ganham maior liberdade para desenvolver seus módulos, visto que eles não precisariam conhecer o sistema inteiro para poder desenvolver um pedaço do mesmo, o que aumenta a velocidade de se programar partes de um dado sistema e auxilia o reaproveitamento e manutenção dos códigos.

Para melhor desempenho de programação utiliza-se o framework PRADO, pois ele separa os códigos referentes às funcionalidades do sistema e os códigos referentes à formatação do mesmo. PRADO é compatível com XML e trata as suas tags como objetos, facilitando assim, o tratamento de eventos e as operações no banco de dados. Para se utilizar tal framework, necessita-se de um editor de texto que tenha compatibilidade com o mesmo. O editor escolhido foi o DreamWeaver, pois é o editor mais indicado para se trabalhar com este framework.

Os erros são inerentes aos sistemas de software. O que se pretende é minimizar a quantidade de erros que o sistema pode gerar. Uma das técnicas aplicadas para a redução desses erros chama-se tratamento de exceções.

Utilizam-se classes que lançam exceções ao sistema para poder simular possíveis erros que possam acarretar um mau funcionamento do mesmo. Ao se tratar essas exceções, a maioria dos erros é manipulada e eliminada antes de se entregar o sistema ao cliente, aumentando a qualidade do software.

O sistema dispõe de um controle de usuários. Cada usuário terá permissões diferentes para utilizar o sistema. Para que haja um controle efetivo dos usuários, faz-se necessária a utilização de restrições de acesso. As restrições de acesso serão feitas por meio de variáveis de seção quando o usuário entrar no sistema. Variáveis de seção têm visibilidade global e são muito úteis para se passar informações entre diferentes páginas. Quando um usuário entra no sistema, uma busca é feita no banco de dados para se obter as permissões de acesso de tal usuário. As variáveis de seção guardam os valores obtidos nessa busca e os utilizam para verificar permissões de acesso de um dado usuário enquanto ele estiver usando o sistema. Ao guardar os valores em variáveis de seção, o sistema não precisa acessar o banco de dados toda vez que houver restrição de acesso numa página, aumentando o desempenho do mesmo.

A interação com o usuário será feita através de uma interface gráfica, feita em PHP, para que as operações no banco de dados sejam feitas mais facilmente. Para confeccionar a interface com o usuário utiliza-se, em conjunto à linguagem PHP, uma técnica chamada AJAX (Assynchronous JavaScript and XML), pois ela permite que as páginas web sejam mais dinâmicas. AJAX faz uso das características da linguagem JavaScript, da portabilidade e flexibilidade do padrão XML e dos recursos de chamadas assíncronas que foram implementadas nos navegadores mais modernos. O objeto que AJAX utiliza chama-se XmlHttpRequest, que é um objeto implementado pela linguagem JavaScript. Tal objeto tem a capacidade de executar a leitura remota de dados de forma assíncrona, permitindo assim a execução de outras tarefas imediatamente após a chamada.

Após a entrada de dados feita pelo usuário através da interface gráfica, faz-se necessário uma comunicação segura com o servidor, seguindo alguns padrões pré-existentes. A comunicação cliente/servidor será feita por um protocolo de

rede conhecido como SOAP (Simple Object Access Protocol), pois tal protocolo permite o envio de objetos pela rede. Os tipos de objetos escolhidos para serem lançados pelo SOAP são os objetos remotos. Tais objetos abstraem toda a complexidade da comunicação em rede, tornando-a mais transparente ao programador. Eles são chamados de remotos, pois seus métodos podem ser invocados remotamente (de máquinas diferentes), como se fossem métodos locais. O encapsulamento de dados por parte dos objetos faz com que haja mais segurança nessa comunicação, visto que, caso alguém consiga capturar os objetos, não conseguiriam acessar seus atributos diretamente. O protocolo SOAP define uma estrutura XML de troca de mensagens que pode chamar e retornar resultados a partir de uma aplicação. Integrar esse protocolo ao sistema traz algumas vantagens ao mesmo, pois o SOAP é independente de plataforma e possui uma decodificação fácil entre os módulos do sistema, facilitando a comunicação sem deixar de lado a segurança dos dados enviados e/ou recebidos. O padrão de projeto utilizado para implementar a comunicação cliente/servidor chama-se Proxy.

Feita a comunicação cliente/servidor, o sistema deve controlar as operações feitas pelos usuários. Como o sistema permite que vários usuários estejam realizando operações simultaneamente no banco de dados, haveria problemas quanto à integridade dos dados caso, por exemplo, dois usuários estivessem alterando o mesmo arquivo. Para isso, necessita-se de um sistema com suporte à múltiplas Threads. Threads são essenciais em várias aplicações e, especificamente nesta, elas melhoram bastante a performance do sistema.

Feita a validação das informações por meio do sistema multithreading, a gravação delas faz-se necessária para uma posterior busca e/ou atualização dos dados enviados. Para resolver esse problema, utiliza-se um Sistema de Gerenciamento de Banco de Dados (SGBD), chamado PostgreSQL, para armazenar e tratar as informações referentes ao sistema. Utilizam-se classes para gerenciar a transferência de informações entre o banco de dados e o servidor. Tais classes têm como objetivo dar mais clareza aos códigos e abstrair a estrutura interna do banco de dados usando uma linguagem de alto nível. A modelagem do banco de dados é feita utilizando-se o padrão Entidade-

Relacionamento. Essa modelagem visa dar mais clareza sobre o funcionamento interno do banco de dados, bem como os relacionamentos inerentes ao mesmo.

2. MODELAGEM DO SISTEMA

2.1. Requisitos funcionais

Requisitos funcionais são requisições que o sistema deve implementar. Foram constatados vários requisitos para o sistema em questão, tais como montagem de relatórios de forma automática, interface gráfica de fácil navegação, operações de remoção e inserção de dados via interface. Um software, modelado conforme os padrões da orientação a objetos, capaz que cumprir esses requisitos é o objetivo deste trabalho. Optou-se por uma modelagem orientada a objetos, pois ela propicia melhor entendimento do funcionamento geral do sistema, auxilia o reaproveitamento e manutenção do código.

2.2. Razões de se utilizar orientação a objetos

O paradigma orientado a objetos surgiu na tentativa de solucionar problemas existentes no desenvolvimento de softwares complexos e confiáveis, com baixo custo de desenvolvimento e manutenção. Este paradigma é muito eficiente na confecção de softwares mais complexos, pois os códigos ficam agrupados conforme sua funcionalidade. Alguns conceitos inerentes à orientação a objetos devem ser esclarecidos para melhor compreensão deste trabalho. Dentre tais conceitos, utilizam-se neste trabalho os conceitos de abstração, classes, objetos, instanciação e herança.

Abstração é uma maneira de representar, em um único elemento, um grupo de características de maneira a expressar suas similaridades e suprir suas diferenças. O elemento citado refere-se a uma classe. Objeto é uma materialização de uma classe, tendo as mesmas características da classe em que se derivou. Instanciação é a maneira de se alocar objetos, ou seja, materializar uma classe. Herança refere-se a um relacionamento “é-um” entre

as classes, ou seja, uma classe A, que herda de uma classe B, possui todas as características da classe B, podendo ter outras características mais específicas.

A idéia de se utilizar orientação a objetos surge do fato de se mapear as funcionalidades do sistema para as classes. No caso do sistema em questão, haveriam classes para conectar e realizar operações no banco de dados, para transferência de informações de forma segura. Um modelo orientado a objetos abstrai todas as funcionalidades do sistema e possui vários diagramas que ajudam a sua compreensão.

2.3. Padrões de projeto

Há vários padrões de projeto para se criar softwares mais robustos. Dentre os padrões existentes, usam-se neste o trabalho os padrões Proxy, que é um modelo para a comunicação cliente/servidor, Singleton, que é um modelo para se fazer objetos com visibilidade em todos os códigos de aplicação. Utilizar padrões de projeto torna os códigos mais legíveis e mais uniformes, visto que seguem um modelo padrão. Seguir padrões de projeto é uma boa prática de programação, pois eles abstraem várias funcionalidades do sistema ao seguir modelos pré-determinados.

2.4. Diagramas UML

UML (Unified Modeling Language) é uma linguagem de modelagem para visualização, especificação, construção, documentação e comunicação de um software orientado a objetos [9, 6]. Há quatro elementos essenciais no estudo de UML: elementos estruturais, comportamentais, de agrupamento e de anotação [6]. Os únicos elementos que são mostrados nesse trabalho referem-se aos elementos estruturais, que são partes estáticas de um modelo, representando elementos que são ou conceituais ou físicos. Há vários tipos de diagramas para representar tais elementos. Dentre tais diagramas, usam-se os diagramas de classes, de casos de uso e de navegabilidade.

2.4.1. Diagramas de classes

Os diagramas de classes são os principais diagramas estruturais da UML. Eles mostram classes, interfaces e seus relacionamentos [6]. Os diagramas de classes referentes ao sistema proposto são mostrados abaixo.

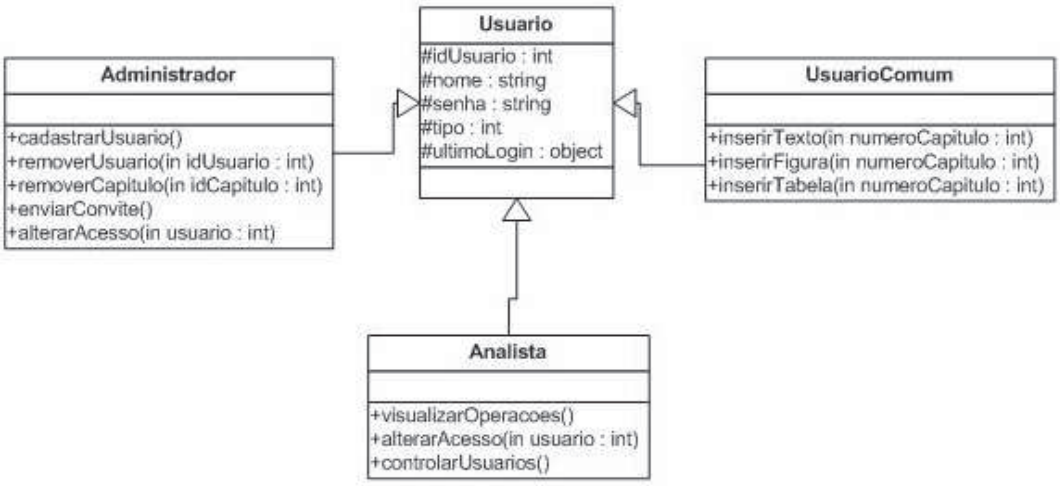

Figura 1 - Tipos de usuários do sistema

A figura 1 mostra que há três tipos de usuários no sistema – administrador, UsuarioComum e Analista. Essas três classes herdam da classe Usuario.

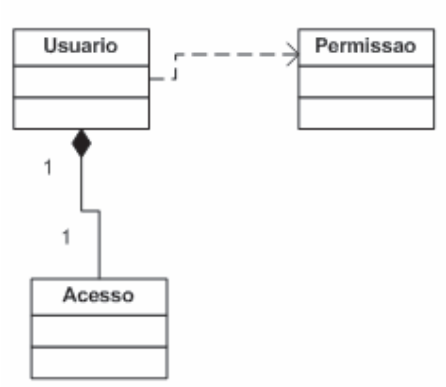

Figura 2 - Relacionamentos entre as classes Usuario, Permissao e Acesso.

Entre as classes Acesso e Usuario há um relacionamento de composição, enquanto que o relacionamento entre Permissao e Usuario é de dependência. Caso haja uma mudança na classe Permissao, a classe Usuario também irá mudar, o que indica um relacionamento de dependência entre essas duas classes.

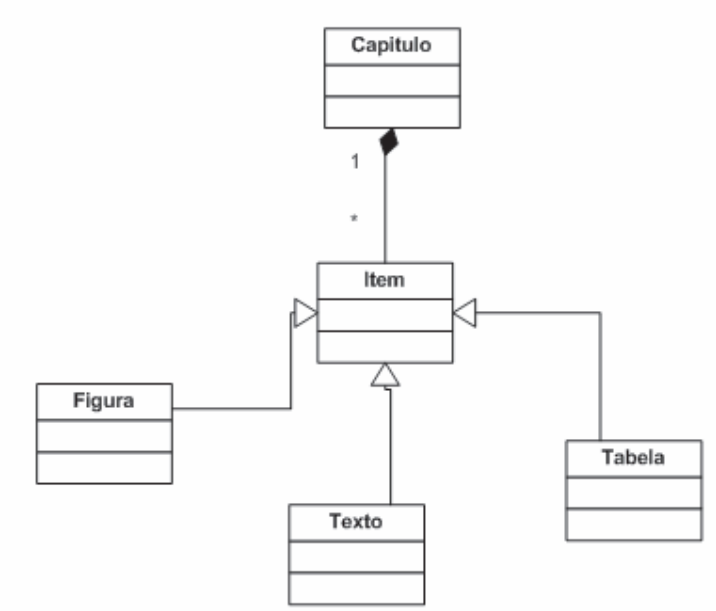

Figura 3 - Relacionamento entre as classes referentes ao relatório

A figura 3 mostra que as classes Figura, Texto e Tabela possuem um relacionamento de herança com a classe Item, enquanto que esta possui um relacionamento de composição com a classe Capitulo. Nesse caso, o relacionamento entre a classe Capitulo e a classe Item deve ser obrigatoriamente de composição, pois ao se remover um objeto da classe Capitulo, todos os objetos da classe Item devem ser removidos também.

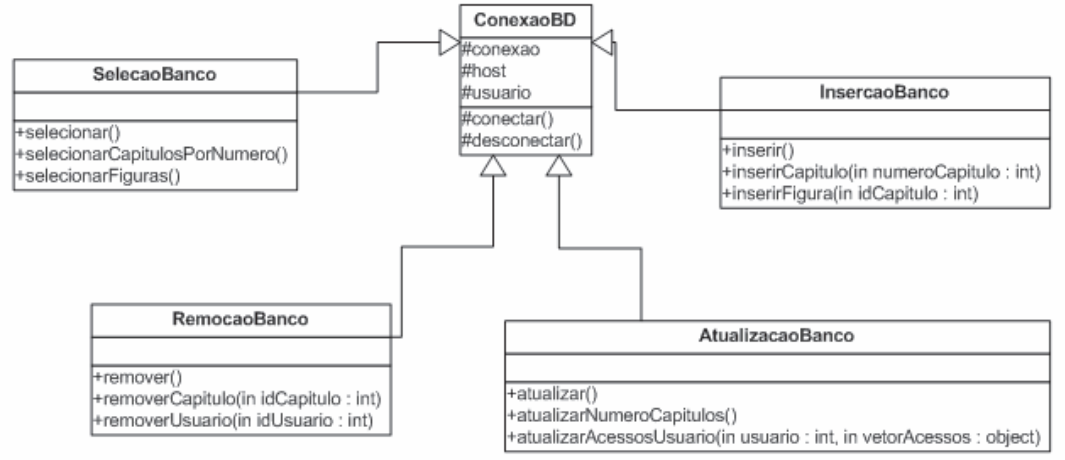

Figura 4 - Classes responsáveis pelas operações no banco de dados

A figura 4 mostra que as classes responsáveis pelas operações no banco de

dados, SelecaoBanco, InsercaoBanco, RemocaoBanco e AtualizacaoBanco, herdam da classe ConexaoBD. Tais classes são utilizadas para abstrair as estruturas internas do banco de dados ao se confeccionar o sistema em PHP. Tal idéia ajuda no reaproveitamento de código e na clareza do mesmo, visto que operações no banco de dados serão mapeadas para métodos de classes.

2.4.2. Diagramas de casos de uso

Diagramas de casos de uso são especialmente importantes na organização e modelagem das principais funcionalidades de um sistema. Um caso é a especificação de seqüências de ações atenderem a uma funcionalidade do sistema, interagindo com seus agentes [6]. Casos de uso também são úteis para mostrar as diferentes visões de um sistema. Os diagramas de casos de uso referentes ao sistema proposto são mostrados abaixo.

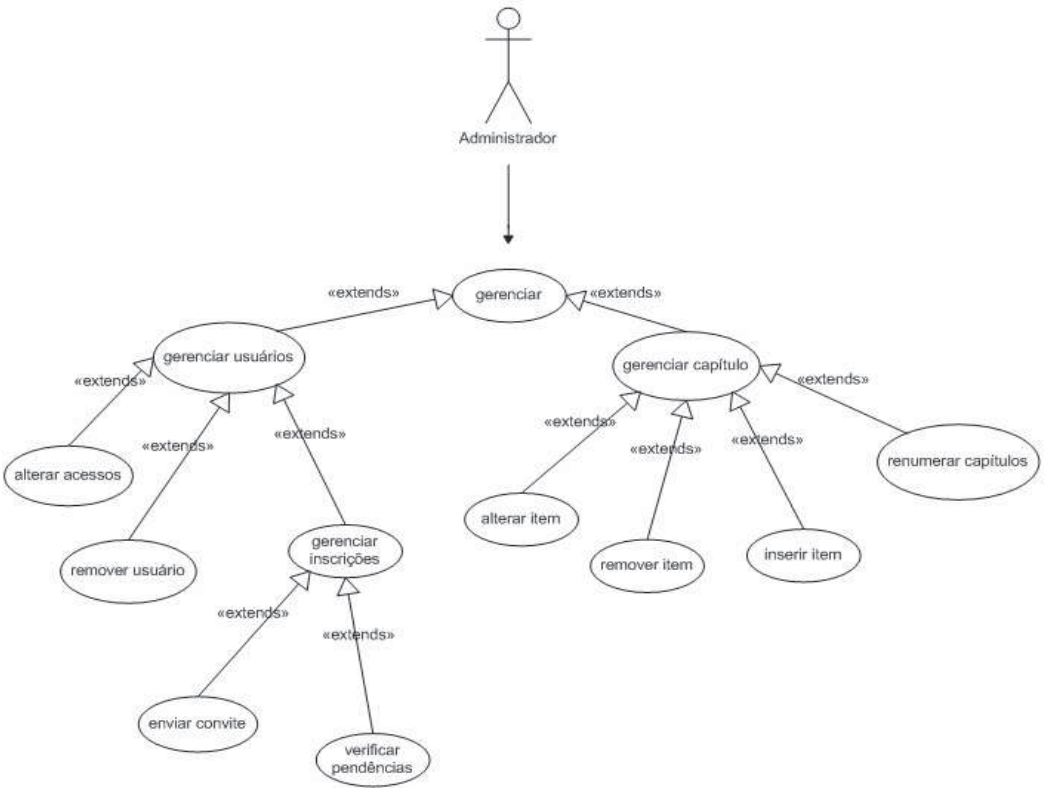

Figura 5 - Casos de uso do administrador do sistema

A figura 5 mostra as possíveis operações que o administrador do sistema pode fazer no mesmo. Notam-se vários relacionamentos de extensão neste

diagrama de casos de uso, o que indica a não obrigatoriedade da realização de uma determinada tarefa por parte do administrador. Relacionamentos de extensão (extends), referindo-se ao diagrama de casos de uso, indicam que há um condicional para a realização de uma outra tarefa. Tomando como exemplo a operação gerenciar, nota-se que há duas operações que se relacionam com ela por extensão, indicando que o administrador pode realizar qualquer uma das tarefas, dependendo de um condicional.

Outro relacionamento importante é o relacionamento de inclusão (include), que indica uma obrigatoriedade no cumprimento de outra tarefa. Tomando-se como exemplo um sistema em que há um objeto da classe Carro, inicialmente desligado, e um objeto da classe Motorista, o qual aciona o método dirigir(). Para realizar a operação de dirigir, o objeto da classe Motorista deve acionar o método ligarCarro() primeiramente, ou seja, o método dirigir() inclui (include) o método ligarCarro().

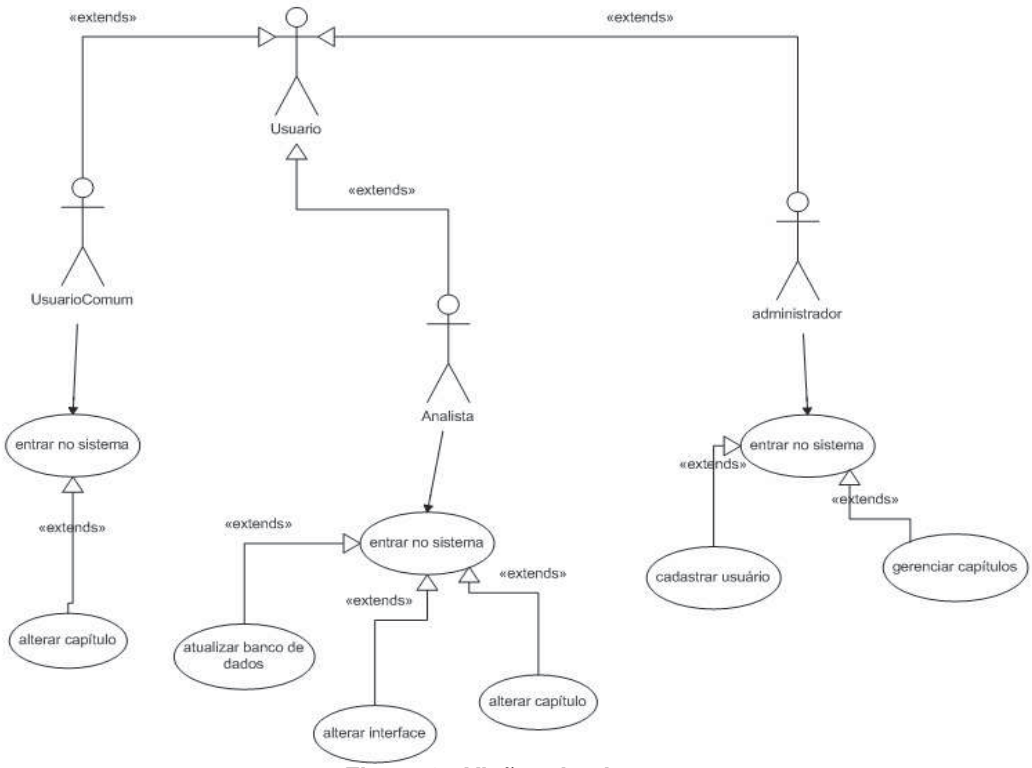

Figura 6 - Visões do sistema

A figura 6 mostra as três diferentes visões propostas para o sistema em questão. Tal diagrama infere que haja um polimorfismo para a implementação das páginas web do sistema. Tal técnica é mostrada abaixo como forma de

pseudo-código para não vincular essa técnica à uma linguagem de programação específica e tornando-a mais geral.

```
Se (usuário = admin)
        Construa a página correspondente ao administrador

Senão se (usuário = analista)
        Verifique as permissões de acesso
        Construa a página correspondente ao analista

Senão
        Verifique as permissões de acesso
        Construa a página correspondente ao usuário
```

2.4.3. Diagrama de navegabilidade

Diagramas de navegabilidade são diagramas que mostram a navegação entre as diferentes funcionalidades do sistema. Tais funcionalidades podem ser designadas tanto para programação web quanto para a programação desktop. No primeiro caso, os nós do diagrama podem representar páginas web do sistema, enquanto que no segundo caso, os nós podem representar as interfaces gráficas disponíveis para o usuário. Diagramas de navegação servem para visualizar os caminhos que podem ser seguidos pelos usuários do sistema. No caso deste trabalho, os nós do diagrama representam as diferentes funcionalidades do sistema e estas serão agrupadas em Master Pages diferentes. Tal diagrama referente ao sistema é mostrado a seguir.

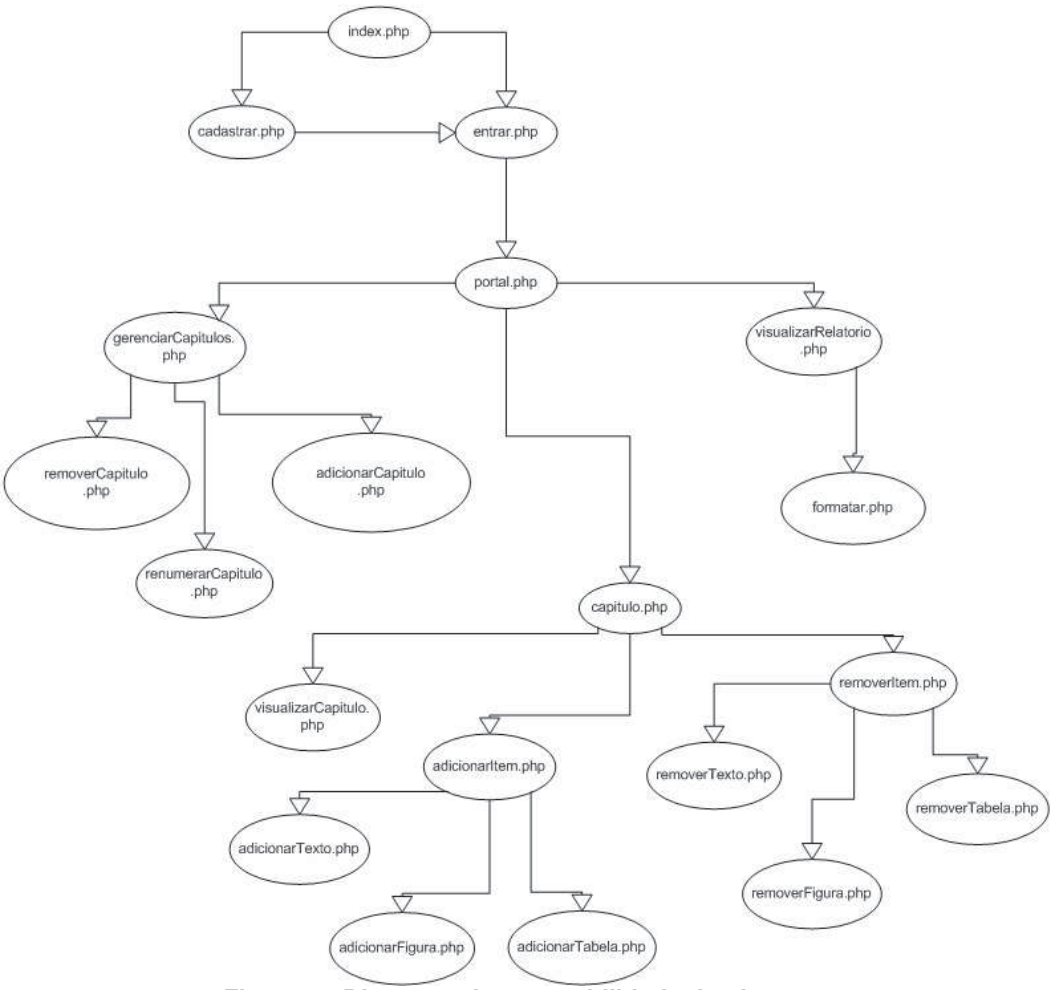

Figura 7 - Diagrama de navegabilidade do sistema

A figura 7 mostra sucintamente a navegação de um usuário pelo sistema. Os usuários começam na página index.php e, dependendo do evento que for gerado, irão para a tela de cadastro (cadastrar.php) ou de login (entrar.php). A tela de cadastro gera um erro caso já exista um usuário com o mesmo nome e/ou senha ou quando um campo não foi preenchido, enquanto que a tela de login gera erro quando não existe o usuário cadastrado e quando houve algum campo não preenchido. Quando o usuário consegue entrar no sistema, é direcionado para a página portal.php. Se o usuário for o administrador, ele pode acessar qualquer parte do sistema.

Os usuários comuns terão apenas as opções de visualizar os capítulos que tem acesso e adicionar itens aos mesmos e não terão acesso para remoção de nenhum item de nenhum dos capítulos.

2.5. Apresentação das ferramentas utilizadas

Para a confecção do sistema foram utilizadas várias ferramentas, dentre as quais cito as linguagens PHP e JavaScript, o protocolo de rede SOAP, o software para modelagem Microsoft Visio, framework PRADO, técnica de objetos remotos, técnica Ájax, técnica Master Page (utilizada no trabalho anterior).

As linguagens PHP e JavaScript atuam do lado do servidor e do cliente, respectivamente. PHP é uma linguagem muito utilizada para acessar banco de dados, construir interfaces gráficas e gerenciar o sistema de software. JavaScript é muito utilizado para melhorar a interação do usuário com as interfaces gráficas. Além disso, JavaScript dispõe de um conjunto de ferramentas poderosas para melhorar o desempenho do sistema, tais como o DOM (Document Object Model), o CSS (Cascading Style Sheets), recursos de segurança. Juntos, PHP e JavaScript, geram uma solução viável para a confecção de um sistema de software robusto e com uma boa interação com o usuário.

Utiliza-se o protocolo de rede SOAP para a comunicação cliente/servidor. Tal protocolo usa a linguagem XML como padrão para o envio de mensagens. Ele lança objetos pela rede para aumentar a segurança dos dados enviados e/ou recebidos entre o cliente e o servidor [8]. Escolheu-se este protocolo para que se pudesse utilizar objetos remotos com maior segurança e para que houvesse uma linguagem padrão de comunicação, neste caso a XML, entre os componentes do sistema.

O software Microsoft Visio foi utilizado para confeccionar os diagramas UML. Este software dispõe de todos os tipos de diagramas presentes na UML e de uma interface gráfica bem acessível, o que facilita sua utilização. Visio é um software compatível com o sistema Windows e necessita de poucos recursos para executar, o que acarretou a sua escolha como software de modelagem do sistema.

Framework PRADO é um estilo de programação em PHP. PRADO trata suas tags como objetos e é compatível com XML, que melhora a comunicação com o restante do sistema. Este framework possui diversas funcionalidades e vantagens e, por isso, será explicado com mais detalhes na seção 3.

Objetos remotos são objetos que podem ser utilizados a partir de outras máquinas como se fossem objetos locais [9]. Ao se tentar acessar um objeto de uma máquina diferente, não teríamos visibilidade sobre o mesmo e não conseguiríamos utilizar seus métodos. Para acessar tal objeto seria necessário ter um vasto conhecimento sobre redes de computadores e saber o local em que o objeto estava na outra máquina. A técnica de objetos remotos surgiu para tentar solucionar este problema. Essa técnica abstrai toda a complexidade da comunicação em rede, tornando-a transparente ao programador. Ela é bem utilizada para a confecção de sistemas complexos e para melhorar a comunicação cliente/servidor, tornando-a mais rápida e dinâmica. Tais fatos mostram a importância da utilização de objetos remotos, levando-se a crer que integrá-los ao sistema aumentaria o desempenho e a segurança do mesmo.

A técnica Ajax é uma técnica de programação em JavaScript para solucionar requisições assíncronas, aumentando o desempenho da interface gráfica, pois apenas as requisições novas é que seriam recarregadas, não necessitando a recarga completa da página atual. É uma técnica muito utilizada em aplicativos web para evitar processamento desperdiçado pelos servidores.

A técnica Master Page, utilizada no trabalho anterior, visa a construção de uma página padrão, chamada página mestre, para a confecção das funcionalidades do sistema. Faz-se um padrão para as páginas a serem exibidas e muda-se apenas um pedaço delas, dependendo da requisiçã do usuário. É uma técnica de programação em aplicativos web muito utilizada atualmente, principalmente quando as páginas web a serem criadas possuem um formato padronizado.

3. FRAMEWORK PRADO

3.1. Vantagens

PRADO é um framework para programação em PHP. Ele separa os códigos de programação dos códigos de formatação, o que aumenta o desempenho de programação e legibilidade dos mesmos. Códigos de programação referem-se às funcionalidades do sistema, tais como eventos causados pelo usuário, acesso ao banco de dados, enquanto códigos de formatação referem-se aos códigos que serão interpretados pelo browser.

O PRADO trata suas tags como objetos, o que facilita a programação e a manutenção dos códigos. Pode-se acessar um banco de dados utilizando-se, simplesmente, as tags deste framework, o que facilita a interação sistema/banco de dados.

3.2. Integração com as demais ferramentas

PRADO é compatível com XML e trata as suas tags como objetos, conforme dito na seção 3.1. A comunicação entre as ferramentas do sistema será feita através da linguagem XML, visto que o framework PRADO é compatível com ela e o protocolo de rede SOAP utiliza essa linguagem como padrão para troca de mensagens entre cliente e servidor.

PRADO tem compatibilidade com JavaSript, que é a linguagem utilizada para capturar requisições do cliente e transferi-las para o servidor. Os códigos feitos em JavaScript ficarão separados dos códigos de formatação para melhor entendimento e manutenção dos mesmos, visto que estes estarão em módulos diferentes.

4. TRATAMENTO DE EXCEÇÕES

4.1. A razão de se usar exceções

Exceções são objetos das linguagens orientadas a objetos que simulam possíveis erros gerados pelo sistema. Lançar esse tipo de objeto auxilia na manutenção do software, pois dependendo do tipo de exceção, já se sabe em

qual classe houve o problema, o que reduz em muito o tempo de procura por erros. Os objetos de exceção são criados fazendo-se um mecanismo de herança com a classe Exception, já existente nas linguagens orientadas a objetos. O código abaixo exemplifica a utilização de exceções em PHP.

```
Class ConexaoBDException extends Exception {
        public function __construct($mensagem){
              parent::__construct($mensagem);
        }
}

Class ConexaoBD{

        /* demais atributos e métodos da classe são suprimidos */
        protected $string_conexao;

        public function conectar(){
                $conectado = pg_pconnect($string_conexao);
                if(empty($conectado)){
                        throw new ConexaoBDException("Erro ao conectar!");
                }
        }
}
```

4.2. Tratamento de exceções visa à qualidade do software

O tratamento de exceções está intimamente ligado à qualidade do software final. Tomando como exemplo o código mostrado na seção 4.1, percebe-se que a classe ConexaoBD lança uma exceção quando uma conexão ao banco de dados não é bem sucedida, ou seja, quando o sistema receber a mensagem "Erro ao conectar!", o programador já saberá que houve erro na classe ConexaoBD, não precisando analisar o código inteiro. Ao se ter um software muito complexo e extenso, ficaria difícil saber em que parte do sistema houve a falha. O uso de exceções facilita a identificação do local do erro, reduzindo o

tempo de busca pelo mesmo.

5. RESTRIÇÕES DE ACESSO

5.1. Necessidade de se fazer restrições de acesso

Restrições de acesso são necessárias em vários sistemas, pois elas restringem o acesso de um determinado usuário em algumas partes dos mesmos. Ao se ter restrições de acesso, o administrador do sistema saberia quais usuários poderiam mexer em determinada funcionalidade, o que garante maior segurança para o software.

5.2. Métodos para se fazer restrições de acesso

Há vários métodos para se fazer restrições de acesso. O método a ser empregado neste sistema utiliza variáveis de seção para fazer tais restrições. Quando o usuário entra no sistema, uma busca no banco de dados deve ser feita para verificar a existência desse usuário e saber as permissões de acesso do mesmo. Os resultados dessa busca são armazenados em variáveis de seção e utilizados posteriormente para montar as interfaces PHP conforme as permissões de cada usuário, ou seja, cada usuário terá uma visão diferente do sistema, dependendo de suas permissões de acesso. Tal técnica denomina-se polimorfismo e pode ser feita facilmente em páginas web com o auxílio de variáveis de seção, visto que estas são variáveis globais, tendo visibilidade em toda a aplicação.

O polimorfismo é uma técnica muito eficiente para desenvolver aplicações web com diferentes visões de usuário e por isso foi escolhida para a confecção do sistema. O polimorfismo, quando se refere às aplicações web, refere-se à mudança da interface gráfica para cada usuário, dependendo de suas permissões de acesso.

6. INTERFACE GRÁFICA

6.1. Utilizando interfaces gráficas

A interação com o usuário será feita através de uma interface gráfica, feita em PHP, para que as operações no banco de dados sejam feitas mais facilmente. Para confeccionar a interface com o usuário utiliza-se, em conjunto à linguagem PHP, uma técnica chamada Ajax (Assynchronous JavaScript and XML), pois ela permite que as páginas web sejam mais dinâmicas.

Ajax faz uso das características da linguagem JavaScript, da portabilidade e flexibilidade do padrão XML e dos recursos de chamadas assíncronas que foram implementadas nos navegadores mais modernos [8]. O objeto que Ajax utiliza chama-se XMLHttpRequest, que é um objeto implementado pela linguagem JavaScript. Tal objeto tem a capacidade de executar a leitura remota de dados de forma assíncrona, permitindo assim a execução de outras tarefas imediatamente após a chamada [2]. Para auxiliar o uso das interfaces gráficas, dispõe-se de uma ferramenta chamada Thread, pois ela é capaz de sincronizar tais interfaces, visando uma melhor interação do sistema com o usuário.

6.2. Entrada e saída de dados

A interface com o usuário é feita como uma maneira de comunicação entre o usuário e o sistema. Tal comunicação é feita pela entrada e saída de dados, ou seja, uma requisição do cliente gerará um evento e, a partir da interpretação deste, será mostrada a interface gráfica correspondente à requisição.

O sistema dispõe de um módulo para verificação de erros causados por eventos gerados pelos usuários, como por exemplo “usuário e/ou senha incorretos”, “área restrita”. Tal módulo de verificação de erros atua em conjunto à técnica Ajax, pois não haveria necessidade de recarregar a página inteira apenas para emitir um erro, e às classes responsáveis pelo gerenciamento das operações no banco de dados, pois há a necessidade de se verificar as permissões de cada usuário logado no sistema.

6.2.1. Editor de texto

Um editor de texto, com o auxílio das linguagens PHP e JavaScript, foi introduzido no sistema para auxiliar a inserção de informações textuais. Tal editor dispõe de funcionalidades de formatação de texto para melhorar a qualidade das informações inseridas. A confecção de um editor visava uma melhor experiência dos usuários com o sistema. A figura a seguir mostra o editor de texto e suas funcionalidades.

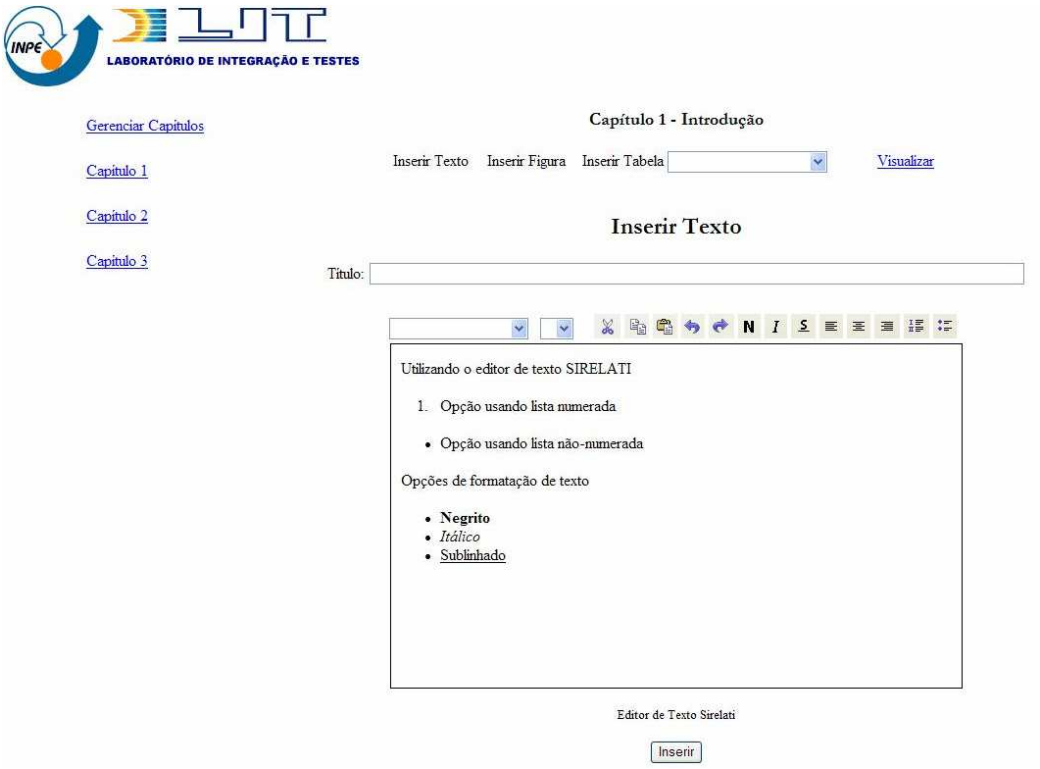

Figura 8 - Editor de texto

7. ARQUITETURA CLIENTE/SERVIDOR

7.1. Conceituação

A arquitetura cliente/servidor é a mais utilizada atualmente, devido ao fato de ser a mais avançada e a que melhor evoluiu nestes últimos anos. Esta arquitetura se divide em duas partes, o cliente e o servidor. Para o total funcionamento dessa arquitetura, faz-se necessária a utilização de três tipos de software:

- Software de gerenciamento de dados: Este software se encarrega da

manipulação e gerenciamento de dados armazenados e requeridos pelas diferentes aplicações. Normalmente este software se hospeda no servidor. No caso do sistema em desenvolvimento, este software refere-se às classes encarregadas pelo gerenciamento dos dados enviados e/ou recebidos pelo servidor.

- Software de desenvolvimento: este tipo de software se hospeda nos clientes e só naqueles que se dediquem ao desenvolvimento de aplicações. No caso do sistema em questão, este software refere-se aos códigos escritos em JavaScript para capturar as requisições dos clientes e transferi-las para o servidor para que este possa processa-las.

- Software de interação com os usuários: Também reside nos clientes e é a aplicação gráfica de usuário para a manipulação de dados, sempre é claro, em nível de usuário (consultas principalmente). Este software refere-se às interfaces gráficas do sistema, que serão responsáveis pela entrada e saída de dados de uma forma mais simples para o usuário.

7.2. Objetos remotos

Os objetos remotos são tipos de objetos que podem ser utilizados entre máquinas diferentes como se fossem objetos locais. Tais objetos abstraem toda a complexidade da comunicação em rede, tornando-a transparente ao programador [9]. Tais objetos tornam a comunicação cliente/servidor mais segura, pois há encapsulamento dos dados enviados. Para o envio destes objetos pela rede, dispõe-se de um protocolo de rede chamado SOAP, que será mostrado na próxima seção.

7.3. Protocolo de rede SOAP

O protocolo de rede SOAP (Simple Object Access Protocol) serve para enviar objetos pela rede, tornando a comunicação mais segura. Esse protocolo utiliza a linguagem XML como padrão para a troca de mensagens. Integrar esse protocolo ao sistema traz algumas vantagens ao mesmo, pois o SOAP é

independente de plataforma e possui uma decodificação fácil entre os módulos do sistema [8], facilitando a comunicação, sem deixar de lado a segurança dos dados enviados e/ou recebidos. O padrão de projeto utilizado para implementar a comunicação cliente/servidor chama-se Proxy, conforme citado na seção 2.3.

8. SISTEMA MULTITHREADING

8.1. Introdução

Thread pode ser conceituada como linha de execução [9]. A utilização de Threads surge do fato de o sistema permitir acesso múltiplo a uma mesma operação. Um sistema formado por múltiplas threads (chamado sistema multithreading) visa sincronizar essas operações para que não haja corrompimento dos dados. Por exemplo, caso dois usuários estejam alterando o mesmo arquivo e o primeiro usuário salva as informações antes do segundo usuário, ocorrerá uma concorrência dos dados, pois haverá duas atualizações do mesmo arquivo. Para tentar resolver esse problema, usam-se múltiplas threads, pois elas surgiram para resolver o problema de sincronismo e concorrência dos dados.

8.2. Necessidade em se utilizar Threads

Threads são muito úteis em várias aplicações, bem como nesta. Threads serão utilizadas para resolver o problema de concorrência dos dados e para a melhor interação da interface gráfica com o usuário.

Tomando o exemplo citado na seção 8.1, nota-se a necessidade de se utilizar threads para resolver o problema de concorrência. Uma maneira pensada para implementar threads no sistema atual seria o seguinte: quando um ou mais usuários acessam o mesmo arquivo para alterá-lo, uma thread será disponibilizada para cada usuário, a fim de que cada thread possa sincronizar as atualizações feitas pelos mesmos. Feitas as atualizações de cada usuário, as threads se encarregariam de enviá-las para o administrador do sistema, a fim de que este tenha acesso a todas as atualizações feitas e possa escolher

qual delas (ou fazer uma miscelânea dessas alterações) enviar para o banco de dados. Com essa situação, percebe-se que a utilização de um sistema multithreading visa garantir a integridade dos dados no banco de dados.

Outra grande utilidade para sistemas multithreading está na área de interfaces gráficas. Ao se utilizar, por exemplo, applets Java para a confecção de uma interface mais interativa, necessita-se de várias threads para sincronizar os applets numa página web, bem como aumentar a interação usuário/interface gráfica. O aumento dessa interação aumenta a satisfação do cliente frente ao software.

9. INTERAÇÃO COM O BANCO DE DADOS

9.1. Modelagem do banco de dados

Um banco de dados íntegro e com menos propensão a falhas é uma das idéias a serem materializadas neste trabalho. Para tanto, precisa-se de uma modelagem bem concisa e de fácil remanejamento, caso se encontre a necessidade de se incluir uma nova estrutura. A modelagem é feita seguindo o padrão Entidade-Relacionamento, visto que é o mais aconselhável quando se trata de banco de dados.

9.1.1. Padrão Entidade-Relacionamento

Entidade-Relacionamento é o padrão de modelagem mais aconselhável quando se trata de banco de dados. Tal padrão visa mostrar as estruturas internas do banco de dados, bem como o relacionamento entre cada uma delas [4]. Para tratar relacionamentos mais complexos, como por exemplo, relacionamentos com cardinalidade M:N, precisa-se de estruturas de associação para fazer o devido relacionamento. Mudanças serão feitas quanto à modelagem do banco de dados, visto que novas exigências referentes à consultas mais específicas foram acrescentadas. Estruturas de associação são utilizadas em relacionamentos mais complexos e para otimizar o tempo de busca ao se ter um banco de dados mais extenso. Classes de associação

também auxiliam a mineração de dados, que será mostrada na seção 10. A modelagem do banco de dados usando o padrão Entidade-Relacionamento foi feito no trabalho anterior.

9.2. Abstração das estruturas internas do banco de dados

A abstração da estrutura interna do banco de dados é feita a partir de classes PHP que mapeiam instruções SQL, de baixo nível, para instruções de alto nível, fazendo com que haja independência entre as estruturas do banco de dados e o sistema. Por exemplo, há uma classe no sistema chamada SelecaoBanco, que herda da classe ConexaoBD. SelecaoBanco é responsável pelas operações de SELECT no banco de dados. Um de seus métodos é selecionarCapitulosPorNumero(), que intuitivamente já indica que o resultado dessa operação irá mostrar os capítulos ordenados pelos seus números, mesmo que o programador que estiver confeccionando o sistema não tenha conhecimento sobre as estruturas internas do banco de dados. Tal estratégia denomina-se abstração das estruturas internas de um banco de dados e faz parte da idéia de independência modular do sistema, o que melhora o desempenho de programação, reduz os tempos de adaptação de um programador ao se deparar com um sistema pronto e o tempo despendido para manutenção dos códigos.

9.3. Interação com o sistema por meio de objetos

A interação do banco de dados com o sistema não é feita de forma direta, mas por meio de objetos para que haja o tratamento das informações antes que estas sejam passadas ao banco de dados. Há classes, escritas em PHP, para o gerenciamento das operações do banco de dados, tais como SelecaoBanco, que gerencia as operações de SELECT, RemocaoBanco, que gerencia as operações de DELETE, InsercaoBanco, que gerencia as operações de INSERT e AtualizacaoBanco, que gerencia as operações de UPDATE. Objetos dessas classes farão as operações no banco de dados usando métodos de alto nível como o método selecionarCapitulosPorNumero() da classe SelecaoBanco, conforme citado na seção 9.2. Caso haja concorrência de informações, elas

serão tratadas pelas threads do sistema, conforme mostrado na seção 8.2. Essas threads atuarão entre os objetos de gerenciamento do banco de dados e o próprio banco de dados para que haja integridade dos dados a serem inseridos no mesmo.

10. MINERAÇÃO DE DADOS

A mineração de dados visa tratar as informações disponíveis no banco de dados de forma inteligente. A automatização dos processos de análise de dados, com a utilização de softwares ligados diretamente à massa de informações tornou-se uma necessidade, já que o aproveitamento das informações já existentes, transformando-as em conhecimento, permite avanços sem paralelo na história do desenvolvimento dos bancos de dados.

Há várias técnicas para se fazer mineração de dados, dentre as quais cito a regressão e a de agrupamento (clustering). A técnica de regressão visa prever o futuro baseado no passado [10], ou seja, a partir de uma análise em uma massa de dados atual, pretende-se chegar a uma conclusão sobre a distribuição dos dados, tendências de convergência dos mesmos. A técnica de clusterização agrupa informações homogêneas de grupos heterogêneos entre os demais e aponta o item que melhor representa cada grupo, permitindo dessa forma, que consigamos perceber a característica de cada grupo [10].

Para auxiliar a mineração de dados, faz-se necessária a criação de estruturas de associação no banco de dados. Tais estruturas têm como finalidade reduzir o tempo de busca numa massa de dados, bem como dispor informações com maior grau de similaridade. A implementação dessas estruturas de associação está em andamento.

Pretende-se, futuramente, integrar a lógica nebulosa (fuzzy) como um método para auxiliar a mineração de dados. Algoritmos envolvendo lógica fuzzy para mineração de dados já existem, tais como o algoritmo Wang-Mendel [10], que

foi primeiramente concebido para aplicação em tarefas de Previsão de Séries Temporais.

11. SISTEMA DE SUPORTE À DECISÃO

Feita a mineração de dados, visa-se a utilização de um sistema de suporte à decisão, o qual decidirá qual dos dados obtidos como resultado da mineração é o mais próximo do resultado esperado. Tal sistema visa auxiliar os usuários quanto à decisão sobre os dados dispostos no banco de dados, segundo um grau de similaridade.

Um sistema de suporte à decisão pode ser considerado como um sistema computacional que fornece ao usuário as informações já estruturadas de maneira a facilitar e direcionar a sua decisão sobre o que é mais relevante para ele.

Para implementar esse tipo de sistema, uma técnica da inteligência computacional conhecida como lógica fuzzy faz-se necessária. Fuzzy tenta simular o cérebro humano através de mecanismos de inferência com auxílio de um ferramental matemático capaz de tratar incertezas e imprecisões. Com o auxílio da lógica Fuzzy, pretende-se criar uma função que indique o grau de similaridade entre a resposta adquirida e a resposta esperada. Tal grau de similaridade deve ser o máximo possível para que o sistema ganhe maior confiabilidade.

12. CONCLUSÃO

Dada a arquitetura e as técnicas mostradas neste trabalho, acredita-se que seja possível a criação de um sistema íntegro, com menos propensão a erros, de fácil manutenção e com interfaces gráficas mais acessíveis aos usuários. Houve resultados positivos no trabalho anterior referentes à interface gráfica e ao desenvolvimento do banco de dados, o que ressalva a possibilidade de se

confeccionar esse sistema conforme as ferramentas apresentadas.

A etapa de modelagem do sistema encontra-se em seu estágio final e aparenta ser um modelo apropriado para a confecção de um sistema de software robusto e de alta qualidade. Várias ferramentas e técnicas foram mostradas no decorrer deste trabalho e pretende-se, futuramente, integrá-las ao sistema da forma em que foi exposta. A modelagem é a etapa principal na criação de um sistema de software robusto que atenda as necessidades exigidas e que tenha maior flexibilidade em futuras alterações.

A mineração de dados e o sistema de suporte à decisão ainda estão muito incipientes, mas pretende-se melhorá-los continuamente a fim de se obter um sistema mais completo possível com certo grau de inteligência.

REFERÊNCIA BIBLIOGRÁFICA

[1] GUTMANS, ANDI; BAKKEN, STIGS; RETHANS, DERICK. PHP 5: programação poderosa. Alta Books, 2005.

[2] GONÇALVES, EDSON. AJAX na prática. Rio de Janeiro: Ciência Moderna, 2007.

[3] FLANAGAN, DAVID. JavaScript : o guia definitivo. 4ª ed. Porto Alegre: Bookman, 2004.

[4] ELMARSRI, R. E.; NAVATHE, S. V. Sistemas de Bancos de Dados: Fundamentos e Aplicações. 3ª ed. Rio de Janeiro: LTC, 2002.

[5] SILBERSCHATZ, ABRAHAM; KORTH, HENRY; SUDARSHAN, S. Sistema de Banco de Dados. 3ª ed. São Paulo: Pearson Makron Books, 1999.

[6] JACOBSON, IVAR; RUMBAUGH, JAMES; BOOCH, GRADY. UML: guia

prático. 2ª ed. Rio de Janeiro: Elsevier, 2005.

[7] RAMALHO, JOSÉ ANTÔNIO. HTML Avançado. São Paulo: Makron Books, 1997.

[8] LOURENÇO, JOSÉ LUÍS. WebServices e Ajax. Slides do Curso Students to Business. Disponível em <www.zezons.com/s2b>. Acesso em 20 jun 2008. São José dos Campos, 2008.

[9] SILVEIRA, FÁBIO FAGUNDES. UML. Slides do curso de programação orientada a objetos II. UNIFESP. Ciência da Computação. São José dos Campos. Disponível em <ead.unifesp.br/graduação>. Acesso em 07 jul 2008.

[10] GOLDSCHMIDT, RONALD; PASSOS, EMANUEL. Data Mining: um guia prático. 3ª ed. São Paulo, 2005.

[11] PRESSMAN, ROGER S. Engenharia de Software. 6ª ed. São Paulo, 2006.

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programas de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. Aceitam-se tanto programas fonte quanto os executáveis.