MICHEL-ANGE RITHCHER JEAN-BAPTISTE JUNIOR

DESENVOLVIMENTO DE UM APLICATIVO PARA GESTÃO DE EVENTOS UTILIZANDO A TECNOLOGIA J2SE

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

Orientador

Prof. Reginaldo Ferreira de Souza

LAVRAS
MINAS GERAIS - BRASIL
2003

MICHEL-ANGE RITHCHER JEAN-BAPTISTE JUNIOR

DESENVOLVIMENTO DE UM APLICATIVO PARA GESTÃO DE EVENTOS UTILIZANDO A TECNOLOGIA J2SE

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

APROVADA em __ de _____ de _____.

Prof. _____

Prof. _____

Prof. _____
UFLA
(Orientador)

LAVRAS
MINAS GERAIS - BRASIL

DEDICATÓRIA

Dedico aos meus pais Rithcher Jean-Baptiste e Anne-Marie, pela oportunidade e confiança, aos meus irmãos Daphné, Medly, e Wilkins pelo apoio dado em todos os momentos, a Roberta que estive do meu lado nas horas mais difíceis e a Harvel que sempre acreditou em mim.

AGRADECIMENTO

Agradeço a meu orientador Reginaldo pela sua competência, meu co-orientador Heitor pela sua dedicação, a minha irmã pelo apoio que ela nunca deixou de me dar, e a todos os meus amigos principalmente o Luiz e o Marcelo que fazem parte dessa conquista.

A informática não pára de crescer e, quanto mais o tempo passa, mais indispensável ela está se tornando. Hoje dia, é difícil imaginar nossas vidas sem ela e nenhuma área escapa a essa realidade. È preciso implementar cada vez mais programas de computadores. E esses cada vez mais complexos e com o objetivo de oferecer a cada área uma ferramenta poderosa e indispensável que possa agilizar e simplificar as suas rotinas de trabalho.

O presente trabalho descreve o inicio de uma implementação de uma aplicação, um software de gestão de eventos de caráter comercial usando a tecnologia J2SE. Todas as técnicas apresentadas aqui foram tiradas das teorias de Sistemas de Informações, Engenharia de Software e Interface Homem-Máquina.

Palavras-chave: modelagem OOHDM, gestão de eventos.

The computer science doesn't stop and, as the time goes by, more indispensable she is becoming. Nowadays, it is difficult to imagine our lives without it and no area escapes to that reality. It is necessary to implement computer programs much more powerfull and much more complex with the objective of offering to each area a powerful and indispensable tool that can activate and simplify their work routines.

This work describes the implementation of a software, a software of administration of events of commercial character using the J2SE technology . All the techniques presented here were removed from the theories of Information Systems, Software Engineering and Man-Machine Interface.

SUMÁRIO

INTRODUÇÃO

Vive-se em uma época de rápidas inovações tecnológicas, principalmente nos meios de comunicação. O sucesso de uma empresa hoje depende, além de outras coisas, da maneira de gerenciar as informações com as quais se trabalha no dia-dia. As empresas hoje têm certeza de que "informação" é a palavra chave deste século e que ela deve ser tratada com prioridade e com muitos cuidados.

Deve ter a preocupação de como tratar a informação, de tal maneira que ela chegue com facilidade de entendimento a quem a acessa. Desta forma, quando se vai disponibilizar algum tipo de informação, esta deve ser clara, objetiva e de rápida acessibilidade.

As empresas (o mercado) hoje participam de um mundo capitalista e globalizado cuja informação é a arma para se sobressair no mercado mundial em relação às suas concorrentes. Ter os melhores equipamentos do momento não basta mais, é preciso ter as ferramentas adequadas para lidar com um volume de informação geralmente grande. Tentar organizar essas informações no melhor computador do mundo não significa garantir uma informação clara ao usuário. É preciso uma ferramenta que disponibilize a informação de acordo com as necessidades da empresa.

Vários aplicativos que existem no mercado têm como objetivo facilitar o trabalho empresarial e diminuir a burocracia trazendo, assim, grandes melhorias nos sistemas. Esses sistemas de software são projetados para solucionar melhor um problema específico de uma dada empresa.

Mesmo com todas essas inovações tecnológicas, alguns serviços que poderiam ficar ainda mais fáceis, mais eficiente e mais prático, ainda estão sendo feitos de maneira arcaica e com muitas redundâncias em vários aspectos. É o caso dos eventos que sempre acontecem em datas predefinidas todos os anos, mas que mesmo assim, geram uma quantidade exorbitante de trabalho braçal. Esse é um dos múltiplos casos onde a informação não está sendo gerenciada com um cuidado especial e, o mais importante, não está sendo utilizada de maneira inteligente. Dados que

poderiam ser re-aproveitados acabam sempre em lugar inapropriado ou, pior, a maior parte das vezes eles nem chegam a ser armazenados.

No caso dos eventos, depois de muitas pesquisas, não se achou produtos de software no mercado que se pode usar para ter um gerenciamento mais dinâmico, rápido e eficiente. Cada evento é realizado separadamente e toda vez que ele está sendo reorganizado, todo o trabalho já realizado anteriormente tem que ser feito de novo. Isso se deve ao fato de que tudo é feito na mão. Do lado do público alvo, nenhuma inovação. Tudo que precisa, tem que ser obtido com um funcionário. Os certificados ainda são emitidos na mão sem dar aos participantes o poder de pegá-los por conta própria, sem nem permitir ao público alvo ter acesso a certas consultas básicas sobre o evento.

Como foi visto, organizar a informação é sinônimo de objetividade e lucro. A fim de automatizar o processo de cadastramento, manutenção e controle total sobre eventos sendo realizados, este trabalho teve por objetivo criar um Sistema de Gestão de Eventos, denominado SGE, sujeito a ser usado por empresas organizadores de eventos. O SGE permite cadastrar, consultar, alterar informações relativas a diversos eventos ou outras requisições que possam ajudar no gerenciamento dos mesmos.

O SGE apresenta uma interface amigável, segura e altamente configurável, permitindo a qualquer usuário ter uma interação intuitiva com o sistema. Bem como uma visualização dessas informações; proporcionando total transparência nas atividades realizadas na empresa.

Trata-se de um projeto-piloto, desenvolvido para a incubadora Berin Eventos com a possibilidade de ser vendido para outras empresas que trabalham nesse ramo.

Sobre este o SGE foram aplicadas técnicas tiradas de algumas disciplinas tais como: Sistemas de Informações, Engenharia de Software, Interface Homem-Máquina.

Analisando a carência deste software no mercado, a empresa Berin Eventos se dispôs a construir um sistema que atendesse melhor a tais necessidades.

REFERENCIAL TEÓRICO

2- Conceitos básicos

Com o objetivo de facilitar o entendimento desse trabalho, optou-se por apresentar alguns conceitos principais e fundamentais. È importante ressaltar que toda vez que a palavra "evento" é mencionado no texto ele deve ser interpretado como um acontecimento de qualquer tipo.

2.1 Sistemas de informação para gestão de eventos

Sistemas de gestão de eventos são sistemas que minimizam o tempo gasto para fazer operações administrativas. E que, se fossem realizadas de maneira convencional, tomariam um tempo considerável para serem realizadas. Eles permitem a concentração do fluxo de informações de uma forma rápida, segura e totalmente dinâmica. Eles fornecem aos organizadores uma visão das atividades regulares da empresa, de modo que possam controlar, organizar e planejar, de maneira eficaz e eficiente, as atividades e eventos.

Os sistemas de gestão de eventos proporcionam uma grande melhora no tempo de resposta das transações
e um aumento de produtividade notável.

Existem no mercado alguns produtos de software que foram desenvolvidos para responder às necessidades da Gestão Empresarial. No entanto, como esses produtos de software são desenvolvidos para uma entidade mais ampla, só existem funções ou agendas integradas que tentam reservam uma pequena seção para realizar tarefas reais de um sistema de gestão de eventos, ou, pior, para agendar datas importantes tais como reuniões, festas ou qualquer outro compromisso.

Outras ferramentas são projetadas especialmente para cuidar da gestão dos eventos nas empresas. Estes têm como objetivo dar aos organizadores um maior controle sobre os eventos sendo coordenados e não agem como um programa "secretária" ou como uma agenda que serviria para lembrar os fatos ou dados importantes na hora certa. São

sistemas dedicados que são otimizados para diminuir ao máximo o trabalho repetitivo e cansativo e dar um controle absoluto aos gerentes.

2.2 Interface homem-máquina

A interface no desenvolvimento de um software é um dos pontos que exige mais cautela e muita preocupação. Os tópicos a seguir têm como objetivo mostrar porque a interface é tão importante.

Objetivos da interface com o usuário

De maneira geral, uma interface homem-máquina compreende os comportamentos do usuário e as características e facilidades do sistema, do equipamento e do ambiente. Segundo [1], a interface não é só o que se vê na tela, mas também os periféricos, os manuais, o local de trabalho, materiais impressos e até o suporte técnico e de treinamento oferecido pelo fabricante.

No entanto, para os usuários, a interface é o sistema (o *software*). Pode se observar que os primeiros computadores e produtos de *software* foram projetados por engenheiros para serem usados por especialistas. Estes especialistas não formam o grupo representativo da atual comunidade de usuários de sistemas interativos. O uso de computadores pela sociedade cresceu muito e em alguns casos esses são invisíveis para o usuário. Em outros, pessoas não especializadas em computação estão usando diretamente equipamentos e periféricos dos mais variados tipos. Isto torna as interfaces homem-computador um elemento relevante na composição de um sistema [1].

O desenvolvimento de um Sistema de Informação Gerencial requer interfaces de fácil uso e transparentes ao usuário

Segundo [2], se são garantidos funcionalidade, confiabilidade e padronização e o gerenciamento dentro do orçamento está realizado, então o desenvolvedor deve se fixar no *design* e teste do sistema. Várias alternativas de *design* devem ser avaliadas para comunidades específicas de usuários e para tarefas diferentes. Um bom *design* para uma classe de tarefa pode ser ineficiente para outra classe. Para cada usuário e para cada

tarefa, objetivos precisos devem guiar o *design*, o avaliador, o computador e o gerente. Segundo [2], os cinco fatores humanos imprescindíveis em uma avaliação de interface com o usuário são:

• tempo de aprendizado: qual o tempo para uma comunidade típica de usuários aprender como utilizar os comandos principais e o conjunto de tarefas?

• performance de velocidade: quanto tempo leva o sistema para apresentar respostas a determinado conjunto de tarefas?

• taxa de erros dos usuários: quantas e quais são os tipos de erro que as pessoas cometem quando executam suas tarefas? O tempo para refazer e corrigir erros deve ser incorporado à performance de velocidade, erros de utilização são componentes críticos de uso do sistema que merecem estudo detalhado.

• retenção com o tempo: quanto tempo levam os usuários para reter (manter) o conhecimento sobre o sistema (uma hora, um dia, uma semana?). Retenção está associada com tempo de aprendizado e freqüência de uso do sistema;

• satisfação subjetiva: o quanto os usuários gostam de utilizar vários aspectos do sistema? As respostas podem ser colhidas através de entrevistas ou questionários impressos que incluam escalas de satisfação e espaços para comentários livre.

2.3 Técnicas de desenvolvimento de software

Para realizar qualquer tarefa, é muito importante ter um plano e seguí-lo para não se perder e não se afastar dos objetivos. Os processos de desenvolvimento de software não escapam a essa realidade. É preciso seguir determinados modelos para se ter um produto final de alta qualidade. Os tópicos a seguir apresentam os principais modelos.

2.3.1 A história do software

Nos anos 90, a computação começou a investir mais nas soluções implementadas em software. No início da computação os problemas que os programadores enfrentavam eram muito mais simples. Ninguém ainda podia prever que os produtos de software iam cuidar de complexas tarefas tipo um sistema para controle de vôos ou então um sistema para controle de metro [3].

Sentiu-se então por volta dessa época (1950), a necessidade de ter linguagens de mais altos níveis para melhorar as soluções implementadas e para diminuir os riscos de erros. Linguagens de alto nível começavam a ser inventadas para facilitar a comunicação com computadores no início dos anos 50.

Segundo [4], no início da década de 1970, a multiprogramação e os sistemas multiusuários foram a base para a implementação de aplicações muito mais sofisticadas; como sistemas de tempo real controlando grandes quantidades de dados em milisegundos e sistemas de gerenciamento de banco de dados com suporte a transações *on-line*. No final dos anos 70, surgiram também novas empresas da área de informática: os *software house*, que se concentravam na produção de software em escala.

Por volta dos anos 80, sistemas distribuídos ampliaram a complexidade dos sistemas de computação. Nessa época, o microprocessador permitiu o lançamento de *produtos inteligentes*: do automóvel aos fornos de microondas, de robôs no chão de fábrica a equipamentos hospitalares. O software estava sendo integrado ao hardware.

Em 1990, o conceito de redes de computadores se tornou popular e começava o sucesso do computador pessoal. Em 1994 algumas pessoas falavam em Internet. E em 1999, centenas de milhões de pessoas navegavam na Web diariamente. Esse fenômeno social implicou no crescimento, em escala exponencial, da demanda mundial por software.

Em três décadas houve uma inversão radical na economia da informática: o software se tornou o ponto principal no projeto de sistemas

de computação. Tanto que, atualmente, o software é a chave para sistemas de computação e, mais ainda, é um fator de diferenciação para uma empresa em relação às suas concorrentes.

2.3.2 Paradigmas da engenharia de software

De acordo com [5], os métodos de engenharia de software proporcionam os detalhes de "como fazer" para construir um software. Dentro desses métodos são incluídos o planejamento e a estimativa do projeto, a análise de requisitos de software e de sistemas, o projeto da estrutura de dados, a arquitetura de programa e o algoritmo de processamento, codificação, teste e manutenção.

2.3.2.1 O ciclo de vida clássico

O ciclo de vida clássico ou modelo cascata é o modelo mais antigo e o mais amplamente usado em engenharia de software. Ele é derivado de modelos existentes de outras engenharias. Ele fornece uma modelagem para o processo de desenvolvimento de software. A sua estrutura é composta de várias fases que são executadas de forma sistemática e seqüencial. A idéia básica do ciclo de vida clássico é terminar com cada uma das fases antes de passar para a fase seguinte, ou seja, não é possível passar à fase seguinte sem terminar a fase corrente. A figura 1 ilustra o ciclo de vida clássico.

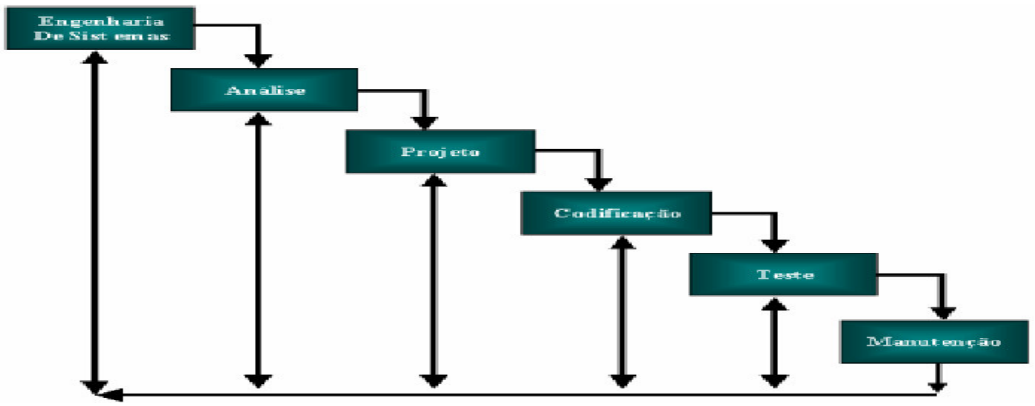

Figura 1- Modelo Clássico

Segundo [6], na prática, existe uma interação entre as fases e cada fase pode levar a modificações nas fases anteriores.

As fases mais comuns do modelo de vida clássico são:

- Análise e engenharia de sistemas;
- Análise de requisitos de software;
- Projeto (design).
- Implementação ou codificação.
- Testes
- Manutenção

2.3.2.2 Prototipação

Na prototipação, o é preciso entender os objetivos do sistema. Ela começa com requisitos vagamente entendidos e sem se importar muito com os detalhes[5].

A primeira fase prevê o desenvolvimento de um programa para o usuário experimentar. No entanto, o objetivo é estabelecer os requisitos do sistema. Na prototipação, a idéia de deixar os usuários interagir com protótipos, ainda em fase de desenvolvimento, é muito atrativa para sistemas grandes e complicados. A figura 2 ilustra o modelo de prototipação.

Figura 2- Prototipação

2.3.2.3 O modelo espiral

O Modelo espiral foi criado visando abranger as melhores características do modelo clássico e da prototipagem. De certa maneira, este modelo é semelhante à programação exploratória. A cada ciclo da espiral, versões progressivamente mais completas do software são construídas e, antes de cada ciclo, uma análise de riscos é feita. Ao fim de cada ciclo, é feita uma avaliação e então deve prosseguir para o próximo ciclo [5].

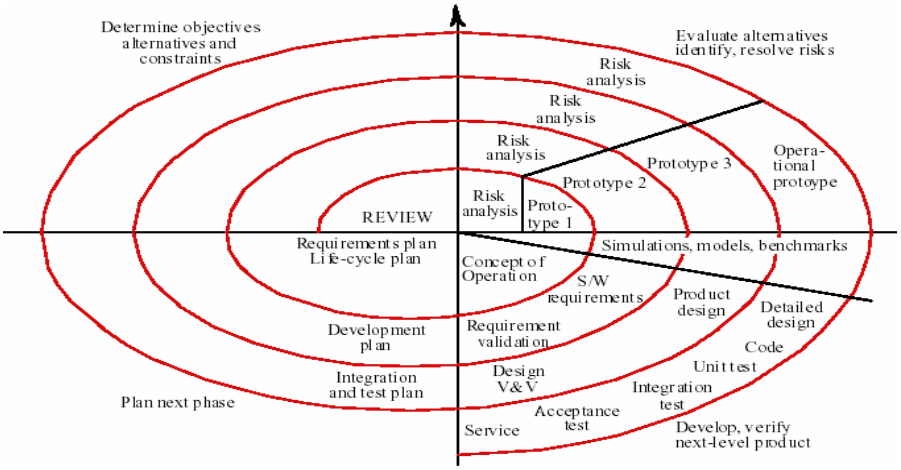

Figura 3 O modelo Espiral

2.3.2.4 Técnicas de quarta geração

Segundo [7], as técnicas de quarta geração (4GT) abrange um amplo conjunto de ferramentas que permitem ao desenvolvedor de software especificar características do software em um nível elevado. A ferramenta automaticamente gera o código fonte, tendo como base a especificação do desenvolvedor. Elas não são um modelo propriamente dito.

Nesse trabalho optou-se por seguir o paradigma da prototipação, porque contribuem para melhorar a qualidade da especificação dos futuros programas, o que leva à diminuição dos gastos com manutenção. Sendo assim, é oferecida ao usuário uma maior integração com o software sendo desenvolvido.

2.4 Tecnologias utilizadas

Para se chegar a um produto final, é preciso usar várias ferramentas e tecnologias que auxiliam o desenvolvimento do software. As seções a seguir citam as ferramentas e tecnologias utilizadas e explicam como cada uma surgiu e o que elas fazem.

2.4.1 A história de Java

Em 1991, a Sun MicroSystems financiou uma pesquisa corporativa interna com o codinome Green. Segundo [8], o projeto resultou no desenvolvimento de uma linguagem baseada em C e C++ que seu criador chamou de OAK (carvalho). Descobriu-se mais tarde que já havia uma linguagem de computador chamado OAK. O nome Java, cidade de origem de um tipo de café importado, foi sugerido e aceito.

Como o mercado para dispositivos eletrônicos inteligentes destinado ao consumidor final não estava se desenvolvendo tão rapidamente como a Sun tinha previsto, o projeto Green atravessou algumas dificuldades e chegou a ter risco de cancelamento. Naquela época, a World Wide Web explodiu e o elenco da Sun reparou o imediato potencial de utilizar Java para criar páginas da Web com conteúdo dinâmico. Isso deu nova vida ao projeto. Java gerou interesse imediato na comunidade comercial por causa do fenomenal interesse pela World Wide Web. Java é agora utilizada para criar páginas para Web com conteúdo interativo e dinâmico, para desenvolver aplicativos corporativos de larga escala e para aprimorar a funcionalidade de servidores da World Wide Web.

2.4.2 Java Server e J2EE

Java 2 Platform Enterprise Edition - J2EE, é a plataforma para desenvolvimento, distribuição e gerenciamento de sistemas de informação multi-camada utilizando tecnologia Java. A plataforma favorece o desenvolvedor de sistemas corporativos com a abilidade de executar em qualquer ambiente dentro uma vasta gama de sistemas operacionais.

2.4.3 JavaServer Pages – JSP

A tecnologia Sun JavaServer Pages (JSP) permite a desenvolvedores e designers Web criar e manter páginas Web dinâmicas, ricas em informação, e implementar poderosas soluções para as intranets. Fazendo parte da família Java da Sun, a tecnologia JSP possibilita rápido desenvolvimento de aplicações que são independentes de plataforma.

Segundo [9], Páginas JavaServer usam cláusulas similares a XML e scriptlets, escritos na linguagem de programação Java, para encapsular a lógica que gera o conteúdo para a página. Além disso, a lógica da aplicação pode residir em recursos baseados no servidor que a página acessa com estas tags e scriptlets e toda e qualquer tag de formato (HTML ou XML) é retornada diretamente para a página de resposta.

2.4.4 Banco de Dados MySQL

MySQL é um sistema gerenciador de banco de dados (SGBD) multi-usuário e multi-threaded. SQL é a linguagem de banco de dados mais popular no mundo atualmente. SQL é também uma linguagem que facilita o armazenamento, o acesso e a manutenção da informação.

3. METODOLOGIA

Essa seção tem como objetivo mostrar como o sistema foi desenvolvido e explicar porque determinadas ferramentas e tecnologias foram utilizadas.

3.1 Ambiente de Trabalho

O presente trabalho foi desenvolvido no Departamento de Ciência da Computação da Universidade Federal de Lavras/MG – UFLA. Para o seu desenvolvimento, foram realizados alguns estudos bibliográficos sobre Sistemas de Informações de Gestão de Eventos, Engenharia de Software e Interfaces Homem-Máquina. Posteriormente, foi necessário aprender ferramentas de desenvolvimento que ajudaram na implementação do sistema. Assim sendo, buscou-se o aprendizado de ferramentas como o Mysql, que serviu como servidor de banco de dados, a linguagem Java, que foi usada para a implementação, o editor de texto Gel, e o IDE Sun One, que facilitou a implementação.

Para a implementação do sistema, os computadores usados foram o do DCC/UFLA rodando o Windows XP e o da empresa Berin eventos, com o sistema operacional Windows 98 instalado. O sistema desenvolvido nos dois casos foi testado com êxito e não apresentou anomalias ou incompatibilidade.

3.2 Linguagem e Tecnologias Utilizadas

Por ser um sistema altamente configurável e portável, a linguagem e as tecnologias utilizadas são gratuitas e podem ser executados em qualquer sistema operacional sem precisar se preocupar com mudanças importantes no código fonte.

Banco de dados

Para o banco de dados o servidor escolhido foi o Mysql, que segundo [12], é um banco de dados relacional que utiliza a linguagem SQL (Structured Query Language) para as consultas. Apesar de existir vários bancos de dados no mercado, a nossa escolha baseou-se no Mysql por ser uma tecnologia gratuita. A aceitação de produtos de software livre e de fonte aberto para o desenvolvimento de soluções tecnológicas vem crescendo cada vez mais. É importante ressaltar também que o atual contexto da administração pública representa economia de recursos, redução de burocracia e independência para buscar soluções de tecnologia de informação. Segundo [13], os pontos fortes do Mysql se resumem nos seguintes tópicos:

Pontos Fortes:

- Natureza econômica. O MySQL é gratuito para a maioria da aplicações e pode ser obtido tanto na Internet quanto nas distribuições do Linux Conectiva e RedHat;

- MySQL foi projetado pela empresa sueca TcX para aplicações baseado na WEB, cujos dados são mantidos por um pequeno conjunto de programas;

- A instalação do produto é muito simples a partir dos pacotes binários;

- A administração do produto é simples e fácil;

- O MySQL possui um manual de referência bem completo sobre o produto que se encontra disponível no site da web http://www.mysql.com. O manual é atualizado a cada alteração do produto que é disponibilizada;

- Aceita programação Java (via Java Database Connectivity – JDBC) e existem muitas classes implementadas e testadas, muitos drivers JDBC que fazem a interface do código Java com o

servidor de banco de dados, e, vários deles são distribuídos gratuitamente para que os programadores de Java possam acessar os seus dados com muita segurança e muita eficiência;

- É uma tecnologia multiplataforma;

- Possui um bom desempenho em relação a outros bancos de dados segundo [14];

Linguagem

Hoje dia, existem centenas de linguagem de programação e cada uma busca satisfazer e dominar o mercado de desenvolvimento de software que gera bilhões de reais todos os anos. A linguagem escolhida para ser utilizada na implementação foi o Java. Alem de ser gratuita e multiplataforma, o Java possui conectividades com vários gerenciadores de banco de dados incluído o MySQL. Entre os principais motivos é importante lembrar que:

- É uma linguagem muito simples, muito poderosa e fácil de aprender;

- Programar e Manter os programas em Java é tarefa simples, uma vez que é uma linguagem orientada a objetos pura [8];

- Não possui elementos complexos e difíceis de programar, como ponteiros ou herança múltipla, o que a torna uma linguagem mais robusta e tolerante a falhas [8];

- O gerenciamento de memória é feito automaticamente;

- É uma linguagem de alto nível e possui sintaxe parecida com a do C++;

- O Java é um membro do paradigma orientado a objetos (OO) das linguagens de programação. As linguagens que aceitam este paradigma, como Java e C++, seguem a mesma filosofia básica, mas diferem em sintaxe e estilo. As linguagens orientadas a objetos oferecem muitas vantagens sobre as linguagens procedurais tradicionais. Como os objetos encapsulam dados e funções relacionados em unidades, é fácil localizar dependências de dados, isolar efeitos de alterações e realizar outras atividades de manutenção e, talvez o mais importante, as linguagens OO facilitam a reutilização [8];

- O Java é distribuído. Distribuição de informações para compartilhamento e trabalho conjunto, com a distribuição de carga de trabalho do processamento, é uma característica essencial dos aplicativos cliente/servidor. Java disponibiliza uma biblioteca de procedimentos TCP/IP incluída nos códigos-fonte e de distribuição binária do Java. Isso facilita aos programadores o acesso remoto às informações, usando protocolos como HTTP e FTP [8];

- O Java é ao mesmo tempo compilado e interpretado. Os programas do Java são compilados em formato binário de código de bytes, que então são interpretados por um ambiente de execução do Java específico da plataforma em questão;

- O Java é independente de arquitetura. A neutralidade do Java em relação à arquitetura é fascinante, mas não se trata de um conceito novo. Derivado da natureza distribuída de cliente/servidor, um importante recurso de projeto do Java é o suporte a cliente e servidores em configurações heterogêneas de rede. O método escolhido para atingir esse objetivo foi uma representação binária de arquitetura neutra para os programas em Java [8];

- O Java é portável. A característica de neutralidade da arquitetura Java é o grande motivo pelo qual os programas em Java são portáveis. Outro aspecto da portabilidade envolve a estrutura ou os tipos de dados inerentes da linguagem, como inteiro, string e

ponto flutuante. O compilador Java foi escrito com o próprio Java, enquanto seu ambiente de tempo e execução foi escrito em ANSI C e tem uma interface de portabilidade bem definida e concisa;

- O Java é multitarefa. Os objetos binários de códigos de bytes do Java são formados por seqüências de execução múltiplas e simultâneas. Essas seqüências são conhecidas como contextos de execução ou processos leves. As linguagens C/C++ são membros de um paradigma de execução em seqüência única, por não oferecerem suporte a seqüências no nível de linguagem. O Java, no entanto, oferece suporte no nível de linguagem para multitarefa, resultando em uma abordagem de programação mais poderosa e de múltiplas facetas;

- O Java é dinâmico. O projeto dinâmico permite que os programas Java se adaptem aos ambientes computacionais mutantes. Por exemplo, a maior parte dos desenvolvimentos típicos em C++ se baseia muito em bibliotecas de classes que podem ser de propriedade desenvolvida por terceiros. Muitas bibliotecas de terceiros, como as distribuídas com sistemas operacionais ou sistemas de janelas, são linkeditadas dinamicamente e vendidas ou distribuídas separadamente dos aplicativos que delas dependam. Quando essas bibliotecas são atualizadas, os aplicativos que dependerem dela poderão apresentar problemas, até que sejam recompilados e redistribuídos. Isso adiciona mais um custo à manutenção do software. O Java evita esse problema atrasando a união dos módulos. Isso permite que os programadores tirem total proveito da orientação a objetos. É possível adicionar novos métodos e variáveis de instâncias em classes de bibliotecas, sem causar problemas aos programas, aplicativos ou clientes já existentes;

- O Java é robusto. Quanto mais robusto um aplicativo, mais confiável ele será. Isso é desejável tanto para os desenvolvedores de software quanto aos consumidores. A maioria das linguagens

OO, como C ++ e Java, possuem tipos bastante fortes. Isso significa que a maior parte da verificação de tipos de dados é realizada em tempo de compilação e não em tempo de execução. Isso evita muitos erros e condições aleatórias nos aplicativos. O Java, ao contrário do C ++, exige declarações explícitas de métodos, aumentando a confiabilidade dos aplicativos;

- O Java é seguro. Como o Java foi criado para ambientes de rede, os recursos de segurança receberam muita atenção. Os aplicativos Java apresentam garantia de resistência contra vírus e de que não são infectados por vírus, pois não são capazes de acessar heaps, stacks ou memória do sistema. No Java, a autenticação do usuário é implementada com um método de chave pública de criptografia. Isso impede de maneira eficaz que hackers e crakers examinem informações protegidas como nomes e senhas de contas;

- O Java é simples. Um dos principais objetivos do projeto do Java foi criar uma linguagem o mais próxima possível do C ++, para garantir sua rápida aceitação no mundo do desenvolvimento OO. Outro objetivo do seu projeto foi eliminar os recursos do C ++, que fugiam à compreensão e aumentavam a confusão que poderia ocorrer durante as fases de desenvolvimento, implementação e manutenção do software. O Java é simples porque é pequeno. O interpretador básico do Java ocupa aproximadamente 40k de RAM. Mesmo a memória combinada de todos esses elementos é insignificante, se comparada a outras linguagens e ambientes de programação;

- O Java oferece alto desempenho. Há muitas situações em que a interpretação de objetos de códigos de bytes proporciona desempenho aceitável. Mas, em outras circunstâncias exigem desempenhos mais altos. O Java concilia tudo isso oferecendo a tradução dos códigos de bytes para o código de máquina nativo em tempo de execução. O alto desempenho permite a implementação de seus aplicativos WEB em Java, na forma de programas pequenos e velozes, que podem ampliar significantemente os recursos tanto do cliente quanto do servidor.

IDE Utilizado

Como IDE (Ambiente de Desenvolvimento) , o Sun One foi um dos aplicativos escolhido para o desenvolvimento da interface gráfica do software. O Sun ONE Studio é uma multiplataforma Java IDE modular e extensível que permite ao engenheiro de software trabalhar onde preferir, nos ambientes operativos Solaris, Linux ou Windows. O Sun ONE Studio fornece, tanto aos profissionais com experiência como sem experiência, todas as ferramentas de que necessitam para serem bem sucedidos. As principais características, segundo [15], do Sun One são:

- Permite o desenvolvimento rápido das aplicações empresariais;
- Capacidade de desenvolvimento iterativo mais curto através de ferramentas integradas;
- Fácil acesso a bases de dados com desenvolvimento simplificado JDBC;
- Ambiente de desenvolvimento altamente produtivo;
- Estrutura de desenvolvimento extensível. Os modelos e as APIs abertas permitem ligar e/ou criar funcionalidade para IDE e aumentar a produtividade;

O Gel também foi usado como IDE por que oferece uma grande facilidade de uso e escrita de código. Com o Gel , foi possível navegar entre as diversas classes do programa e fazer as correções necessárias sem ter que mexer no programa como um tudo, mas somente em uma linha.

3.3 Detalhes de implementação

Ao longo da implementação do sistema, muitos problemas e dificuldades surgiram e diversas soluções apareceram para resolvê-los. A principal dificuldade foi para criar um aplicativo que pudesse gerar qualquer evento em tempo real e que pudesse atender a qualquer usuário.

Uma das maiores dificuldades era prever as requisições dos futuros usuários. Como era impossível prevê-las, optou-se para deixá-los fazê-lo em tempo real. Buscou-se então implementar uma solução onde a interface com o usuário teve um papel importante e decisivo.

Foram usadas caixas de seleção múltiplas que informam ao banco de dados quais tabelas exatamente seriam essenciais para uma dado evento e permitia ao aplicativo automaticamente criar o essencial e descartar as outras possibilidades. As caixas de seleção múltiplas são os meios de interação entre os usuários do sistema e o software.

Depois de ter achado uma forma de gerar os eventos de maneira dinâmica, foram realizadas três implementações diferentes do mesmo aplicativo para destacar a melhor opção de implementação em relação ao acesso dos dados a partir do banco de dados escolhido e do driver utilizado para fazer a ponte entre as informações do sistema e o sistema propriamente dito, a saber:

- Uma implementação usa somente um banco de dados totalmente estático. Um banco de dados estático é um sistema que tem o seu banco de dados (estrutura e tabelas) criadas na hora da codificação.

- Uma outra implementação usa um banco de dados estático, mas com tabelas dinâmicas criadas em tempo de execução;

- Na última, além de cada evento ter tabelas dinâmicas, cada evento ganha também o seu próprio banco de dados gerado em tempo real. No entanto, a estrutura que irá abrigar o

conjunto de tabelas é criada em tempo real assim como as tabelas.

As vantagens e desvantagens de cada uma dessas implementações serão discutidas na próxima seção.

Vantagens e desvantagens da primeira implementação

Cada implementação apresentou suas especificidades e cada uma tem os seus pontos fortes e fracos que podem ser vistos como as suas vantagens e desvantagens. Quando se usa um único banco de dados contendo tabelas já definidas como esquema, o aplicativo apresentou os seguintes pontos fortes:

- O software fica muito mais leve e muito mais rápido. Isso se deve ao fato de ter as estruturas do banco de dados já criadas fora do aplicativo (nome do banco de dados e as tabelas). Assim, o software não perde tempo com a definição dos dados (DDL-Data Definition Language), mas somente com a manipulação dos dados coletados em tempo real;

- Cada tabela à medida que novos eventos são criados, são preenchidas com vários campos e não somente com um campo.

- Ocupa menos espaço.

Os pontos fracos dessa abordagem são:

- Como todos os dados estão dentro de um único banco de dados gigante, quanto mais os dados crescem, mais fica inviável a manutenção dos mesmos. O único meio de localizar um evento ou as suas características é através de uma chave primária inserida em todos os campos de todas as tabelas;

- Muitos dados são repetidos, o controle de redundância é muito precário;

- Não tem nenhuma restrição aos dados;

- O que era muito complexo na administração dos dados vira um caos quando já se tem um número razoável de eventos gerados;

- Não tem flexibilidade para futuras mudanças das estruturas de dados;

- Segurança dos dados muito fraca, só existe um banco de dados com tabelas estáticas. Se essa única estrutura for prejudicada, o risco de perder todos os dados é muito grande;

- Backup demorado;

- Difícil liberação de espaço.

Vantagens e desvantagens da segunda implementação

Como foi mostrado, a primeira implementação tem alguns pontos positivos, mas deixou muito a desejar em relação à segurança, à flexibilidade e na redundância dos dados. A segunda implementação trouxe as seguintes modificações:

Pontos fortes:

- Backup dados mais viável e mais rápido;

- A liberação de espaço no disco é muito mais simples de realizar e muito mais lógica;

- A gerência dos dados não é tão complexa, visto que cada evento tem as suas próprias tabelas e não está incluído dentro de campos gigante;

- O tempo de execução diminui um pouco, mas, de maneira desprezível em relação aos ganhos de gerência e de qualidade da estrutura de dados;

- Não precisa acrescentar uma tabela para guardar as relações entre as tabelas;

- Os dados são mais independentes.

Pontos fracos:

- O programa ficou um pouco mais lento;

- Ainda temos um único banco de dados. Isso significa que se o banco de dados principal se perde, os dados de todos os eventos serão automaticamente perdidos também;

- Em 80% dos casos cada tabela só guarda uma tupla de dados, em outras palavras, criou-se uma tabela em tempo de execução só para guardar um registro ou uma linha de dados;

- A segurança melhorou, mas ainda deixa a desejar.

Vantagens e desvantagens da terceira implementação

A terceira e última implementação é á que mais se destacou entre todos. Escolheu-se essa última pelos seguintes fatos:

Ponto forte:

- A manutenção dos dados ficou muito fácil de realizar porque cada evento ganhou agora o seu próprio banco de dados. Agora não se

precisa somente de uma chave para localizar um evento mais pode localizá-lo a partir das suas características próprias;

- A redundância é quase inexistente;

- A administração dos dados, apesar de ser complexa, se tornou muito simples;

- A flexibilidade para futuras mudanças das estruturas de dados se destacou, agora elas são realmente independentes;

- A segurança dos dados é muito bem estruturada, existem vários banco de dados com tabelas dinâmicas (um banco de dados para cada evento). Com isso, não existe mais o risco de perder todos os dados de uma vez só por uma fatalidade qualquer;

- Fácil liberação de espaço;

- Backup rápido e simples.

Pontos fracos:

- O software fica mais lento e um pouco mais pesado, isso se deve ao fato de ter as estruturas do banco de dados criadas dentro do aplicativo (nome do banco de dados e as tabelas), em tempo de execução. Assim, o software perde tempo com a definição dos dados (DDL -Data Definition Language) e também com a manipulação dos dados coletados em tempo real (DML –Data Management Language);

- Em 80% dos casos, as tabelas, à medida que novos eventos são criados, são preenchidas somente com um campo ou uma tupla;

- Ocupa mais espaço;

- O custo para abrir uma conexão toda vez que temos que fazer uma alteração ou uma consulta é muito grande.

4. Resultados e discussão

Neste tópico será mostrado como é o desenvolvimento de um software comercial utilizando o Java2SE.

O desenvolvimento da aplicação se baseia em uma série de dados sobre a realização de um evento comercial. O objetivo é a criação de uma aplicação genérica. Será criado uma aplicação que possa ser usada tanto na organização de um evento específico como de qualquer outro, bastando para isso acrescentar textos e figuras correspondentes a um determinado evento.

4.1 A modelagem

A figura 4 mostra o esquema do primeiro passo da construção da aplicação [17].

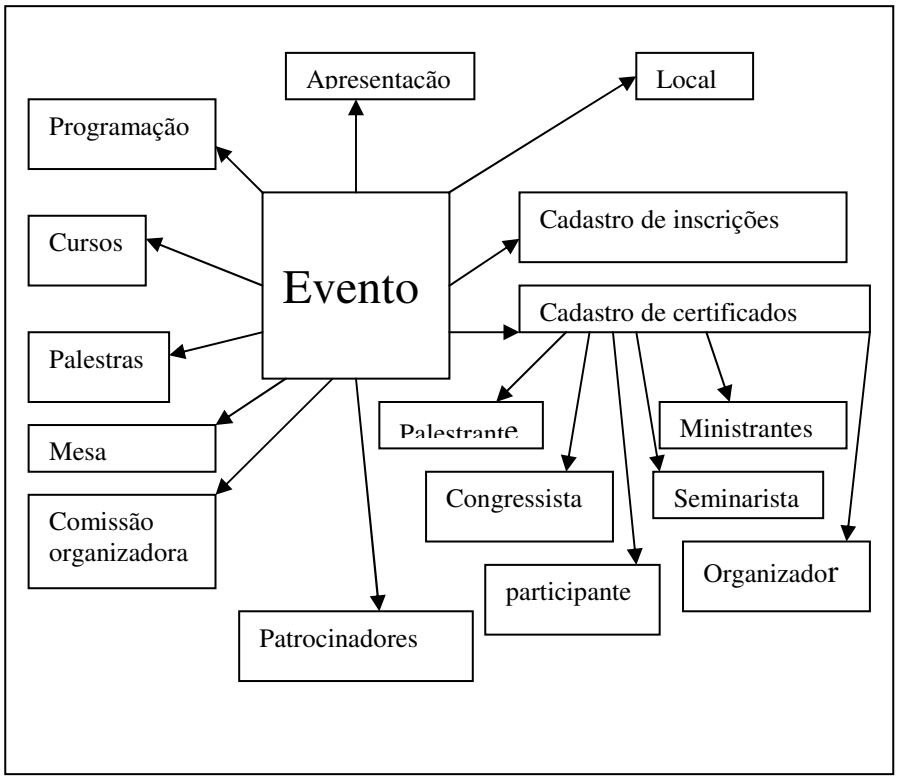

Figura 4 Esquema de classes da aplicação

Segundo [16], o esquema mostrado na figura 4 possui uma estrutura de acesso global denominada **Evento**, que é o índice principal, através do qual se tem acesso às principais classes do esquema.

Serão mostrados alguns detalhes das classes presentes na figura relativo aos seus atributos, perspectivas, sub-classes, etc.

A primeira classe a ser analisada é a classe denominada **Principal**. Nesta classe, existe uma **descrição** sobre a apresentação do evento, bem como os principais objetivos e metas a serem. A figura 5 ilustra a classe **Principal.**

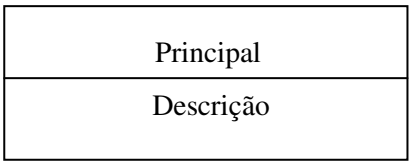

Figura 5 – A classe "Principal"

A próxima classe a ser analisada é a classe Local. Nela se encontram os atributos de Localização Física, que especifica o endereço do local onde será realizado o evento. Estes atributos são: **Nome**, **Telefone**, **Cidade**, **U.F**, **Capacidade**, **Reserva**, **Data**, **Horário** e **Valor_aluguel**; a figura 6 ilustra a classe Local.

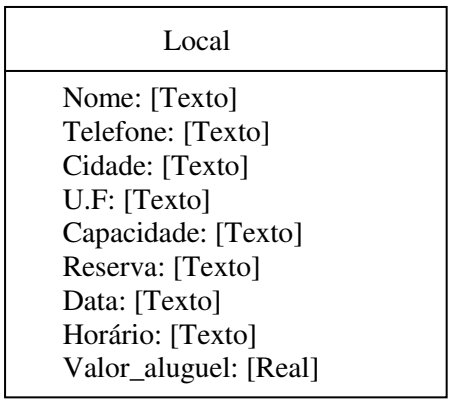

Figura 6 – A classe "Local" e seus atributos

A figura 7 apresenta a classe Programação. Nesta classe se encontra uma descrição sobre a **Data**, o **Horário** e o nome do **Evento** que será realizado.

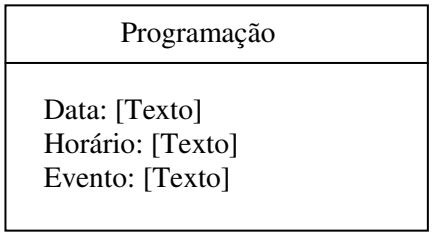

Figura 7 – A classe "Programação" e seus atributos

A próxima classe ilustrada na figura 8 é a classe Cursos. Ela contém uma descrição sobre o **nome** do curso, sua **descrição** e seu respectivo **código**, **ministrante**, **datas** de início e fim do curso, **horário** e **local** em que será realizado, número de **vagas**, **carga horária** e seu respectivo **valor**.

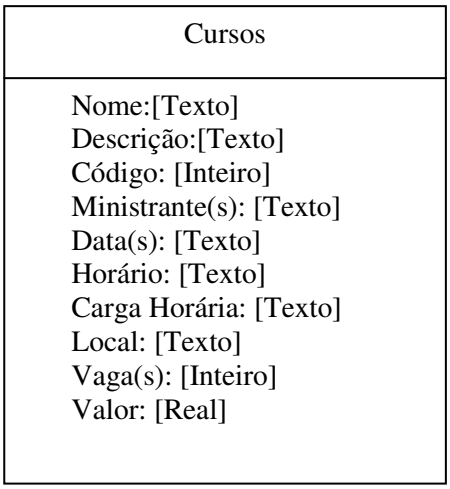

Figura 8 – A classe "Cursos" e seus atributos

A figura 9 apresenta a classe Palestras. Ela contém uma descrição sobre o **tema** da palestra, sua **descrição** e seu respectivo **código**, nome do

palestrante, **datas** de início e fim da palestra, **horário** e **local** em que será realizada, número de **vagas**, **carga horária** e **valor**.

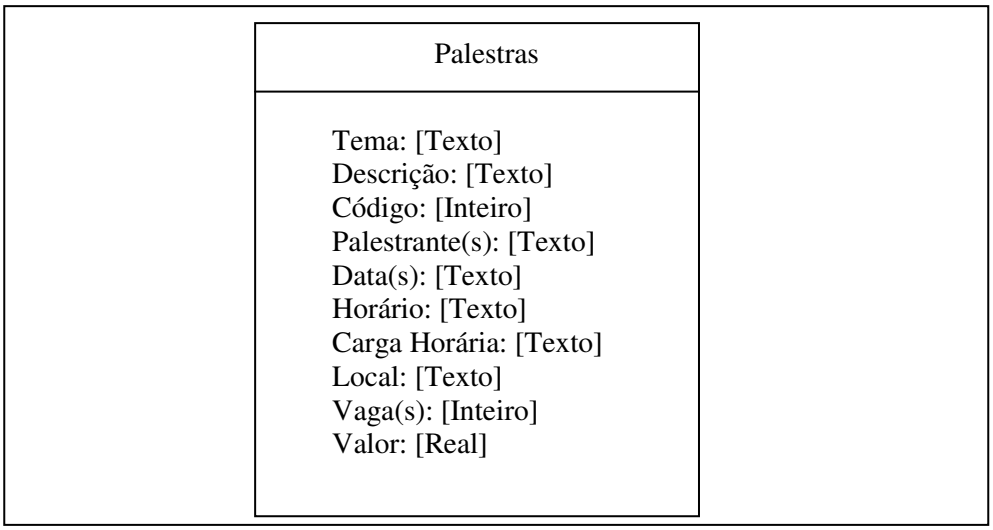

Figura 9 – A classe "Palestras" e seus atributos

A classe Seminários apresentada na figura 10 contém uma descrição sobre o **tema** do seminário, sua **descrição**, seu **código**, **seminarista**, **datas** de início e fim do seminário, **horário** e **local** em que será realizado, número de **vagas**, **carga horária** e o respectivo **valor**.

Figura 10 – A classe "Seminários" e seus atributos

A figura 11 apresenta a classe Mesa Redonda, que contém uma descrição sobre o **tema** da me sa redonda, sua **descrição**, seu **código**, nome dos **integrantes** da mesa, **datas** de início e fim do debate, **horário** e **local** em que será realizado, número de **vagas**, **carga horária** e seu **valor**.

Figura 11 – A classe "Mesa Redonda" e seus atributos

A classe **Celebrações**, apresentada na figura 12, contém os atributos **tipo** de celebração com suas respectivas **Datas**, **Horários**, **Local** e **responsável** de realização.

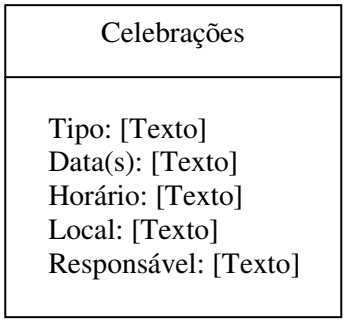

Figura 12 – A classe "Celebrações" e seus atributos

A próxima classe, que é apresentada na figura 13, é a classe **Participantes**. Esta classe contém os atributos **Nome** , **endereço**, **cidade**,

U.F., **C.P.F**., **R.G**, **Atividades**, **Telefone**, **E-mail**, **Página**, e **Código do evento.**

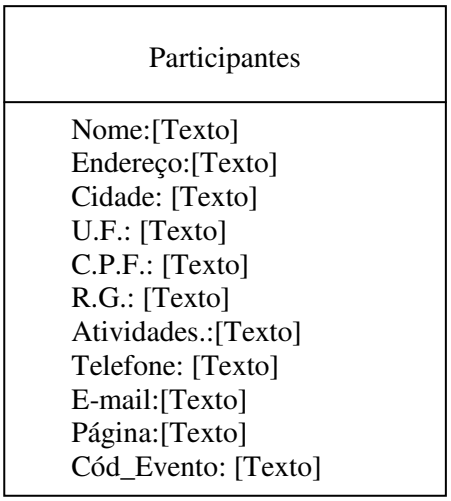

Figura 13 – A classe "Participantes" e seus atri butos

A classe **Comissão Organizadora**, apresentada na figura 15, contém atributos como **nome**, **Tipo**, **E-mail** da lista, **Componentes** e **Números de Integrantes**.

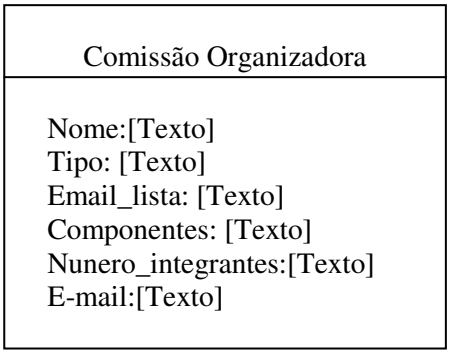

Figura 15– A classe "Comissão Organizadora" e seus atributos

A última classe a ser apresentada é a classe **Patrocinadores** apresentada na figura 16. Ela contém **logos** dos principais patrocinadores do evento, além de ter também os atributos: **Nome**, **tipo**, **Página**, **E-mail**, **Ajuda**, **telefone**, e **Pessoa_contato**.

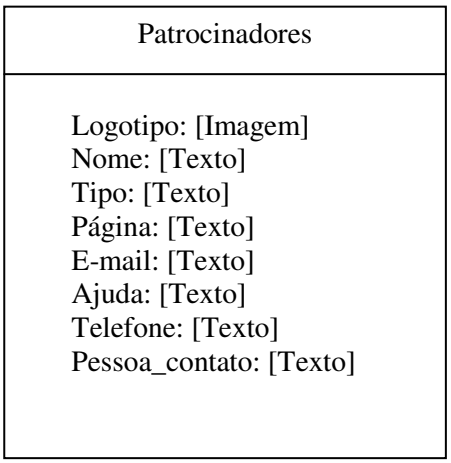

Figura 16 – A classe "Patrocinadores" e seus atributos

4.2 **Design da interface**

A interface é uma das partes mais importante no desenvolvimento de um sistema e, na sua construção, deve-se preocupar com todos os detalhes que lhe dizem respeito, mesmo com aqueles que parecem não ser importante.

A primeira tela a ser definida é a tela de **Autenticação**. Essa tela permite á um usuário entrar no sistema e começar a trabalhar. Se a senha informada não for correta, uma outra tela de erro é mostrada e o usuário deve entrar de novo com os dados corretos. A figura 17 mostra a tela de autenticação e a figura 18 a tela de erro mostrada ao usuário quando os dados não são preenchidos corretamente.

Figura 17 – tela de autenticação do sistema

Figura 18 – erro

A próxima tela, mostrada na figura 19, é a tela principal do sistema. Nela os usuários podem criar eventos, inscrever participantes cadastrar ou alterar dados ou qualquer outra tarefa administrativa.

Figura 19- Tela principal do sistema

A tela onde é possível criar novos eventos é mostrada na figura 20. A partir dessa tela, o usuário do sistema pode escolher quais características pertence ao evento que está sendo criado.

Figura 20- Tela que estrutura um novo evento

As outras telas que dizem respeito à administração e a criação de um dado evento apresentam a mesma estrutura global. Uma janela interna com várias caixas de texto é apresentada ao usuário para que ele possa entrar com os dados no sistema. No final da janela, tem a opção de cadastrar os dados digitados ou de fazer a mesma operação de novo, mas, desta vez para uma outra modalidade, um botão para limpar os campos e um outro para cadastrar de vez os dados e passar para uma outra tela.

Na tela seguinte, é possível cadastrar as celebrações que constam no evento com a possibilidade de cadastrar mais de uma celebração.

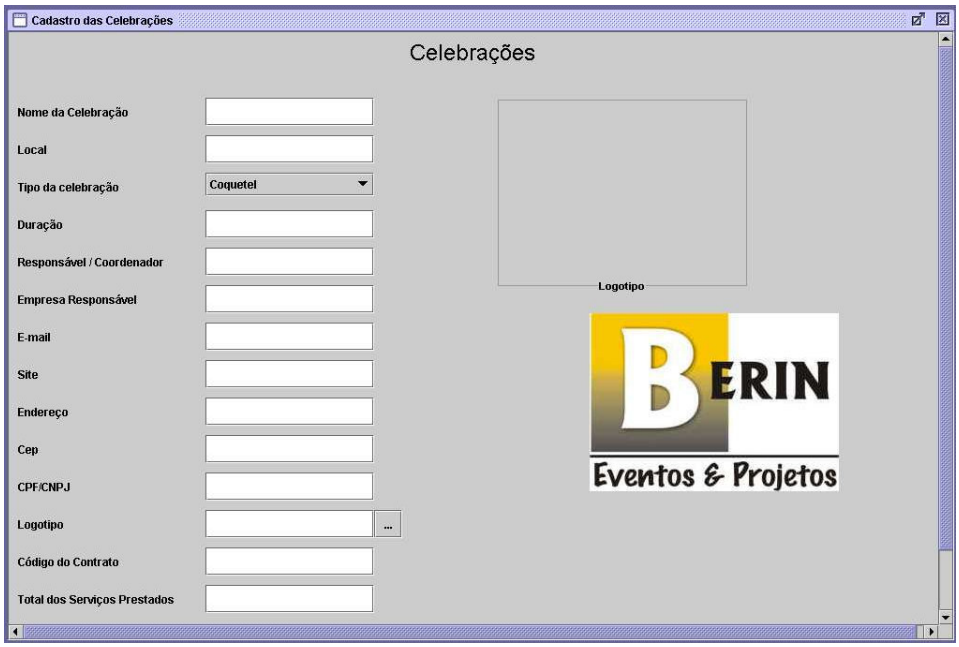

Figura 21- Celebrações

As telas **Hospedagem, Palestrantes, Palestras, Patrocinadores, Promotores, Publicidade, Tipos de Comissões e Transporte,** terão o mesmo comportamento da tela "**Celebrações**" (elas serão acessadas através da tela de criação do novo evento) e por isso não são mostradas.

O Sistema de Gestão de Eventos permite também realizar várias consultas. Algumas delas são mostradas nas figuras 22, 23, 24 e 25.

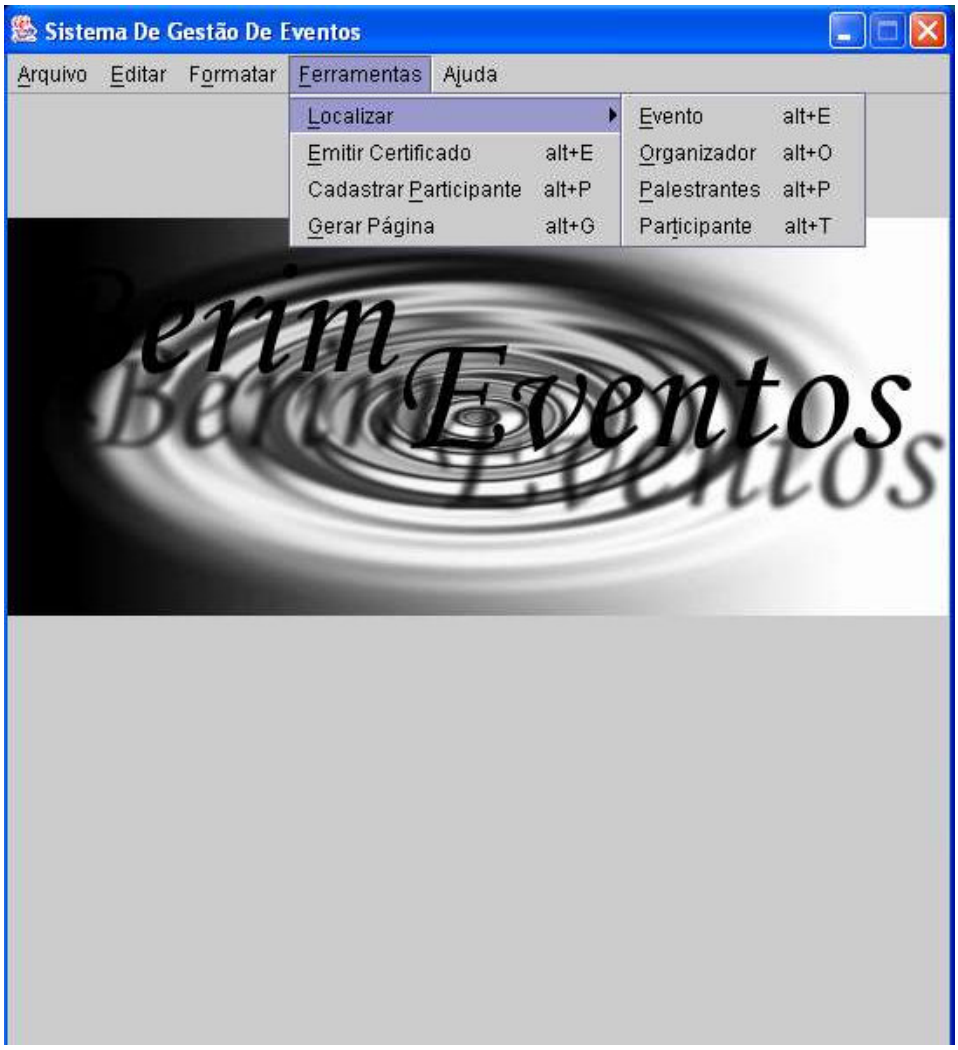

Figura 22- Opções de consulta

Um exemplo de busca de um participante de um dado evento é ilustrado na figura 23.

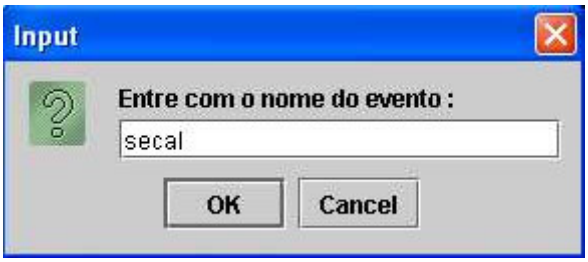

Figura 23- Para achar um aluno é preciso entrar como nome do evento

Figura 24- Janela que precisa do nome do participante para iniciar a busca

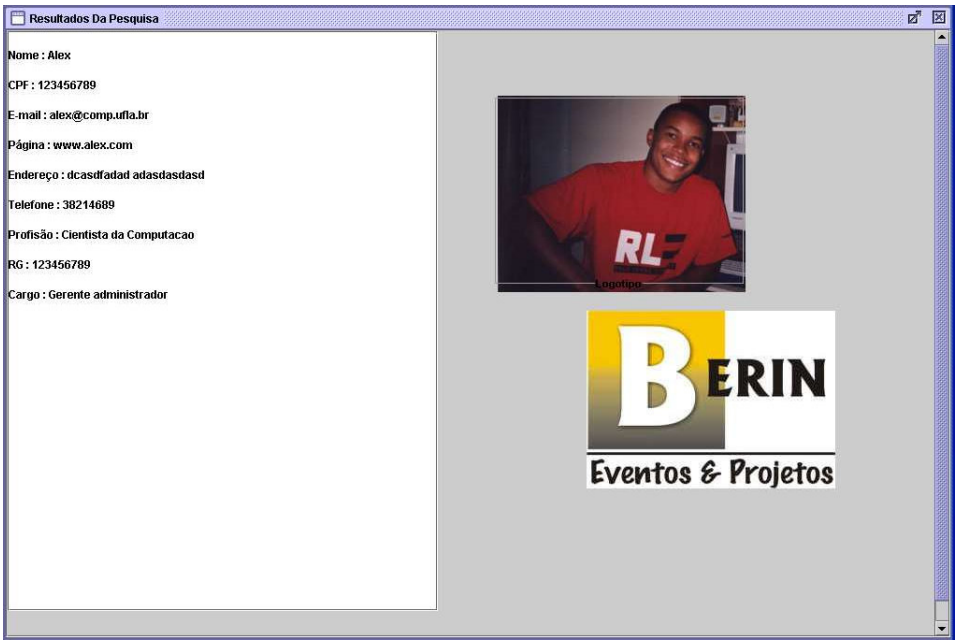

Figura 25- Resultado da pesquisa

O Sistema de Gestão de Eventos também permite que a aparência da tela inicial seja configurada do jeito que o usuário quiser. Isso é possível acessando o menu formatar e cor de fundo. A figura 35 mostra a janela de interação com o usuário.

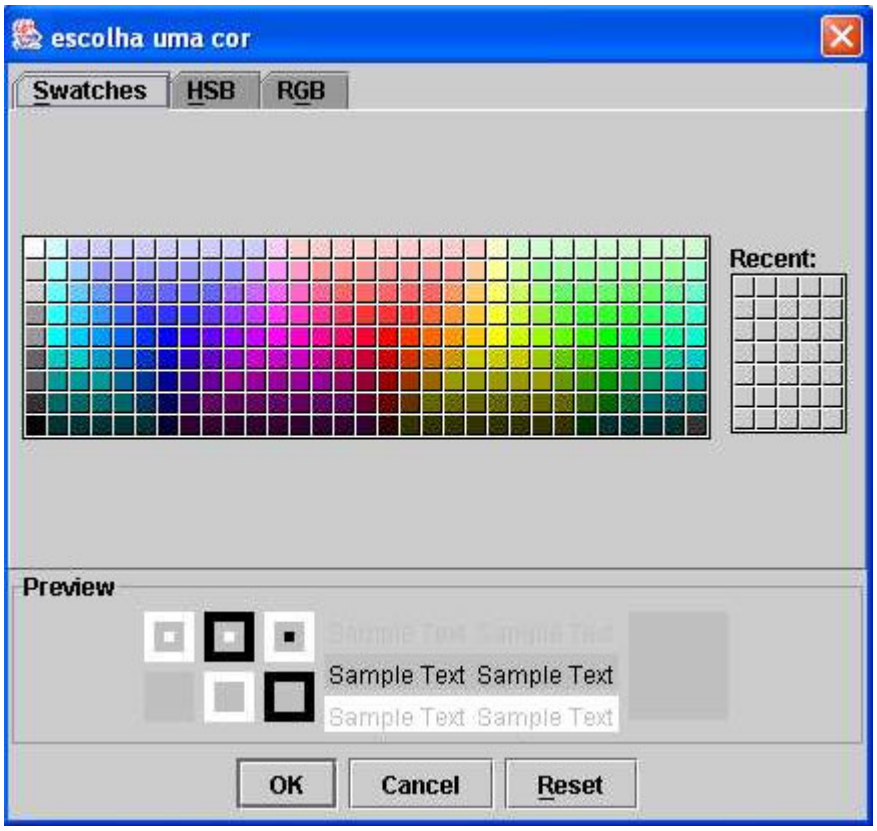

Figura 26- Janela que permite ao usuário personalizar a janela principal do sistema

Conclusão

O presente trabalho apresentou o início do desenvolvimento de um Sistema de Gestão de Eventos para criar, gerenciar e administrar um evento de qualquer tipo e da forma mais específica que ele possa ser criado.

O uso deste sistema trás muitos benefícios e elimina uma boa parte da burocracia que existe nesse ramo. Com ele, é possível organizar, armazenar, e tabelar diversos eventos de maneira segura, simples e rápida.

Em futuras versões, pretende-se implementar rotinas que possibilitam ter um controle maior sobre o acesso aos dados e ao sistema, melhorar o sistema de autenticação e as suas políticas, implementar versões clientes dependentes de uma versão principal e que serão instaladas em máquinas clientes e, finalmente, permitir que o sistema possa ser executado não somente dentro de uma intranet, mas também na Internet, utilizando por isso as tecnologias de JSP e de JEE.

REFERÊNCIA BIBLIOGRÁFICA

[1] SHNEIDERMAN, b. **Designing The User Interface;** Strategies for Effective Human-Computer Interaction. 3.Ed. Reading Addison-Wesley, 1998.

[2] Departamento de Computação – Unicamp

[3] ROGER S. PRESSMAN, **Engenharia de Software**; 1995. p. 3.

[4] **EWERTON**, Plataforma .net .

[5] ROGER S. PRESSMAN, **Engenharia de Software**; 1995.

[6] **ANTONIO MARIA PEREIRA RESENDE**, disponível em WWW em http://www.comp.ufla.br/~tony

[8] **H. M. DEITEL, P.J. DEITEL,** JAVA Como Programar 3ª Edição.

[9]**Sun Microsystems,** disponível em WWW em http://www.java.sunn.com/products/ejb

[10] NETCRAFT. **Netcraft Web Server Survey**. 2001. Disponível em WWW em http://www.netcraft.com/survey/

[11] Apache HTTP Server Project – Disponível em WWW em http://httpd.apache.org/

[12] Marcelo Caetano Martins Muniz, Proposta De Um Framework De Gerenciamento Eletrônico De Documentos Via Web; 2002. p.10 http://httpd.apache.org/

[13] Gabriela Ferreira Drumond, Avaliação Técnica do MySQL; 2001. p.5.

[14] MySQL Benchmarks. 2002 – Disponível em WWW em http://httpd.mysql.com/information/benchmarks.html/

[15] Sun One Studio 4, Suporte & Formação –Disponível em WWW em http://pt.sun.com/produtos/software/development/studio4.html

[16] Zambalde, A. L., Alves, R.M. e Lopes, M.A.: "Modelagem, autoria e análise de usabilidade de aplicação hipermídia direcionada ao setor agropecuário", UFLA,1999.

[17] Gizelle, Modelagem de um sistema de gestão de eventos