Universidade Estadual do Centro-Oeste – UNICENTRO
Centro de Ciências Exatas e de Tecnologia – CET
Departamento de Análise de Sistemas – DESIS

Software de Gestão Escolar - Contraltdel

Gervásio Pires dos Santos

Hermano Pereira

Orientadora: Profª. Josiane Michalak Hauagge

Monografia apresentada ao Departamento de Análise de Sistemas da UNICENTRO, como parte dos requisitos para aprovação na disciplina Estágio Supervisionado.

Guarapuava – 2001

Resumo

Este trabalho tem como objetivo descrever as atividades realizadas durante o desenvolvimento do Sistema de Gestão Escolar do Colégio Decisão, para auxiliar no desenvolvimento das tarefas desempenhadas pela Secretaria do mesmo colégio.

O software permite a emissão de boletins a partir da automação do cadastro de alunos, turmas, disciplinas, séries e cursos. As consultas de boletins podem ser visualizadas através de terminais de computadores no próprio colégio ou através do *site*[1] desenvolvido para este propósito.

A monografia apresenta todo o embasamento teórico necessário para o desenvolvimento do sistema, que foi realizado de acordo com a metodologia de Orientação a Objetos (OO), utilizando-se ferramentas visuais que proporcionam uma interface visual amigável ao usuário, bem como, uma descrição dos métodos, técnicas e ferramentas utilizadas no desenvolvimento do sistema.

[1] s*ite* - página na *Internet* onde se disponibiliza as informações.

Dedicatória

Dedicamos esta monografia à nossas famílias que constantemente incentiva-nos na busca de novos ideais, e a nossa orientadora Prof. Josiane Michalak Hauagge, pela orientação e colaboração.

Agradecimentos

A Deus, que nos deu forças, e que nos presenteou muitos momentos de vitória e sucessos, compartilhando conosco as horas difíceis; nós agradecemos e pedimos; acompanha-nos no feliz desempenho de nossa missão.

Aos Pais que nos deram a vida e nos ensinaram a vive-la com dignidade, não bastaria um obrigado.

Aos Professores que souberam, além de transmitir seus conhecimentos, transmitir-nos sua experiência e apoiar-nos em nossas dificuldades.

Aos Colegas, Amigos, Tios e Primos que também nos apoiaram e incentivaram diante das alegrias e dificuldades encontradas.

Índice

Capítulo I
Introdução

Neste capítulo será apresentado um breve histórico do Colégio Decisão, instituição concedente do estágio, bem como sua área de atuação, um relato quanto à utilização da informática pela secretaria, motivação e os objetivos propostos.

1.1 Histórico do Colégio Decisão

A partir de reuniões realizadas em novembro de 2000, a diretoria do Curso Decisão começou a encaminhar os processos necessários para a formulação oficial do Colégio Decisão.

Denominado *Curso e Colégio Decisão,* a instituição de ensino é representada pelo sócio-gerente Adriano Zagorski, e pelo diretor pedagógico Prof. Nelson Zagorski. O Colégio Decisão foi criado em 30/11/2000 pela Resolução de Nº 3.801/2000 e autorizado pelo CEF/SEED, teve sua grade curricular aprovada em 15/12/2000 pertencendo ao NRE de Guarapuava.

1.2 Uso da Informática na Secretaria do Colégio

Nas dependências da Secretaria do Colégio Decisão, existem dois microcomputadores e uma impressora interligados pela rede da instituição.

O uso da informática como meio facilitador das atividades do setor, se restringe apenas a planilhas e editores de textos. A emissão de relatórios de boletins e listas de alunos e o lançamento das notas e faltas é feita através de sistemas individuais e incompletos, cujas informações são digitadas em processadores de texto e planilhas de cálculo.

1.3 Motivação

Preocupando-se em melhorar o atendimento e otimizar os serviços feitos manualmente pela secretaria do colégio, sentiu-se a eminente necessidade de implantar através de métodos computacionais, um sistema que viesse suprir as necessidades da secretaria, agilizando os serviços e disponibilizando informações de maneira rápida e eficiente aos seus usuários. Após estudos realizados junto aos funcionários e dirigentes do colégio, notou-se a necessidade do desenvolvimento de um sistema de gestão escolar, para que os funcionários pudessem agilizar seus trabalhos e os alunos pudessem ter acesso a suas notas.

1.4 Objetivos Propostos

Após a identificação dos problemas e estudos realizados junto à secretaria do Colégio Decisão, foi proposto o desenvolvimento de um software para emissão de boletins e consultas de notas e freqüências por terminais de computadores instalados no próprio colégio ou através de um *site* na Internet.

Este sistema abrange a automatização do cadastro de alunos, professores, cursos, séries, turmas, áreas de conhecimento, disciplinas, matrículas, lançamento de notas e faltas, e a emissão de boletins e listas de alunos matriculados na turma.

No estudo dos objetivos da secretaria, identificou-se a necessidade de novas informações, que a partir do trabalho manual realizado tornava-se inviável. O software em questão propõe diminuir esta carência, desenvolvendo consultas e pesquisas ao banco de dados para obtenção das informações desejadas.

1.5 Organização da Monografia

O capítulo II apresenta a Revisão Bibliográfica, os principais conceitos de Engenharia de Software que forneceram o embasamento científico para o desenvolvimento do software. Também descreve as metodologias adotadas para análise e especificação do sistema. Ainda, referencia todas as ferramentas que foram utilizadas para o desenvolvimento do software e do *site*, bem como a ferramenta utilizada para modelagem do sistema.

O capítulo III aborda o documento de Análise de Requisitos do sistema e todas as funcionalidades do sistema. Ainda no capítulo III é descrita a fase de Análise do Sistema, apresentando a especificação de requisitos e os casos de uso. Também apresenta a fase de Desenvolvimento do Sistema, os diagramas de classe, seqüência, entidade relacionamento e o projeto de interface do sistema.

O capítulo IV apresenta as conclusões, contribuições e trabalhos futuros oriundos do desenvolvimento do Software de Gestão Escolar.

Capítulo II
Revisão Bibliográfica

Neste capítulo serão apresentados alguns conceitos sobre Engenharia de Software, que através dos seus elementos fundamentais, possibilita o controle do processo de desenvolvimento e oferece o embasamento necessário para construção de software de alta qualidade.

Também serão descritas as metodologias adotadas para o desenvolvimento de um software utilizando os conceitos de Orientação a Objetos e serão apresentadas as ferramentas utilizadas para o desenvolvimento deste trabalho, abordando seus principais aspectos técnicos.

2.1 Software e Engenharia de Software

Software pode ser definido como o conjunto de instruções que quando executadas produzem a função e o desempenho desejado [PRE95].

Devido ao avanço da microeletrônica houve um maior poder de computação a baixo custo e o *software* é o mecanismo que possibilita aproveitar e dar solução a este potencial [AMB97].

O *software* tornou-se o elemento-chave dos sistemas e produtos baseados em computador. No decorrer das últimas quatro décadas, o *software* evoluiu de uma ferramenta de análise de informações e de resolução de problemas especializada para uma indústria em si mesma.

Mas logo a cultura e a história da programação criaram um conjunto de problemas que persistem até hoje. O *software* tornou-se um fator limitante na evolução dos sistemas baseados em computador [PRE95].

A engenharia de *software* é uma disciplina que integra métodos, ferramentas e procedimentos para o desenvolvimento de *software* de computadores.

Os métodos proporcionam os detalhes de como fazer para construir o *software*. Eles envolvem um amplo conjunto de tarefas que incluem: planejamento e estimativas de projeto, análise de requisitos de *software* e de sistemas, projeto da estrutura dos dados, arquitetura de programa e algoritmo de processamento, codificação, teste e manutenção.

As ferramentas fornecem o apoio automatizado ou semi-automatizado aos métodos, podendo ser integradas de forma que a informação criada por uma ferramenta possa ser usada por outra.

Os procedimentos são o elo de ligação entre métodos e ferramentas que permitem construir o *software* de maneira racional e oportuna.

2.1.1 Paradigmas da Engenharia de Software

Uma série de diferentes paradigmas de engenharia de *software* foram propostos, cada um exibindo potencialidades e fragilidades, mas todos tendo uma série de fases genéricas em comum [PRE95]. A seguir serão apresentados os paradigmas para o desenvolvimento de *software*, segundo Pressman [PRE95].

Ciclo de vida clássico

Às vezes chamado de *modelo cascata*, o paradigma deste ciclo de vida requer uma abordagem sistemática, seqüencial ao desenvolvimento de *software*:

Engenharia de Sistemas → Análise → Projeto → Codificação → Teste → Manutenção.

Prototipação

Processo que capacita o desenvolvedor a criar um modelo de *software*. Este paradigma apresenta as seguintes fases:

Início → Coleta e Refinamento dos Requisitos → Projeto Rápido → Construção do Protótipo → Avaliação do Protótipo → Refinamento do Protótipo ou ir para o Projeto Rápido → Engenharia do Produto → Fim.

Modelo Espiral

Foi desenvolvido para abranger as melhores características do ciclo de vida clássico e da prototipação, acrescentando ao mesmo tempo, um novo elemento - a análise de riscos - que faltava a esses paradigmas.

A – Planejamento

B – Análise dos riscos

C – Engenharia

D – Avaliação do Cliente

Coleta inicial dos requisitos e planejamento do projeto → (A) → (B) → (C) → (D) → (A) → (B) → (C) → (D) → ... → (A) → (B) → (C) → (D) → Sistema concluído.

Técnicas de 4ª geração

Cada uma destas ferramentas permite ao desenvolvedor criar uma parte do *software*, pois gera automaticamente o código fonte.

Coleta de Requisitos → Estratégia de Projeto → Implementação usando 4GL → Teste.

Combinando Paradigmas

Os paradigmas podem ser combinados, se necessário, para que se proceda o desenvolvimento do *software*.

2.1.2 Visão Genérica da Engenharia de Software

Algumas fases são genéricas no desenvolvimento de software, sendo utilizadas em todos os paradigmas:

- **Fase de definição:** identifica quais informações devem ser processadas.
- **Fase de desenvolvimento:** tenta definir a estrutura de dados e a arquitetura de *software* a ser projetada.
- **Fase de manutenção:** referente às mudanças que estão associadas à correção de erros e adaptações exigidas.

2.2 Orientação a Objetos

A orientação a objetos começou a se popularizar no mundo corporativo a partir do final da década de 80, e sua aceitação vem aumentando continuamente desde então [AMB97].

É uma estratégia de desenvolvimento baseada no conceito de que os sistemas devem ser construídos a partir de um conjunto de componentes reutilizáveis, chamados

de objetos. Os objetos procuram incorporar simultaneamente os dados e a funcionalidade de um sistema, em vez de considerá-los isoladamente, tal como é feito no paradigma de programação estruturada [SCH2000].

2.2.1 Vantagens da Orientação a Objetos

A orientação a objeto oferece o benefício potencial de resolver muitos dos problemas atualmente enfrentados pela indústria de informática. Algumas das vantagens apresentadas pela OO segundo Amber [AMB97] são:

- **Reusabilidade:** possibilita uma maior eficiência no projeto e oferece oportunidades de reutilização através dos conceitos de herança, polimorfismo, encapsulamento, modularidade e coesão.
- **Extensibilidade:** como as classes possuem tanto dados como funcionalidade, para acrescentar novos recursos ao sistema basta introduzir as modificações necessárias em um só lugar.
- **Aumento da qualidade:** a melhoria da qualidade é possível graças ao maior envolvimento dos usuários na elaboração dos sistemas.
- **Vantagens financeiras:** reusabilidade, extensibilidade e aumento da qualidade conduzem as vantagens econômicas da OO. Possibilita o desenvolvimento de sistemas com qualidade, rapidez e economia.

2.2.2 Paradigmas da Estruturação X Paradigmas da Orientação a Objetos

O paradigma estruturado é uma estratégia de desenvolvimento baseado no conceito de que um sistema deve ser dividido em duas partes: os dados definidos por um modelo de dados e a funcionalidade definida por um modelo de processamento. Assim, utilizando a abordagem estruturada, desenvolvemos aplicações nas quais os dados ficam separados do comportamento tanto no projeto quanto na implementação do sistema [SCH2000].

O paradigma de orientação a objetos, em vez de definir sistemas com duas partes separadas (os dados e a funcionalidade), passa a conceber os sistemas como um conjunto de objetos interativos. Os objetos desempenham ações (ou seja, possuem funcionalidades) e armazenam informações (isto é, contém dados).

2.2.3 UML

Em 1995, Ivar Jacobson, Grady Booch e James Rumbaugh decidiram criar uma linguagem de modelagem padrão que foi chamado de Linguagem de Modelagem Unificada (UML - *Unified Modeling Language*) [SCH2000].

A UML é uma linguagem padrão para especificar, visualizar, documentar e construir artefatos de um sistema que pode ser utilizada com todos os processos ao longo do ciclo de desenvolvimento e através de diferentes tecnologias de implementação. Ela busca unificar as perspectivas entre os diversos tipos de sistemas e fases de desenvolvimento de forma que permite levar adiante determinados projetos que antes não eram possíveis pelos métodos existentes [FUR98].

A UML busca uma simples padronização de notação unificada, uma vez que contém conceitos novos que não são encontrados em outros métodos orientados a objetos. A UML recebeu influência das técnicas de modelagem de dados (Diagrama de Entidade e Relacionamento), modelagem de negócio, modelagem de objetos e componentes [FUR98]. Segundo Furlan [FUR98] os objetivos da UML são:

- fornecer aos usuários uma linguagem de modelagem visual expressiva e pronta para uso visando o desenvolvimento de modelos de negócio;
- fornecer mecanismos de extensibilidade e de especialização para apoiar conceitos essenciais;
- ser independente de linguagens de programação e processos de desenvolvimento;
- prover uma base formal para entender a linguagem de modelagem;
- encorajar o crescimento no número de ferramentas orientadas a objeto no mercado;
- suportar conceitos de desenvolvimento de nível mais elevado tais como, colaborações, estrutura de trabalho, padrões e componentes.

2.2.4 Ciclo de Vida Modelo Incremental

Para amenizar os problemas causados pela pouca informação e pelas alterações de requisitos durante o projeto de sistema de informação, o sistema é dividido em partes ou fragmentos menores, e mais gerenciáveis.

Este modelo de ciclo de vida é chamado de Modelo Incremental, apresentado na figura 2.1, sendo um modelo de processo em que cada estágio mostra incrementos na funcionalidade do produto e tem como característica importante antecipar o *feedback*[2] do usuário.

Após a especificação dos requisitos, o produto é divido em módulos, sendo que cada um destes módulos deve oferecer alguma função que agregue valor ao usuário [SCH2000].

Segundo Schmitz [SCH2000], como neste modelo o usuário vai recebendo partes do sistema que já podem ser utilizadas e como as partes são menores, as variações com relação a prazo e custo também são menores, sendo assim as alterações afetam partes menores do sistema, diminuindo os impactos no custo e no prazo.

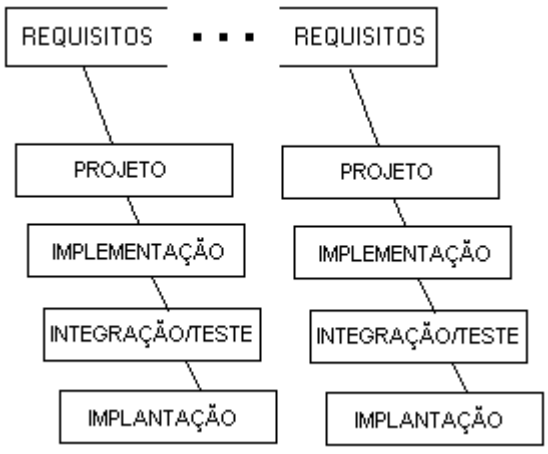


Figura 2.1 – Modelo Incremental

[2] Feedback: fato que se repete.

2.2.5 Metodologia Rápida para Desenvolvimento de Sistemas Orientados a Objetos (MRDS)

É uma metodologia para a especificação de sistemas orientados a objetos, compatível com a UML. A MRDS foi desenvolvida a partir das principais idéias para especificação de sistemas concebidas ao longo destas últimas décadas e testadas exaustivamente dentro do Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro em projetos do "mundo real" [SCH2000].

A MRDS tem por objetivo auxiliar o projetista no desenvolvimento de sistema de informação. Ela cobre as fases de desenvolvimento que vão desde a captura inicial de requisitos até a especificação completa da interface de todos os módulos componentes do sistema. Segundo Schmitz e Silveira [SCH2000], a MRDS se divide em duas fases:

- **1ª. Fase - Definição dos Requisitos:** O objetivo desta fase é definir a funcionalidade esperada pelo usuário do sistema criando um Modelo de Requisitos que é composto pelo Modelo de Caso de Uso, Maquete e Glossário. O Modelo de Caso de Uso por sua vez é composto pelos Diagramas de Caso de Uso e as Descrições do Caso de Uso.
- **2ª. Fase – Análise e Projeto :** O objetivo desta fase é de produzir um plano que permita a construção do sistema criando um Modelo de Análise e Projeto que é composto pelos Modelos Estático e Dinâmico. O Modelo Estático é composto por um conjunto de Diagramas de Classes. O Modelo Dinâmico é composto por um conjunto de Diagramas de Seqüências e Diagramas de Estados.

2.2.6 Bancos de Dados Relacionais

O Modelo de Dados Relacional representa os dados contidos em um Banco de Dados através de relações. Estas relações contém informações sobre as entidades representadas e seus relacionamentos. Consiste em uma coleção de tabelas, cada uma das quais com um nome único [MED2001].

Os bancos de dados relacionais utilizam uma linguagem de consulta, que é a linguagem por meio da qual os usuários obtêm informações do banco de dados. Os sistemas de banco de dados comerciais oferecem linguagem de consulta de alto nível, uma das mais populares é a SQL. Essas linguagens de consulta são concisas e formais [MED2001].

2.2.7 SQL

Quando os bancos de dados relacionais estavam sendo desenvolvidos, foram criadas linguagens destinadas à sua manipulação. O Departamento de Pesquisas da IBM, desenvolveu a SQL(*Structured Query Language*) como forma de interface para o sistema de banco de dados relacional, no início dos anos 70. Em 1986 o *American National Standard Institute (ANSI*), publicou um padrão SQL [SUR97].

Características da Linguagem SQL segundo Surian [SUR97]:

- capacidade de gerenciar índices, sem a necessidade de controle individualizado de índice corrente, algo muito comum nas linguagens de manipulação de dados do tipo registro a registro;
- capacidade de construção de visões, que são formas de visualizarmos os dados na forma de listagens independente das tabelas e organização lógica dos dados,
- capacidade de cancelar uma série de atualizações ou de as gravar, depois de iniciar uma seqüência de atualizações e
- garantia de integridade das relações existentes entre as tabelas e seus índices.

2.2.8 Arquitetura Cliente/Servidor

É uma arquitetura de rede, onde existem dois módulos básicos na rede: o Servidor e os Clientes. O *Servidor* é alguma máquina da rede que é responsável por servir os Clientes da rede com aquilo que é solicitado. Os *Clientes* são as máquinas que solicitaram informações que estarão contidas no Servidor [NEV2000].

O servidor oferece serviços aos seus clientes, por um meio de processamento específico e a interação entre os processos cliente e servidor é uma troca cooperativa, transacional, em que o cliente é ativo e o servidor é reativo. Sendo esta, a principal distinção entre cliente/servidor [NEV2000].

Quando os processos cliente e servidor são executados em máquinas diferentes, estes são conectados através de redes locais ou remotas, sendo que as redes locais são as implementações mais comuns de cliente/servidor. O usuário do sistema interage com um cliente, que por sua vez emite pedidos e recebe resultados do servidor [NEV2000].

É no servidor que normalmente ficam os sistemas mais pesados da rede, tais como o banco de dados. As máquinas clientes são menos poderosas, pois rodam aplicativos que requerem menos recursos das máquinas [NEV2000].

2.3 Ferramentas Utilizadas

A seguir serão apresentadas as ferramentas utilizadas para o desenvolvimento deste trabalho, abordando as vantagens técnicas das linguagens de desenvolvimento do *site* e do sistema, descrevendo o banco de dados utilizado para o armazenamento das informações e a ferramenta Case utilizada para elaboração dos diagramas do sistema.

2.3.1 HTML

Em 1990 em um laboratório europeu, Tim Benners-Lee desenvolveu uma linguagem chamada HTML (*HipperText Markup Language*). Logo após a sua criação, ela foi utilizada como linguagem padrão pela NCSA (*National Center for Supercomputing Applications*) em seu navegador chamado Mosaic™.

Em 1995 a IETF (*Internet Engeneering Task Force*) lançou a versão 2.0, e a partir daí, empresas como a Netscape Communications Corporation™ e a Microsoft Corporation™ inseriram novos comandos no HTML para executar funções em seus navegadores. Atualmente na versão 4.0, o HTML é hoje uma aplicação ISO 8879 (*International Standards Organization*) [RIT96, TAN97, HOR97].

HTML é uma linguagem que determina um padrão para a definição de vinculação entre documentos (hipertexto), sendo assim, uma linguagem básica para a construção de páginas de Internet, que trata de várias características como: inserção de figuras, textos, vinculação, figuras animadas, formulários, entre outros [RIT96].

Suas principais características segundo Ritchey [RIT96]:

- os documentos feitos em HTML tem a mesma aparência em diferentes plataformas;
- por ser somente texto, é de tamanho reduzido e de rápido carregamento, tanto pela Internet como no navegador,
- é possível criar vínculos a outros documentos HTML e
- é possível inserir ilustrações e multimídia através de vínculos.

2.3.2 PHP

A linguagem PHP (*Personal Home Page*) foi concebida durante o outono de 1994 por Rasmus Lerdorf. As primeiras versões não foram disponibilizadas, tendo sido utilizadas em sua *home-page* apenas para que ele pudesse ter informações sobre as visitas que estavam sendo feitas [ROC2000].

A primeira versão utilizada por outras pessoas foi disponibilizada em 1995, e ficou conhecida como *Personal Home Page Tools* (ferramentas para página pessoal). Era composta por um sistema bastante simples que interpretava algumas macros e alguns utilitários que rodavam por trás das *home-pages*: um livro de visitas, um contador e algumas outras coisas [ROC2000].

Em meados de 1995 o interpretador foi reescrito, e ganhou o nome de PHP/FI, o FI veio de um outro pacote escrito por Rasmus que interpretava dados de formulários HTML. Ele combinou os *scripts*[3] do pacote *Personal Home Page Tools* com o FI e adicionou suporte a mSQL, nascendo assim o PHP/FI, que cresceu bastante, e as pessoas passaram a contribuir com o projeto [ROC2000].

[3] *Scripts* são executados diretamente pelo seu interpretador de código-fonte, não precisando compilar.

Houve uma mudança no desenvolvimento do PHP. Ele deixou de ser um projeto de Rasmus com contribuições de outras pessoas para ter uma equipe de desenvolvimento mais organizada. O interpretador foi reescrito por Zeev Suraski e Andi Gutmans, e esse novo interpretador foi a base para a versão 3. Atualmente é disponilizado o PHP4 [ROC2000].

PHP é uma linguagem de *script* utilizado para criação de *websites*[4] dinâmicos. A diferença de PHP com relação à linguagens semelhantes a Javascript é que o código PHP é executado no servidor, sendo enviado para o cliente apenas o HTML. Desta maneira é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente [STO2000].

Assim como as linguagens de programação híbridas como: C, Java e Perl, o PHP também atua como uma HTML, que possibilita a rápida criação de conteúdo dinâmico. O código é executado no servidor *web*[5], juntamente com os documentos HTML. Ele proporciona uma solução de alta escalabilidade e performance [STO2000].

Características do PHP segundo Stoco [STO2000]:

- permite coletar dados de um formulário, gerar páginas dinamicamente ou enviar e receber *cookies*[6];
- suporte a um grande número de bancos de dados, como dBase, Interbase, mSQL, MySQL, Oracle, Sybase, PostgreSQL e vários outros;
- suporte a outros serviços através de protocolos como IMAP, SNMP, NNTP, POP3 e HTTP, sendo ainda possível abrir *sockets*[7] e interagir com outros protocolos;
- é gratuito, não é preciso ter licença para utilizá-lo, além de suportar múltiplas plataformas (Windows, Linux, Unix).

[4] *Websites* – páginas na *Internet* onde se disponibiliza as informações.
[5] Servidor *Web* – local onde as páginas e o banco de dados são armazenados.
[6] Cookies – são informações breves trocadas entre o cliente e servidor.
[7] Sockets - são pontes de conexão entre dois computadores pela *Internet.*

2.3.3 Delphi

O Borland Delphi, é fornecido pela Borland Software Corporation™. É derivado de uma linguagem de nível médio - o Pascal, com muita diferença, pois o Pascal não suporta orientação a objeto. Sendo lançado a primeira versão do Delphi em 1995 foi considerada a melhor ferramenta de desenvolvimento para MS Windows [SON2000].

O Delphi é um ambiente de desenvolvimento de aplicações baseado em componentes que permitem desenvolver poderosas aplicações baseadas em MS Windows. Atualmente o Delphi está disponível na versão 6.0 [SON2000].

Também oferece ferramentas de desenvolvimento, que permitem criar e testar rapidamente o protótipo das aplicações. Sendo orientado a objetos, combina a facilidade de utilização de um ambiente de desenvolvimento visual. Oferece ferramentas de bancos de dados que lhe permitem desenvolver aplicações cliente/servidor e relatórios [SON2000].

O Delphi consiste de vários elementos, ferramentas de *design* e de banco de dados para auxiliar no desenvolvimento de sistemas. Suas principais características segundo Sonnino [SON2000] são:

- construtor visual de interface com o usuário;
- arquitetura baseada em componentes;
- biblioteca de componentes visuais;
- ferramenta RAD[8] (*Rapid Application Development*);
- linguagem orientada a objetos (em Object Pascal);
- gerador de relatórios,
- servidor local baseado em SQL[9] e
- arquitetura Cliente/Servidor.

[8] RAD - ferramenta rápida de aplicação.
[9] SQL – Structured Query Language.

2.3.4 Kylix

Desde 1995 o Delphi era uma ferramenta voltada ao desenvolvimento de programas para o ambiente MS Windows. Mas devido à necessidade de uma ferramenta portável para Linux surgiu então o Kylix [SON2001].

Em julho de 1999, a Borland colocou uma pesquisa na Internet sobre a necessidade de desenvolver-se Delphi para Linux, havendo assim um grande número de interessados nesta nova ferramenta. Alguns meses depois foi então anunciado publicamente o Kylix (Delphi para Linux).

Com o Kylix podem ser criados programas para Linux, da mesma forma que se fazia no MS Windows, com a grande vantagem de que os programas escritos para Windows podem ser portados para o Linux [SON2001].

O Borland Kylix é um ambiente de desenvolvimento visual, permitindo desenvolver aplicações, tanto gráficas como em modo texto, dispondo de várias bibliotecas de componentes [SON2001].

Atualmente o Kylix está disponível em sua versão 1.0. As suas principais características segundo Sonnino [SON2001] são:

- Primeiro ambiente RAD para a plataforma Linux, sendo um ambiente de desenvolvimento baseado em componentes para desenvolvimento em aplicações de GUI[10] (*Graphical User Interface*), *Internet*, banco de dados e servidor *web*.
- Utiliza bancos de dados que trabalham com SQL, tais como: Interbase, DB/2, Oracle e MySQL. São bancos de dados que fornecem conexão cliente/servidor.

2.3.5 MySQL

Em 1995, uma equipe de programação utilizava o mSQL para conectar suas tabelas, usando suas próprias rotinas de baixo nível, mas notaram que o mSQL não era rápido e nem flexível o suficiente. Então, os mesmos resolveram desenvolver uma nova interface SQL surgindo o MySQL [FRI2000].

[10] GUI – Interface Gráfica com o Usuário.

MySQL é um sistema gerenciador de banco de dados, multi-tarefas e robusto. Ele é um dos bancos de dados mais usados para a *Internet*, principalmente por servidores Linux. Por ser gratuito, apresenta uma grande vantagem sobre diversos bancos de dados, entrando em grande disputa neste mercado [DIA2000].

Características do MySQL segundo Stoco [STO2000] e Frigeri [FRI2000]:

- permite um número ilimitado de utilização por usuários simultâneos;
- trabalha em diferentes plataformas;
- pode associar tabelas de banco de dados diferentes em uma mesma consulta;
- fornece um sistema de privilégio de senhas flexível e seguro;
- permite até 32 indexadores por tabela;
- apresenta capacidade de manipulação de tabelas com mais de 50.000.000 registros;
- permite a conexão ao servidor utilizando TCP/IP,
- fornece licença de uso grátis, sendo cobrado apenas o suporte se o fizer necessário.

2.3.6 FAST CASE

Com ajuda de seus alunos, Eber A. Schimtz e Denis S. Silveira professores na UFRJ (Universidade Federal do Rio de Janeiro) desenvolveram a ferramenta FAST CASE e lançaram em 2000 a versão 1.0., que está disponível para cópia no *site* <http://www.nce.ufrj.br/fastcase/ > [SCH2000].

O FAST CASE é uma ferramenta CASE (*Computer Aided Software Engineering*) que suporta um subconjunto da UML (*Unified Modeling Language*) para ser utilizada na análise e projeto orientados a objetos.

O FAST CASE provê todas as funcionalidades necessárias para a construção dos seguintes modelos, segundo Schmitz [SCH2000]:

- de Requisitos – suportando a técnicas de casos de usos;
- de Análise e Projeto – suportando os modelos de classes, interação (diagrama de seqüência) e ciclo de vida dos objetos (diagrama de estados).

Capítulo III

O Software de Gestão Escolar - Contraltdel

Este capítulo apresenta as atividades realizadas durante o desenvolvimento do Software de Gestão Escolar - Contraltdel.

3.1 Análise de Requisitos

Buscou-se através da definição de requisitos especificar a funcionalidade do sistema a ser desenvolvido. Esta tarefa é uma forma de traduzir os "desejos" dos usuários para um documento formal.

3.1.1 Visão Geral do Sistema

O trabalho busca o desenvolvimento de um software que disponibilize a emissão de boletins e possibilite a consulta em terminais ou através de um *site* pela Internet onde alunos e pais possam verificar o andamento escolar.

O trabalho tem por objetivo o desenvolvimento de um sistema que otimize as atividades de matriculas dos alunos, possibilitando a partir de dados inseridos no sistema de cadastro de alunos, séries e áreas de conhecimentos disponibilizar o rendimento escolar durante o ano letivo, fornecendo notas e faltas dos alunos a cada bimestre, bem como o resultado final para sua aprovação na série, possibilitando assim mais rapidez e eficiência na emissão e consulta de boletins.

3.1.2 Documento de Especificação de Requisitos

O objetivo principal deste sistema é o de gerenciar os dados necessários para a consulta e a emissão de boletins, através do acesso ao sistema em terminais ou em um *site* na Internet.

Funcionalidade

Nesta seção serão especificadas as funcionalidades do sistema, ou seja, o que ele deve fazer de acordo com as necessidades do Colégio Decisão. O sistema está dividido em três módulos, descritos à seguir.

- ✓ **Módulo Boletim**

Neste módulo, os funcionários serão os principais usuários. O sistema deve permitir:

a) o cadastro de alunos, de acordo com a matrícula;

b) o cadastro de professores;

c) o cadastro de disciplinas;

d) o cadastro de turmas;

e) o cadastro de notas dos alunos;

f) o cadastro de freqüência dos alunos;

g) a emissão de boletins;

h) a emissão de lista de alunos para os professores lançar as notas e faltas.

- ✓ **Módulo Aluno**

Este módulo será disponibilizado em terminais no Colégio Decisão, para uso dos alunos. O sistema deve permitir que, a partir do código de matrícula do aluno, o mesmo possa consultar o seu boletim.

- ✓ **Módulo *Site***

Este módulo é semelhante ao módulo anterior, mas será disponibilizado em um *site* na *Internet*, permitindo que os alunos consultem seu boletim de maneira *on-line*. O sistema deve permitir que, a partir do código de matrícula do aluno, o mesmo possa consultar o seu boletim.

Usabilidade

Todos os módulos serão de fácil entendimento, pois suas interfaces utilizam ambiente gráfico (tanto em MS Windows como em Linux) facilitando a interatividade do usuário com o sistema.

Confiabilidade

Esta seção leva em consideração a confiabilidade do sistema e sua segurança.

- ✓ **Módulo Boletim**

Este módulo permitirá que o FUNCIONÁRIO tenha os privilégios de consulta, inserção, alteração e exclusão dos dados.

- ✓ **Módulos Aluno e Site**

Estes módulos serão de alta confiabilidade, pois qualquer acesso será apenas para consultas, sendo que os usuários não poderão manipular os dados.

- ✓ **Banco de Dados**

O banco de dados estará armazenado em um servidor, onde o sistema poderá manipular seus dados de maneira segura. As senhas de acesso ao banco de dados são de conhecimento apenas de seus desenvolvedores. Como essas senhas são aplicadas diretamente no sistema, não se faz necessário que outros usuários as conheçam.

Eficiência

Nesta seção será apresentada a eficiência do sistema.

- ✓ **Módulo Boletim**

Neste módulo o sistema deve ser eficiente em seu cadastro de notas e freqüências, permitindo que o funcionário os faça sem ter que recorrer a vários métodos, e permitindo que este serviço seja feito por mais de um funcionário ao mesmo tempo.

- ✓ **Módulo *Site***

Neste módulo, o sistema deve ser eficiente na busca das informações e na sua interface, pois os serviços de *Internet* fazem busca de arquivos de texto e imagens, e dependendo do tamanho dos mesmos, podem trazer frustração para quem está acessando o *site*.

Portabilidade

Nesta seção será definida a portabilidade do sistema: o sistema operacional, banco de dados, configuração do computador e sua instalação.

- ✓ **Banco de Dados**

O banco de dados utilizado para armazenar os dados do sistema é o MySQL, o qual estará apto à funcionar em MS Windows, Linux ou Unix, desde que seja possível disponibilizar uma porta para comunicação com o banco de dados.

De momento a versão mais adequada ao sistema é o MySQL 3.22.32 devido sua alta compatibilidade com clientes desenvolvidos em Kylix 1.0 – uma das ferramentas utilizadas para desenvolvimento em ambiente Linux.

Para sistemas clientes em MS Windows/Unix é necessário o pacote MyODBC para realizar a conexão. Já para clientes Linux o pacote é o MySQL-Client, a mesma versão do MySQL utilizado. Quanto à internet, não há necessidade que seus usuários utilizem MySQL-Client, tendo em vista que o servidor fará esta conexão transparente ao internauta.

A configuração mínima do computador deve ser avaliada quanto ao número de clientes que vão acessá-lo ao mesmo tempo e quais os tipos de serviços eles estarão fazendo.

✓ **Módulos Boletim e Aluno**

Estes sistemas desenvolvidos em Object Pascal, estão aptos a funcionarem tanto em MS Windows utilizando Borland Delphi como em Linux utilizando Borland Kylix. Os sistemas operacionais deverão ter configuração mínima de:

- MS Windows: 95/98/ME, mas preferencialmente NT/2000.
- Linux com libc 2.1.2 ou posterior, kernel 2.2 ou posterior e libjpeg 6.2 ou posterior.

A configuração mínima do computador para utilizar o Módulo Boletim será de 128 MB de RAM, processador 350 Mhz, e 10 GB de HD. Rodando os sistemas operacionais Linux ou MS Windows 2000.

Configuração mínima do computador para utilizar o módulo aluno será de 64 MB de RAM, processador 233 Mhz, e 4 GB de HD, podendo utilizar Linux ou MS Windows 95/98/ME/NT/2000.

✓ **Módulo *Site***

Este módulo está apto à rodar em um servidor *web*, seja ele MS Windows, Linux ou Unix, desde que o interpretador de *scripts* PHP e o banco de dados MySQL, estejam instalados e sejam compatíveis com o sistema. O servidor deverá ter:

- link direto com a Telecom 24 horas por dia;
- permitir mais de 50 MB de armazenamento em disco;
- PHP 4.0.0 ou superior instalado;
- MySQL 3.22.32 ou superior instalado;
- cópias de segurança.

Para que o aluno possa de seu navegador fazer uma acesso direto à página do Colégio, será necessário um domínio que poderá ser registrado no *site* Registro.br.

3.2 Análise do Sistema

Buscou-se a partir da análise orientada a objetos, planejar um conjunto de atividades e procedimentos, aliado aos métodos e a documentação específica de cada fase, que permitissem a construção de um software com qualidade.

3.2.1 Casos de Uso

Um caso de uso é a descrição de um curso completo de eventos, iniciado por um ator, demonstrando sua interação com o sistema. É a organização do fluxo de operação em pequenos blocos coesos podendo ser facilmente visualizado e alterado. A figura 3.1 apresenta o caso de uso Principal do sistema, a figura 3.2 apresenta o caso de uso Mantendo Informações, a figura 3.3 apresenta o caso de uso Matriculando Aluno, a figura 3.4 apresenta o caso de uso Lançando Notas e Faltas e a figura 3.5 apresenta o caso de uso Consultando Boletim.

As definições dos casos de uso Mantendo Informações, Matriculando Aluno, Lançando Notas e Faltas e Consultando Boletim também são apresentadas. Os demais casos de uso serão apresentado no apêndice A.

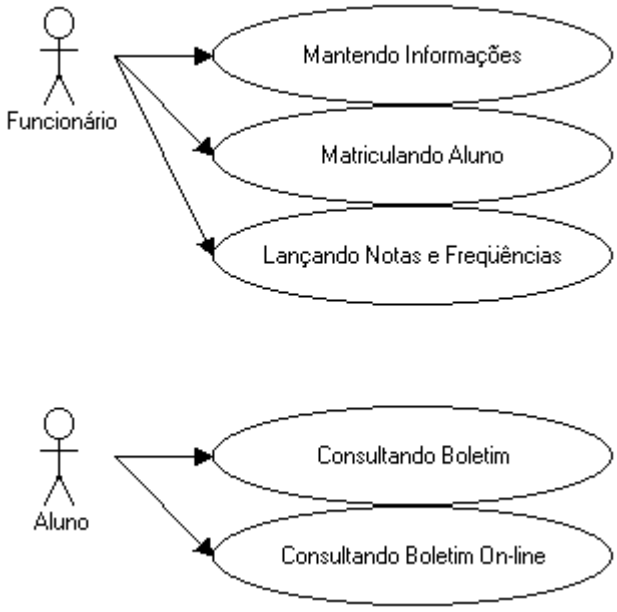


Figura 3.1 – Caso de Uso Principal

DEFINIÇÃO DO CASO DE USO - Mantendo Informações

ATOR(ES): Funcionário

PRÉ-CONDIÇÃO:

1 - O usuário precisa manipular informações do sistema.

ROTEIRO:

1 - O usuário pode optar pelos casos (uses) de #Mantendo Informações#.
2 - O usuário pode encerrar este caso.

PÓS-CONDIÇÃO:

1 - O usuário manipulou as informações do sistema.

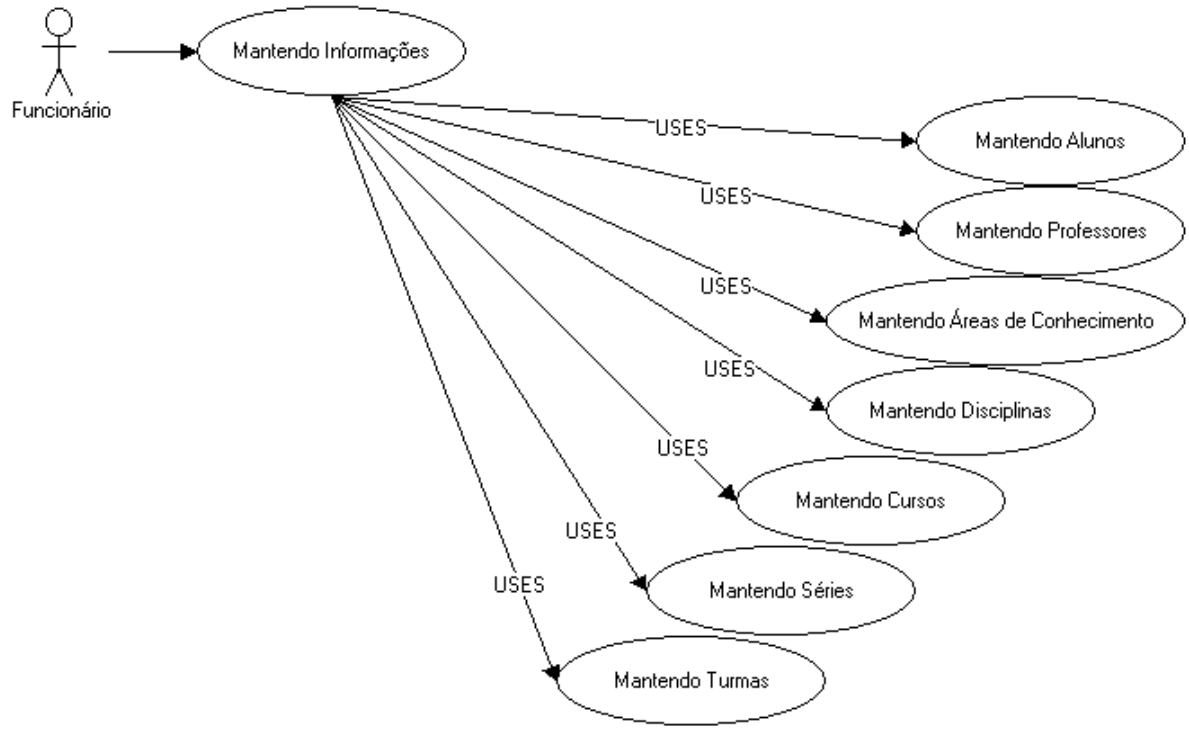


Figura 3.2 – Caso de Uso Mantendo Informações

DEFINIÇÃO DO CASO DE USO - Matriculando Aluno

ATOR(ES): Funcionário

PRÉ-CONDIÇÃO:

1 - O usuário quer matricular um aluno já cadastrado no sistema.

ROTEIRO:

1 - O usuário deve selecionar o aluno no caso #Selecionando Aluno#.
2 - O usuário deve selecionar a turma no caso #Selecionando a Turma#
3 - O sistema gera o código da matrícula automático.
4 - O caso pode ser encerrado.

PÓS-CONDIÇÃO:

1 - O usuário efetivou a matrícula do aluno.

EXCEÇÃO:

1a - O aluno está matriculado: o sistema deve informar que existe uma matricula efetuada para o aluno.

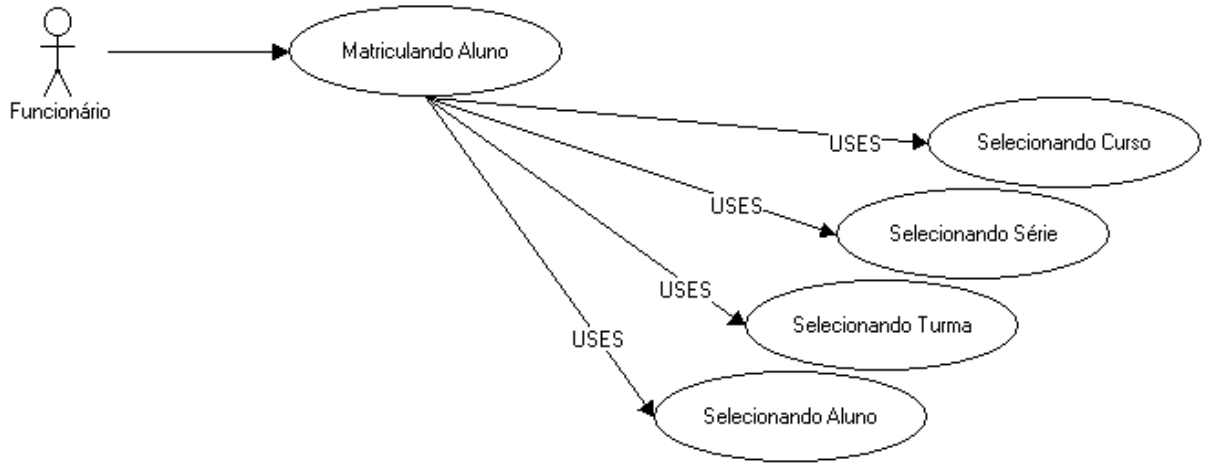


Figura 3.3 – Caso de Uso Matriculando Aluno

DEFINIÇÃO DO CASO DE USO - Lançando Notas e Freqüências

ATOR(ES): Funcionário

PRÉ-CONDIÇÃO:

1 - O professor fornece os dados de notas e freqüências para o usuário lançar as informações no sistema.

ROTEIRO:

1 - O usuário deve selecionar a turma usando #Selecionando Turma#.
2 - O usuário deve selecionar a disciplina usando #Selecionando Disciplina# .
3 - O usuário seleciona por *RadioGroup* o período que serão lançadas as notas. (1º, 2º, 3º, 4º Bimestre ou Exame).
4 - O usuário poderá através de uma *ListView* selecionar o aluno e lançar suas notas e faltas.
5 - O usuário poderá confirmar as notas e faltas digitadas.
6 - O usuário pode encerrar este caso.

PÓS-CONDIÇÃO:

1 - O usuário efetuou o lançamento de notas e freqüências.

EXCEÇÃO:

5a - Erro nos valores da *ListView*: O sistema deve informar quais foram os erros possíveis ((erro no valor da nota, negativa ou acima de 100) ou erro na quantidade de faltas), seleciona a linha da *ListView* com erro e volta ao passo 4.

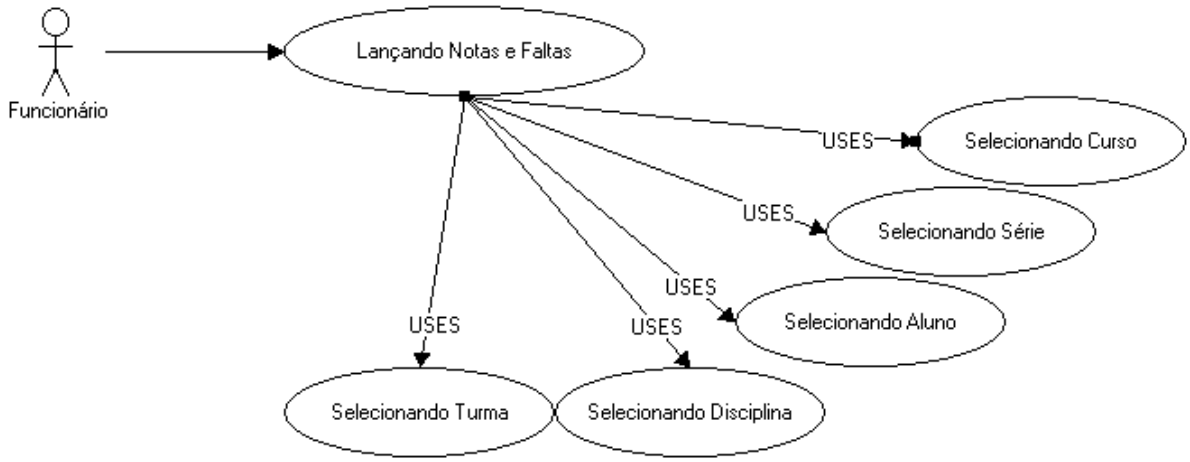


Figura 3.4 – Caso de Uso Lançando Notas e Faltas

DEFINIÇÃO DO CASO DE USO - Consultando Boletim

ATOR(ES): Aluno

PRÉ-CONDIÇÃO:

1 - O aluno matriculado no colégio quer ver suas notas e freqüências (boletim).

ROTEIRO:

1 - O usuário deve preencher sua matrícula no caso #Selecionando Matrícula#.
2 - O usuário pode visualizar as notas e freqüências em todos os períodos nesta área de conhecimento com detalhes das disciplinas.
3 - O usuário pode voltar ao passo 2 ou encerrar o caso.

PÓS-CONDIÇÃO:

1 - O usuário pesquisou notas e freqüências (boletim) do aluno matriculado.

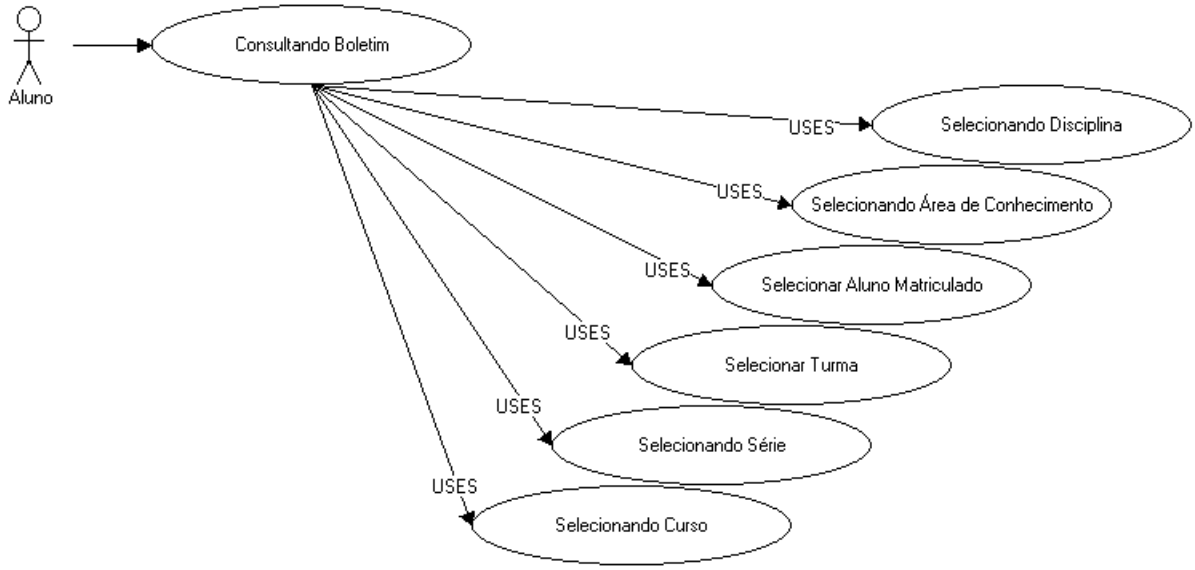


Figura 3.5 – Caso de Uso Consultado Boletim

3.3 Desenvolvimento do Sistema

Após a identificação dos requisitos do sistema e o desenvolvimento dos casos de uso, buscou-se identificar como os objetos interagem na execução de cada um dos casos de uso, através de diagramas de classe e entidade relacionamento, que definem a interação com o sistema e diagramas de seqüência para mostrar a troca de serviços entre os objetos participantes do sistema.

3.3.1 Diagrama de Classe

As classes representam o empacotamento de uma estrutura de dados com suas rotinas de acesso e constituem uma forma natural de modularizar o sistema. Um sistema bem modularizado apresenta alta coesão interna e um baixo acoplamento com outros módulos [SCH2000]. A figura 3.6 apresenta o diagrama de classes do sistema.

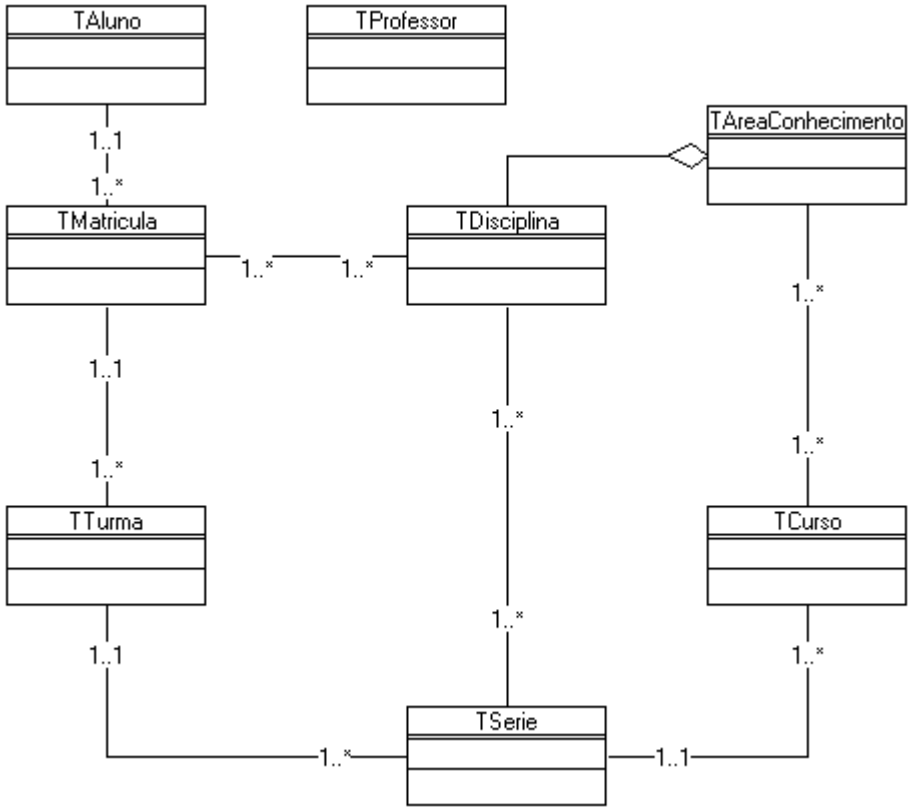


Figura 3.6 – Diagrama de Classe

3.3.2 Diagrama de Seqüência

Os diagramas de seqüência mostram a troca de serviços entre os objetos participantes de cada pacote do sistema. É durante esta etapa que os Casos de Uso, escritos em linguagem natural, serão traduzidos em uma seqüência de mensagens trocadas entre os objetos participantes dos casos de uso. A figura 3.7 mostra o diagrama de seqüência Matriculando Aluno, figura 3.8 mostra o diagrama de seqüência Lançando Notas e Falas e a figura 3.9 mostra o diagrama de seqüência Consultando Boletim. Os demais diagramas de seqüência do sistema serão apresentados no apêndice B.

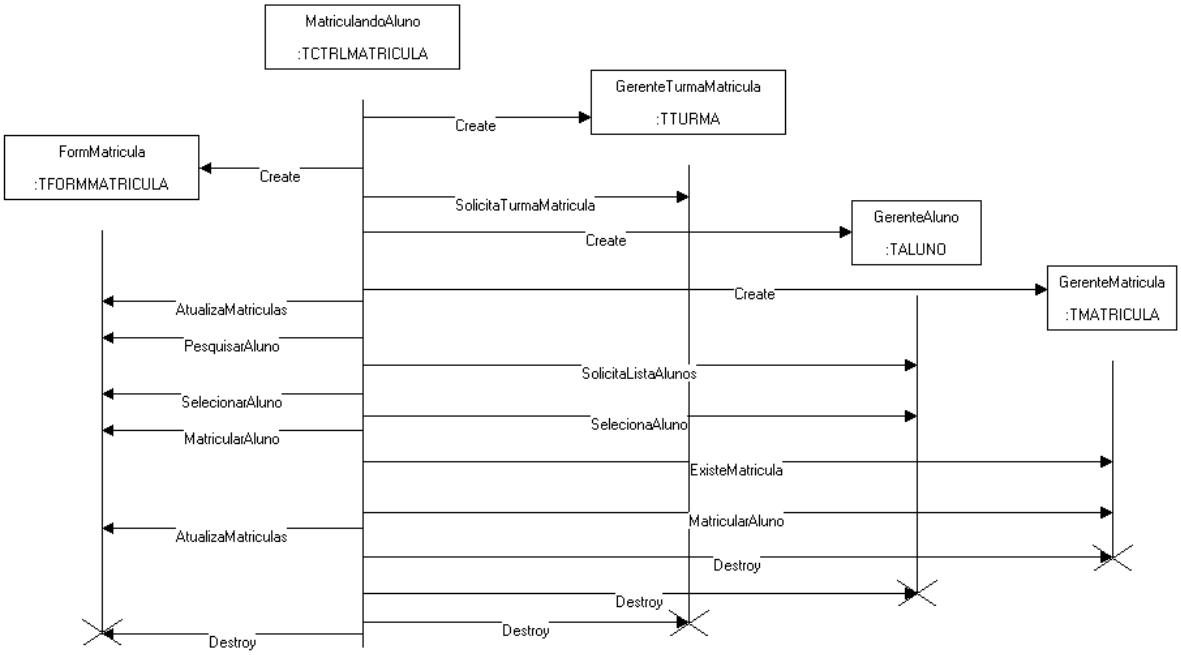


Figura 3.7 – Diagrama de Seqüência Matriculando Aluno

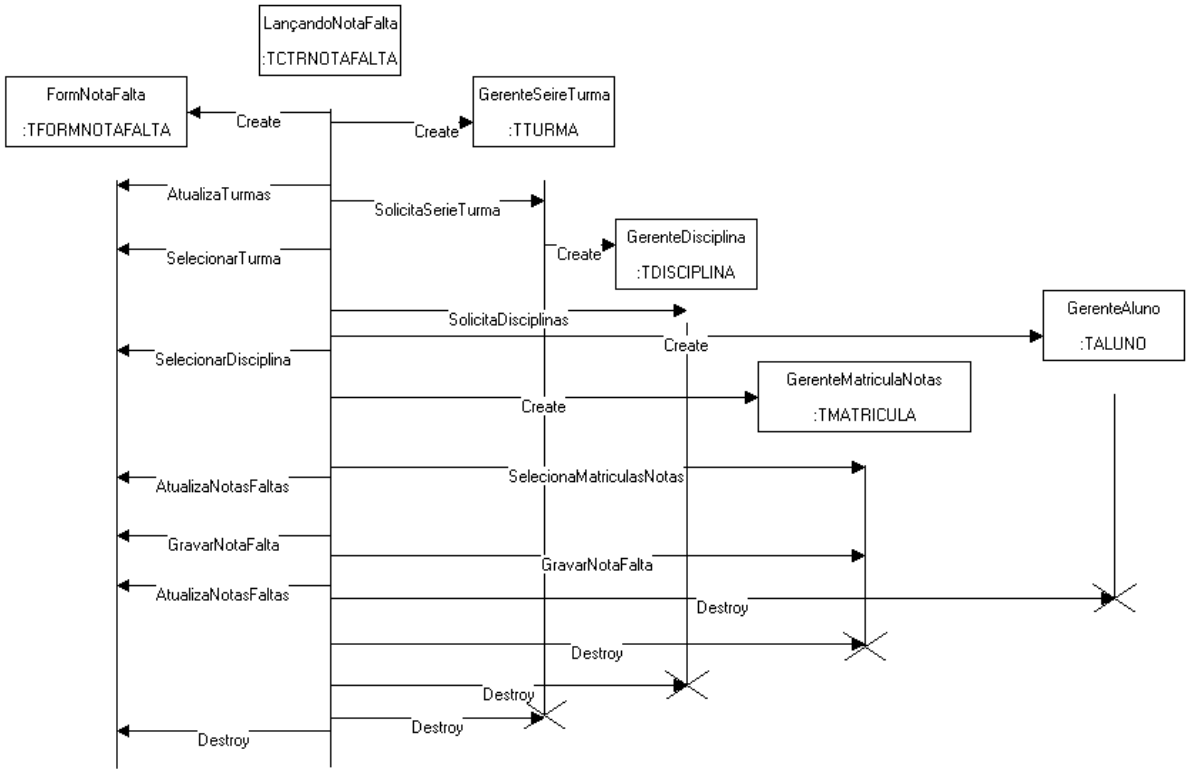


Figura 3.8 – Diagrama de Seqüência Lançando Notas e Faltas

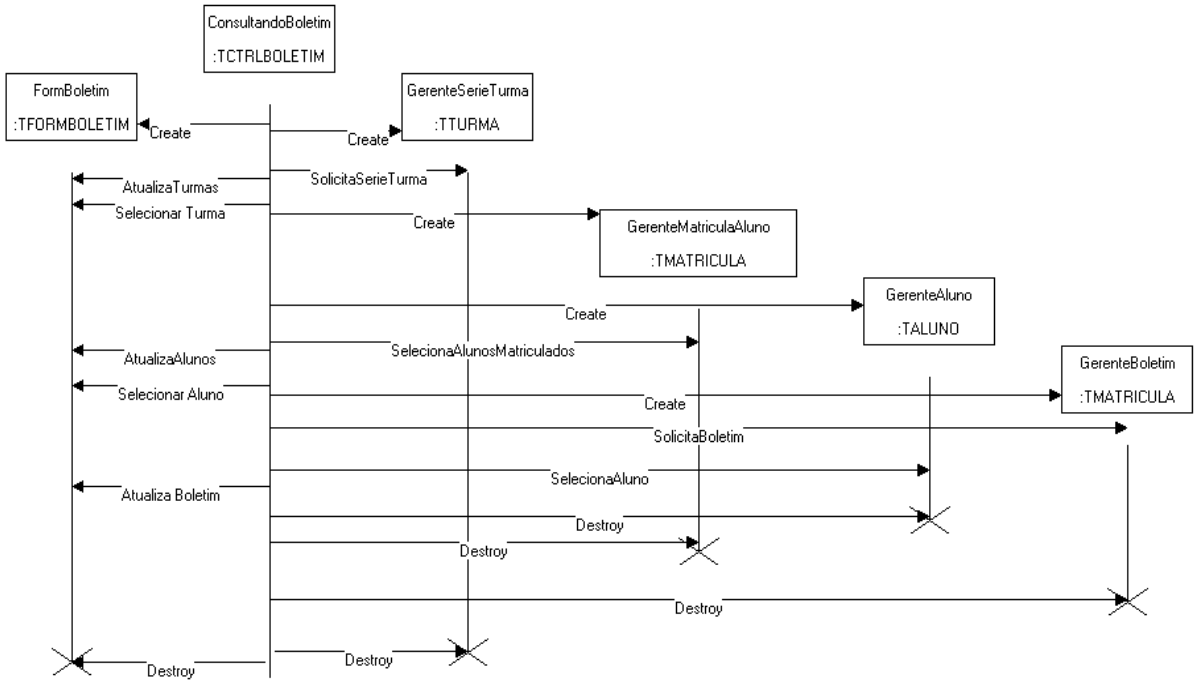


Figura 3.9 – Diagrama de Seqüência Consultando Boletim

3.3.3 Diagrama Entidade Relacionamento

É um diagrama voltado para descrição gráfica das entidades e seus relacionamentos, caracterizando uma visão lógica de alto nível dos dados do sistema. A figura 3.10 apresenta o diagrama Entidade Relacionamento do sistema.

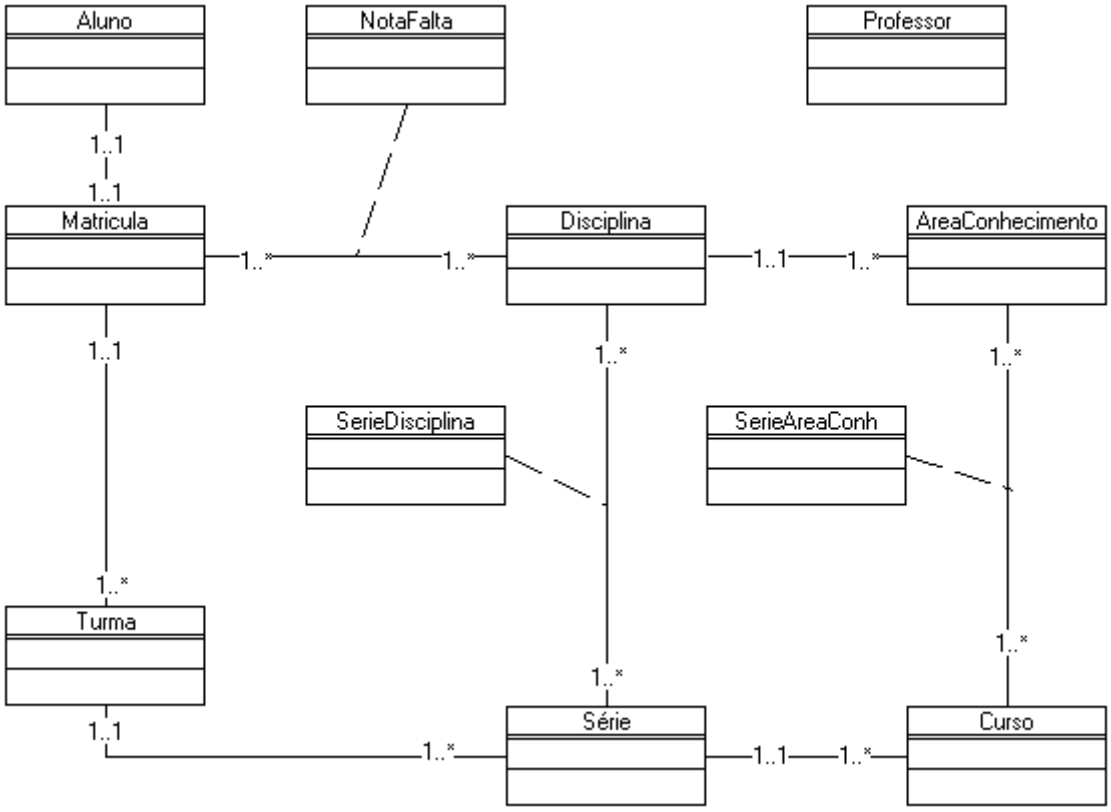


Figura 3.10 – Diagrama Entidade Relacionamento

3.3.4 Codificação

A codificação do sistema foi realizada em Object Pascal, utilizando Borland Delphi para MS Windows, Borland Kylix para Linux e banco de dados MySQL, estando sua versão final disponível para as duas plataformas.

Não utilizou-se de APIs para que o sistema seja altamente portável. Para o desenvolvimento dos relatórios utilizou-se de componentes próprios das ferramentas.

A integridade do sistema não permite que cursos, séries e turmas que possuam alunos matriculados, áreas de conhecimento e disciplinas com notas e faltas lançadas, sejam excluídos.

O sistema fornece o recurso de inclusão de fotos de alunos e professores. As fotos dos alunos podem ser visualizadas nos formulários Matricular Alunos, Lançar Notas e Faltas e Consultar Boletim no Módulo Boletim.

O sistema está desenvolvido em uma arquitetura de 4 camadas, influenciando diretamente em duas características importantes do software que são a facilidade de manutenção e capacidade de reutilização.

As camadas estão divididas em duas partes: uma chamada de **interface** e a segunda de **implementação.** A camada de interface presta o serviço de interação com o usuário final, utilizada para delegar, ao controle as tarefas solicitadas. A camada de implementação ainda é dividida entre a camada de **classe** que realiza os procedimentos dos objetos de negócio do sistema, sendo tratados todos os erros referentes à validação dos dados, campo nulo, valor esperado maior que o valor digitado ou valor esperado menor que o valor digitado e a camada **gerente** responsável por realizar a conexão e os procedimentos necessários para a comunicação com o banco de dados, são realizadas todas as consultas SQL para que posteriormente os dados sejam fornecidos a camada de classe. Estas camadas ficam totalmente invisíveis ao usuário final.

Para fazer a ligação das camadas de interface e implementação utilizou-se uma outra chamada de camada de **controle,** que realiza a chamada dos métodos e procedimentos entre a camada de interface e a de implementação, sendo está camada responsável pela criação e destruição de todos objetos de interfaces, classes e gerentes.

3.4 Projeto de Interface

A interface é a embalagem do *software* de computador, sendo o mecanismo por meio do qual se estabelece um diálogo entre o programa e o ser humano. Se for fácil de aprender, simples de usar, apresentando uma interface direta e amigável, o usuário estará inclinado a fazer bom uso do software [PRE95].

> "Frustrações e ansiedade fazem parte da vida diária de muitos usuários de sistemas de informação computadorizados. Eles lutam para aprender a linguagem de comandos ou sistemas de seleção por menu, os quais, presume-se, devem ajudá-los [PRE95]."

3.4.1 Módulo Boletim

A figura 3.11 mostra a interface do formulário Matricular Alunos. Este formulário é responsável por realizar as matrículas dos alunos cadastrados no sistema. Através do botão Pesquisar, faz-se a consulta aos alunos cadastros. Ao selecionar um aluno, escolhe-se a turma na qual o mesmo será matriculado. Todos os alunos matriculados nas turmas e o número de matriculas existentes no ano letivo correspondente, são visualizados em uma *TreeView*.

Figura 3.11 – Matricular Alunos

A figura 3.12 mostra a interface do formulário Lançar Notas e Faltas. Neste formulário pode-se realizar o lançamento das notas e faltas nas disciplinas de uma turma.

Após selecionar-se a turma e a disciplina serão listados todos os alunos da turma, para que o lançamento das notas e faltas possa ser efetivado. Pode-se lançar as notas e faltas em todos os bimestres simultaneamente ou selecionando-se os períodos em que serão realizados os lançamentos (1º, 2º, 3º, 4º Bimestre ou Exame).

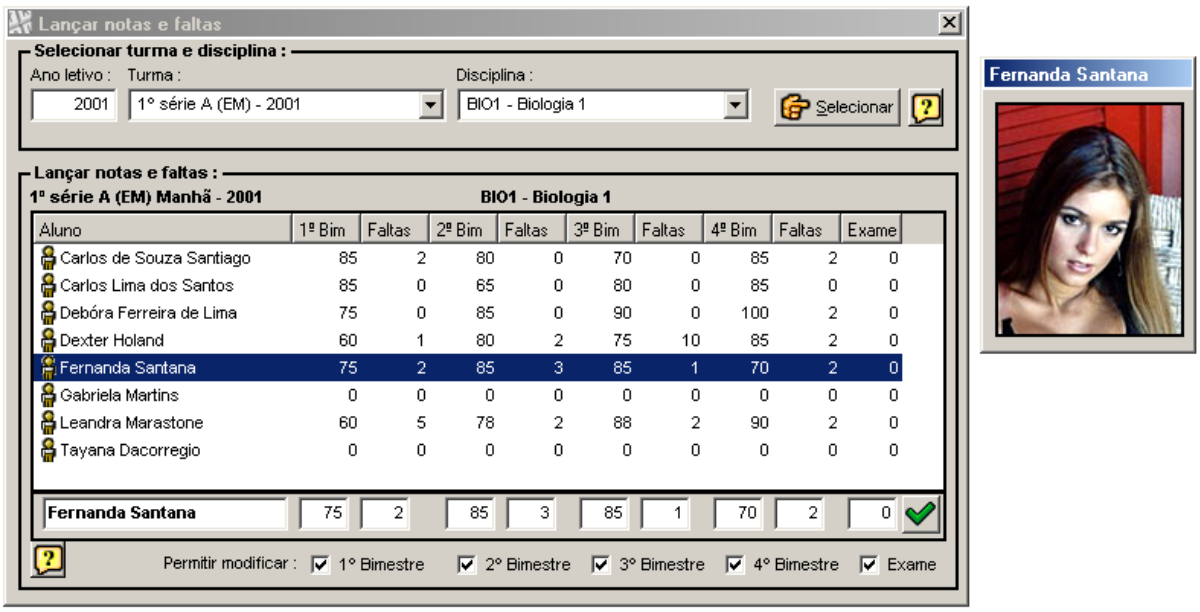


Figura 3.12 – Lançar Notas e Faltas

A figura 3.13 apresenta a interface do formulário Consultar Boletim. Neste formulário pode-se visualizar as notas e faltas dos alunos dentro das áreas de conhecimento e disciplinas, sendo o mesmo utilizado somente pelo funcionário. Após selecionar-se a turma e o aluno serão listadas todas as áreas de conhecimento, cada uma com suas respectivas notas e faltas, sendo possível a visualização das notas e faltas obtidas nas disciplinas, média final, faltas, percentual de freqüência e o resultado final aprovado ✚ ou reprovado ▬ na área de conhecimento.

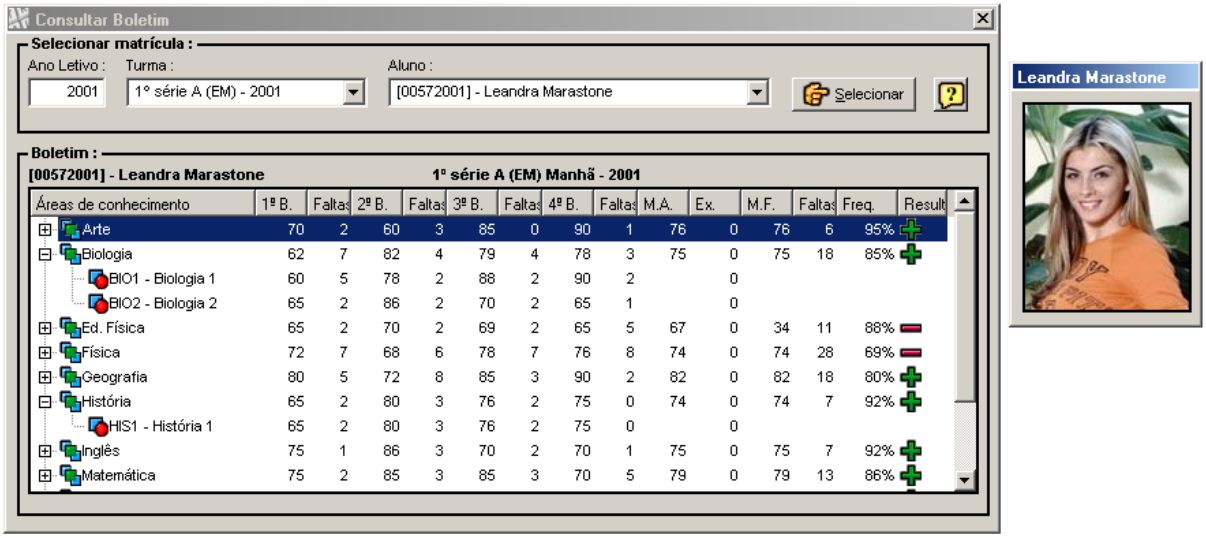
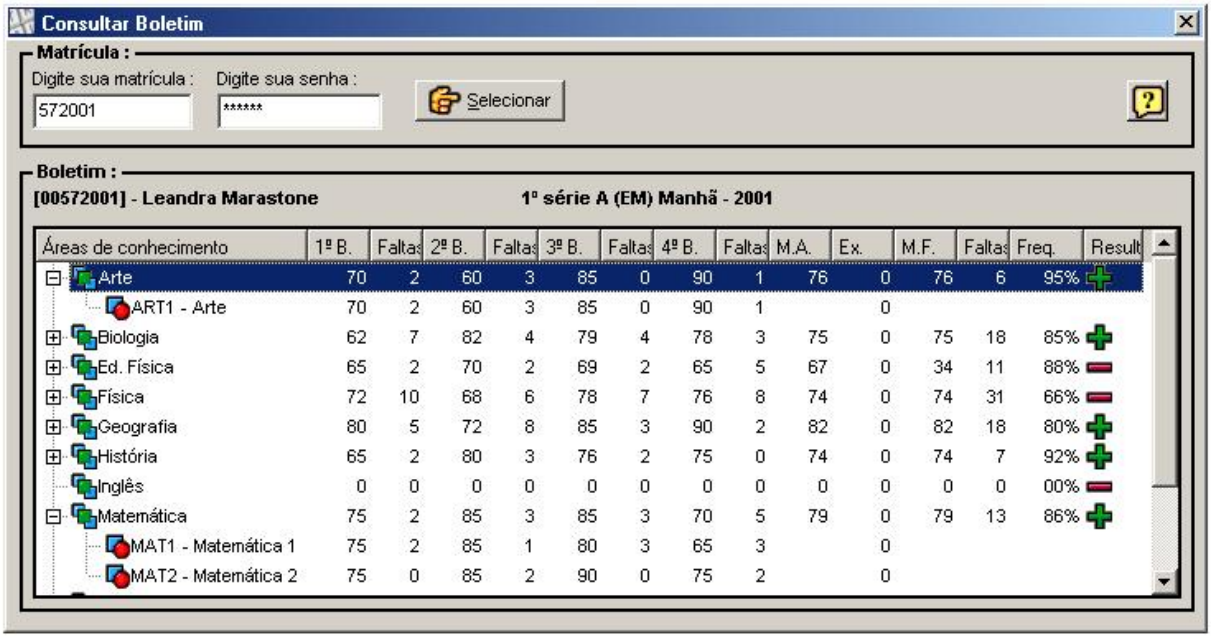


Figura 3.13 – Consultar Boletim – Módulo Boletim

3.4.2 Módulo Aluno

A figura 3.14 apresenta a interface do formulário Consultar Boletim. Este formulário é responsável por realizar a consulta do boletim dos alunos matriculados. Este módulo é utilizado para consultas de notas e freqüências nos terminais de computadores instalados no próprio colégio.

Após digitar-se o número da matrícula e repeti-lo no campo senha, o aluno visualiza suas notas e faltas nas áreas de conhecimento, sendo possível consultar as notas e faltas obtidas nas respectivas disciplinas da área de conhecimento.



Áreas de conhecimento	1ª B.	Faltas	2ª B.	Faltas	3ª B.	Faltas	4ª B.	Faltas	M.A.	Ex.	M.F.	Faltas	Freq.	Result
Arte	70	2	60	3	85	0	90	1	76	0	76	6	95%	
ART1 - Arte	70	2	60	3	85	0	90	1		0				
Biologia	62	7	82	4	79	4	78	3	75	0	75	18	85%	
Ed. Física	65	2	70	2	69	2	65	5	67	0	34	11	88%	
Física	72	10	68	6	78	7	76	8	74	0	74	31	66%	
Geografia	80	5	72	8	85	3	90	2	82	0	82	18	80%	
História	65	2	80	3	76	2	75	0	74	0	74	7	92%	
Inglês	0	0	0	0	0	0	0	0	0	0	0	0	00%	
Matemática	75	2	85	3	85	3	70	5	79	0	79	13	86%	
MAT1 - Matemática 1	75	2	85	1	80	3	65	3		0				
MAT2 - Matemática 2	75	0	85	2	90	0	75	2		0				

Figura 3.14 – Consultar Boletim – Módulo Aluno

3.4.3 Módulo *Site*

No Módulo *Site* pode-se realizar a consulta de boletim via Internet. Na página principal (apresentada na figura 3.15 (A)), faz-se necessário que o aluno forneça o número da matrícula no campo Matrícula, e repita-o no campo Senha. Solicita-se a confirmação dos dados, através de uma nova página (apresentada na figura 3.15 (B)) onde se visualiza as notas e faltas obtidas nas áreas de conhecimento e suas respectivas disciplinas, exame, média final, total de faltas, percentual de freqüência, e resultado final aprovado (A) e reprovado (R).

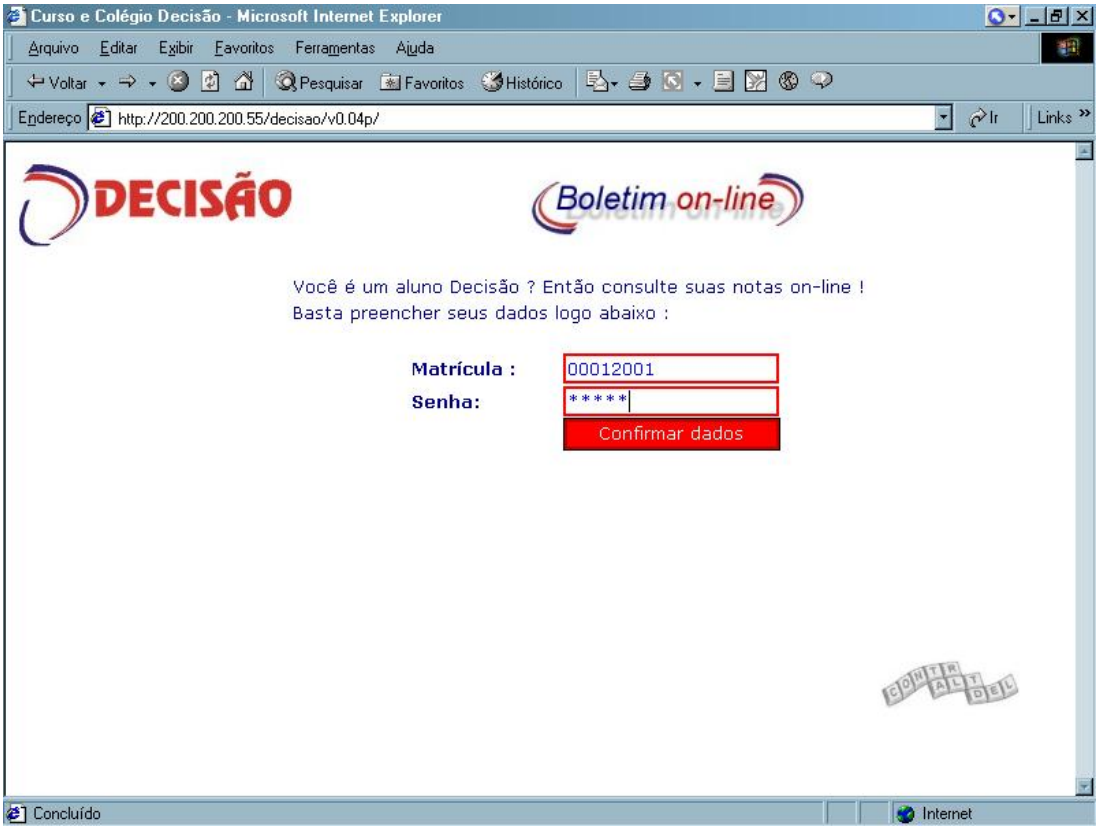
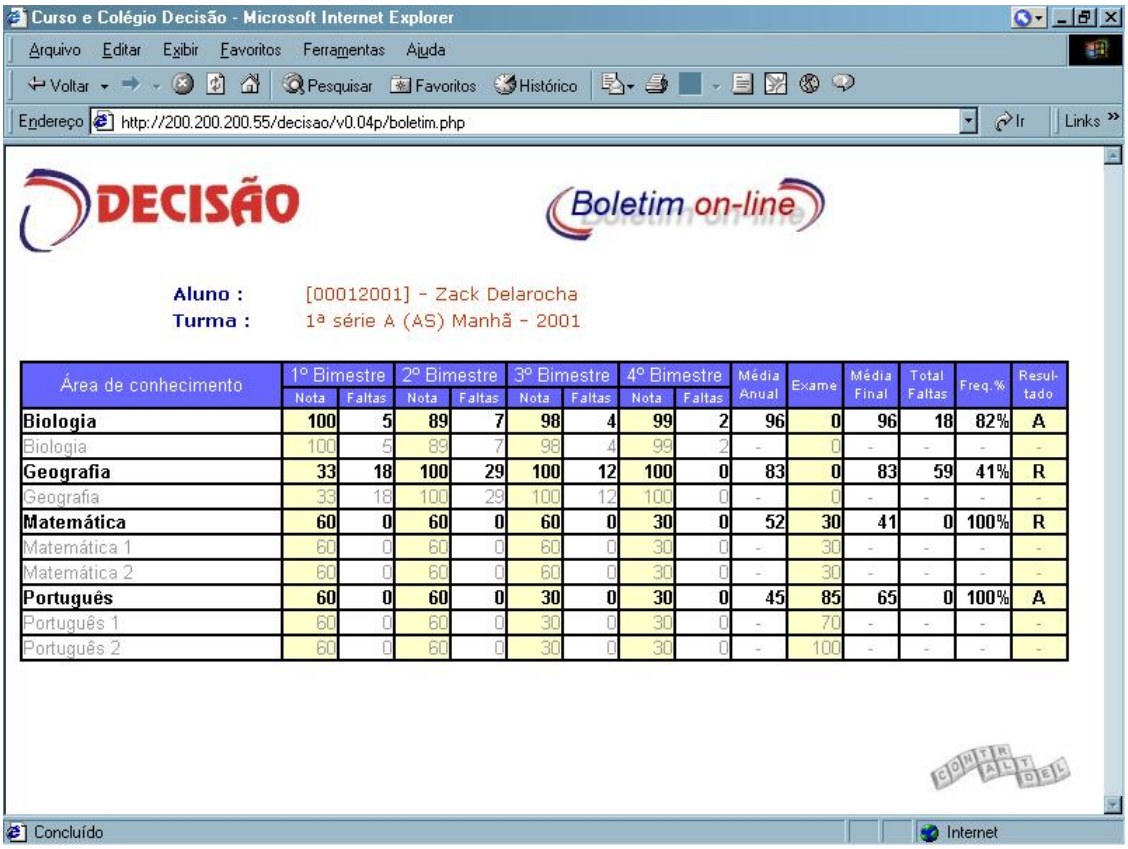


Figura 3.15 (A) – Consultar Boletim – Módulo *Site*



Área de conhecimento	1º Bimestre		2º Bimestre		3º Bimestre		4º Bimestre		Média Anual	Exame	Média Final	Total Faltas	Freq.%	Resultado
	Nota	Faltas	Nota	Faltas	Nota	Faltas	Nota	Faltas						
Biologia	100	5	89	7	98	4	99	2	96	0	96	18	82%	A
Biologia	100	5	89	7	98	4	99	2	-	0	-	-	-	-
Geografia	33	18	100	29	100	12	100	0	83	0	83	59	41%	R
Geografia	33	18	100	29	100	12	100	0	-	0	-	-	-	-
Matemática	60	0	60	0	60	0	30	0	52	30	41	0	100%	R
Matemática 1	60	0	60	0	60	0	30	0	-	30	-	-	-	-
Matemática 2	60	0	60	0	60	0	30	0	-	30	-	-	-	-
Português	60	0	60	0	30	0	30	0	45	85	65	0	100%	A
Português 1	60	0	60	0	30	0	30	0	-	70	-	-	-	-
Português 2	60	0	60	0	30	0	30	0	-	100	-	-	-	-

Figura 3.15 (B) – Consultando Boletim – Módulo *Site*

Os demais formulários do sistema serão apresentados no Apêndice C.

3.3.4 Relatório do Sistema

A elaboração dos relatórios foi realizada utilizando-se componentes próprios das ferramentas Delphi e Kylix, sendo assim os conceitos de camadas também foram utilizados no desenvolvimento dos relatórios. A emissão de boletim é o principal relatório do sistema. Na próxima página apresenta-se um boletim impresso.

Capítulo IV
Conclusões

Este capítulo apresenta as conclusões do trabalho de estágio e a sugestão de trabalhos futuros.

4.1 Considerações Finais

Diante de um mercado de trabalho competitivo e presenciando a diminuição maciça do trabalho formal, percebe-se que a criatividade e busca de informação, são poderosas ferramentas que um analista de sistemas deverá ter para obter sucesso profissional.

É neste contexto que o estágio supervisionado está inserido: o de promover a aplicação dos conhecimentos adquiridos durante a graduação. Soma-se a isso a possibilidade de crescer como pessoa e como profissional. É um período de erros e acertos. Nesta etapa de aprendizagem o mais difícil é aprender a trabalhar com o medo, e em alguns momentos a gerenciar frustrações.

O estágio, além disso, proporciona uma visão de mercado, tanto no conhecimento da criação de softwares que atendam os padrões de aceitação, quanto das necessidades reais do mercado, que envolvem o domínio de tecnologias e ferramentas.

4.2 Trabalhos futuros

Como parte de uma primeira experiência no desenvolvimento de software, o Sistema de Gestão Escolar ainda pode ser melhorado e expandido:

- O *site,* desenvolvendo-se uma página de apresentação do Colégio Decisão com *links* para consulta de boletim, horários de aulas, provas, eventos importantes e disponibilização de contas de *e-mails*.
- O sistema com acoplamento de novos módulos para biblioteca, cobrança de mensalidade, e cadastro de horários de aulas.

Referências Bibliográficas

[AMB97] Amber, Scott W. – *Análise e Projeto Orientados a Objeto* – Editora Infobook, Rio de Janeiro – RJ, 1997.

[DIA2000] Dias, Adilson de Souza - *Delphi & MySQL* – Editora Ciência Moderna, Rio de Janeiro – RJ, 2000.

[FRI2000] Frigeri, Sandra Rovena – *MySQL Introdução*, disponível em <http://www.ucs.br/formacao/semestre004/PesquisaArquivos/MySql/intro.htm>, desde 2000. Acesso em 01/05/2001.

[FUR98] Furlan, José Davi – *Modelagem de Objetos através da UML* – Editora Makron Books, São Paulo – SP, 1998.

[HOR97] Hort, HTML – *A Linguagem que mudou o... Mundo!*, disponível em <http://www.terravista.pt/BaiaGatas/1144/>, desde 1997. Acesso em 30/04/2001.

[MED2001] Medeiros Jr, Roberto - *Esta é a idéia! Banco de Dados*, disponível em <http://www.yeh.hpg.com.br/info/bd/mod.html>, desde 2001. Acesso em 24/06/2001.

[NEV2000] Neves, Anderson de Oliveira - *Cliente/Servidor,* disponível em <http://www1.cptec.inpe.br/~anderson/apost/info/paper_07.htm>, desde 2000. Acesso em 24/06/2001.

[PRE95] Pressman, Roger S. - *Engenharia de Software* - Editora Makron Books do Brasil, São Paulo - SP, 1995.

[RIT96] Ritchey,Tim - *Programando Java & JavaScript para Netscape 2.0* - Quark Editora, São Paulo – SP, 1996.

[ROC2000] Rocha, Ana Célia - *História do PHP*, disponível em <http://www.cgi-net.com.br/php/historia.shtml>, desde 2000. Acesso em 11/04/2001.

[SCH2000] Schmitz, Eber A., Silveira, Denis S. – *Desenvolvimento de Software Orientado a Objetos Utilizando UML e Delphi 5* – Editora Brasport, Rio de Janeiro – RJ, 2000.

[SON2000] Sonnino, Bruno - *Desenvolvendo Aplicações com Delphi 5* – Editora Makron Books do Brasil, São Paulo – SP, 2000.

[SON2001] Sonnino, Bruno - *Kylix – Delphi para Linux* - Editora Makron Books do Brasil, São Paulo - SP, 2001.

[STO2000] Stoco, Lucio M. - *Integrando PHP com MySQL - Guia de Consulta*

Rápida - Editora Novatec, São Paulo – SP, 2000.

[SUR97] Surian, Jorge e Nicochelli, Luiz - *Apostila de Banco de Dados e SQL,* disponível em <http://www.geocities.com/SouthBeach/Marina/ 2164/ delphi/documentos/docto6.html>, desde 1997. Acesso em: 24/06/2001.

[TAN97] Tanenbaum, Andrew S. - *Redes de Computadores – Tradução da Terceira Edição* - Editora Campus, Rio de Janeiro – RJ, 1997.

Casos de Uso

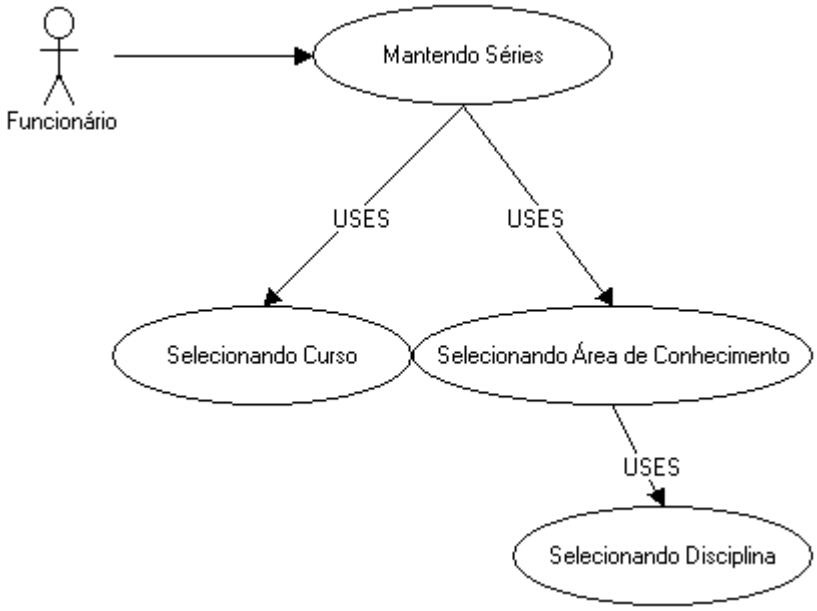


Figura A.1 – Caso de Uso Mantendo Série

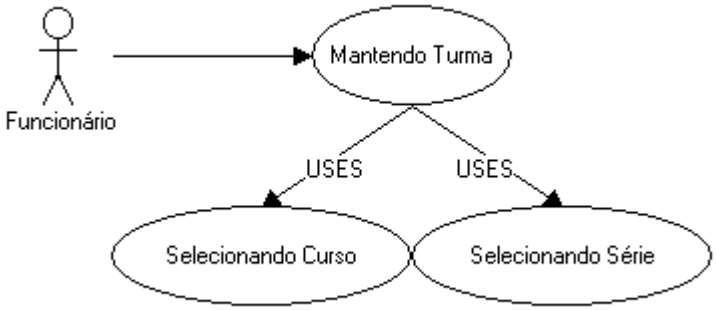


Figura A.2 – Caso de Uso Mantendo Turma

Diagramas de Seqüência

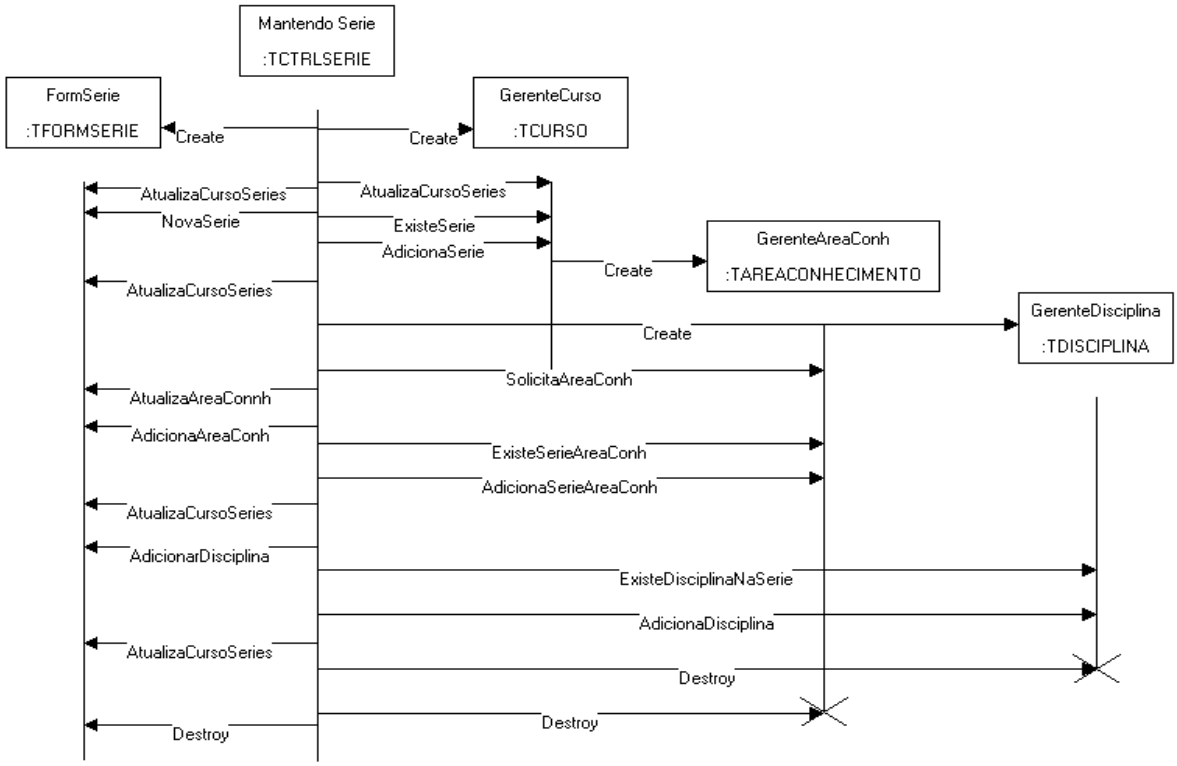


Figura B.1 – Diagrama de Seqüência Mantendo Série

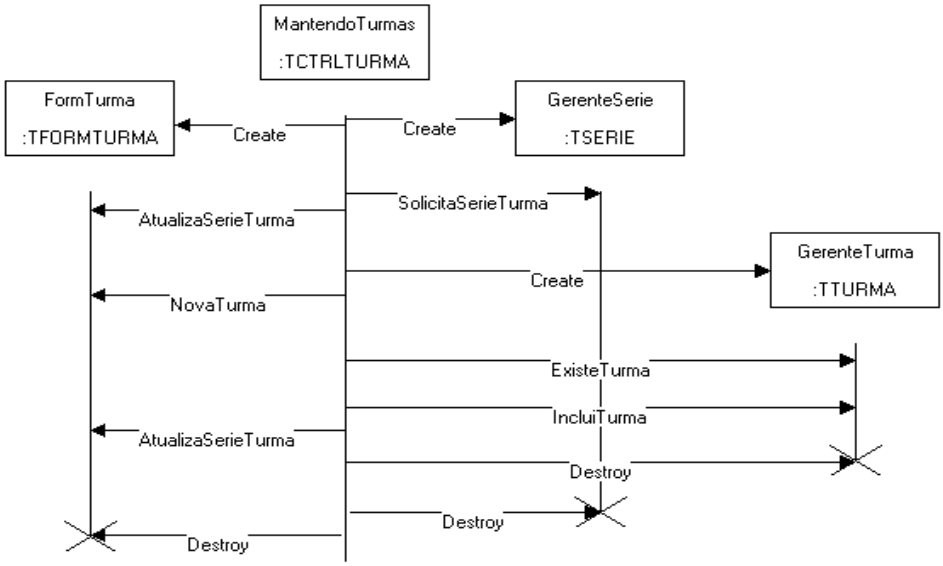


Figura B.2 – Diagrama de Seqüência Mantendo Turma

Formulários do Sistema

Figura C.1 – Formulário de Abertura

Figura C.2 – Formulário de Cadastro de Alunos

Figura C.3 – Formulário de Cadastro de Áreas de Conhecimento e Disciplina

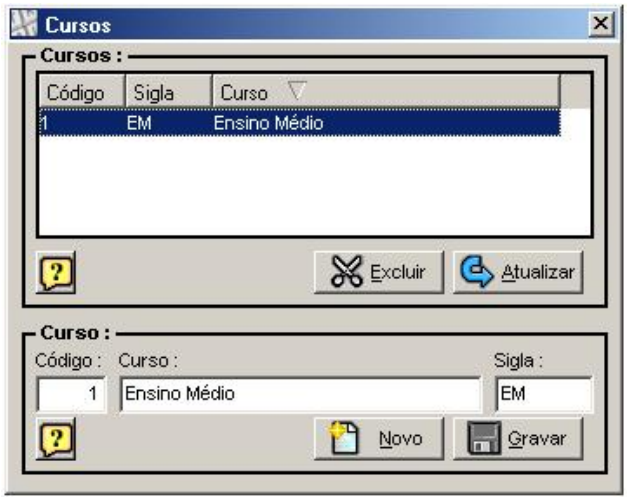


Figura C.4 – Formulário de Cadastro de Cursos

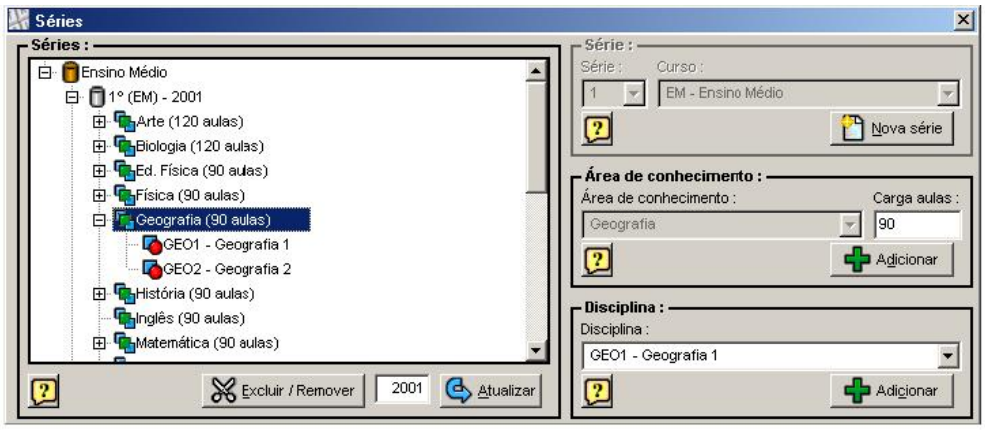


Figura C.5 – Formulário de Cadastro de Séries

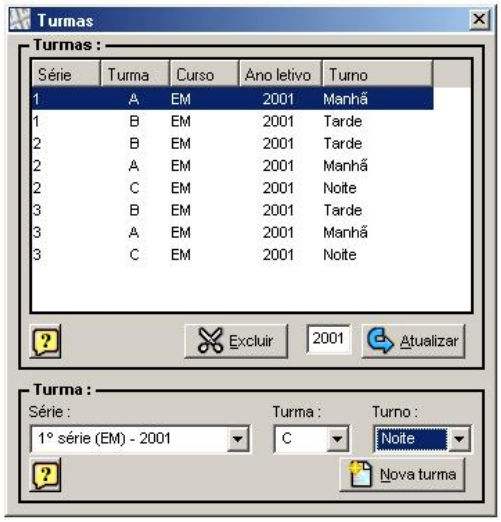


Figura C.6 – Formulário de Cadastro de Turmas

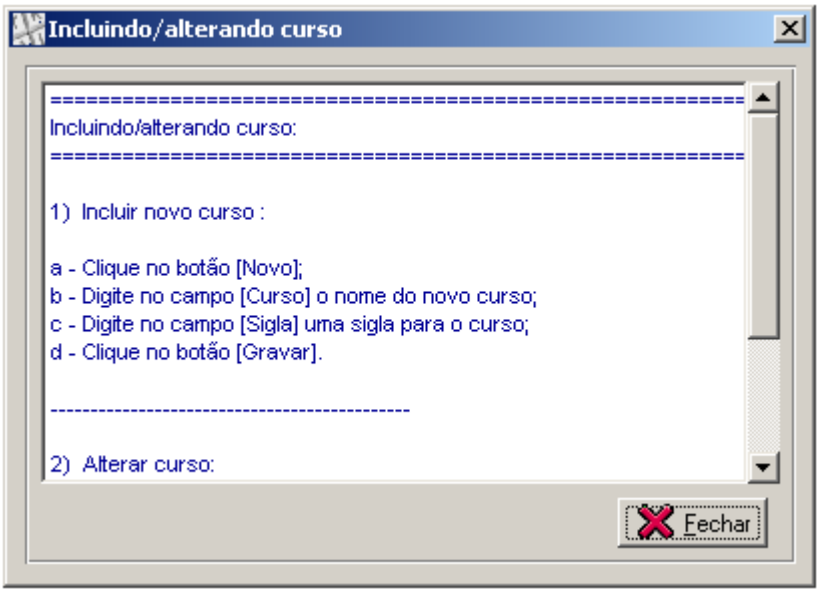


Figura C.7 – Formulário de Ajuda

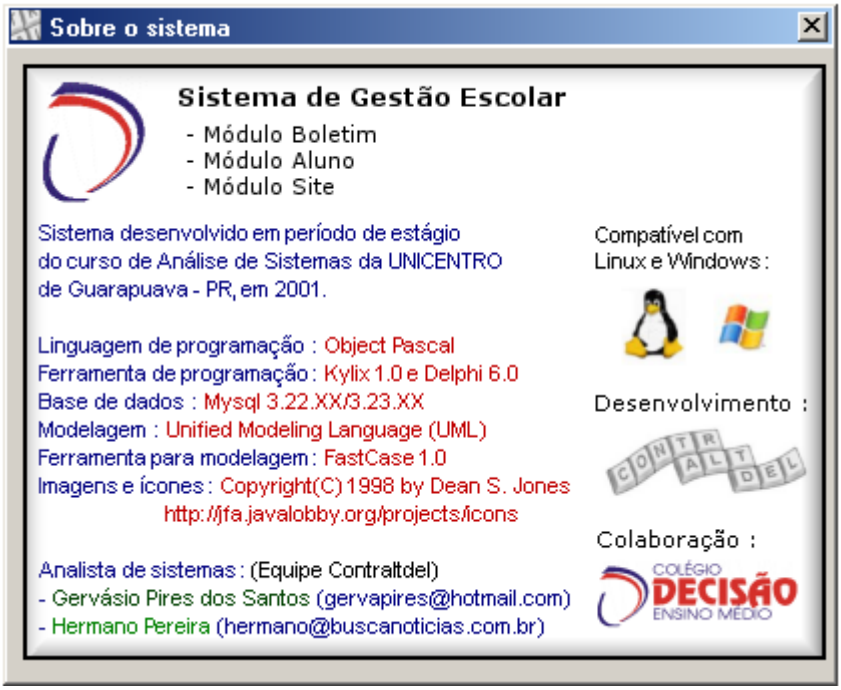


Figura C.8 – Formulário Sobre o Sistema

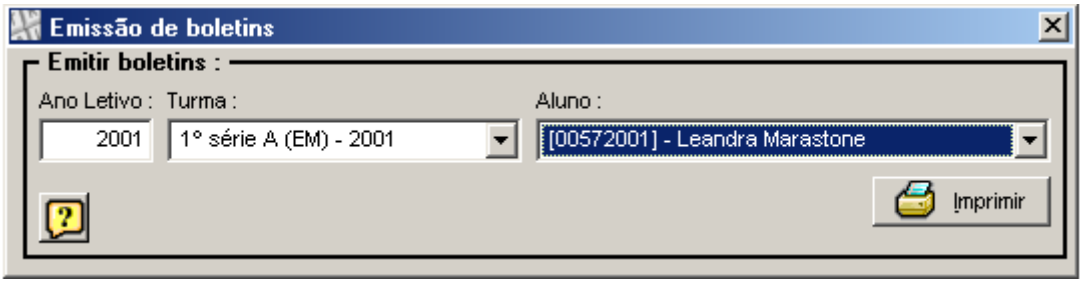


Figura C.9 – Formulário de Emissão de Boletins

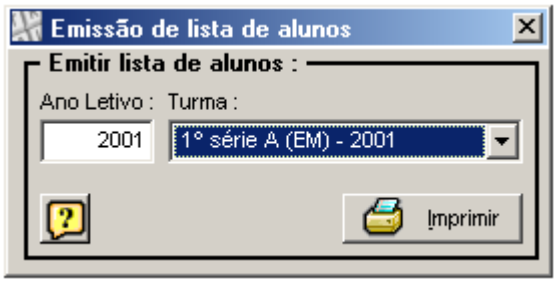


Figura C.10 – Formulário Emissão de Lista de Alunos

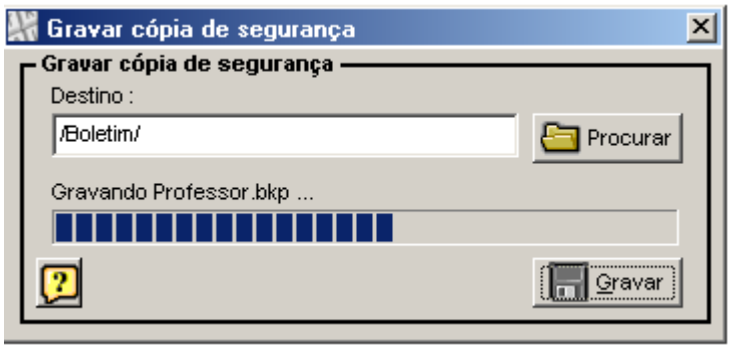


Figura C.11 – Formulário Gravar Cópia de Segurança

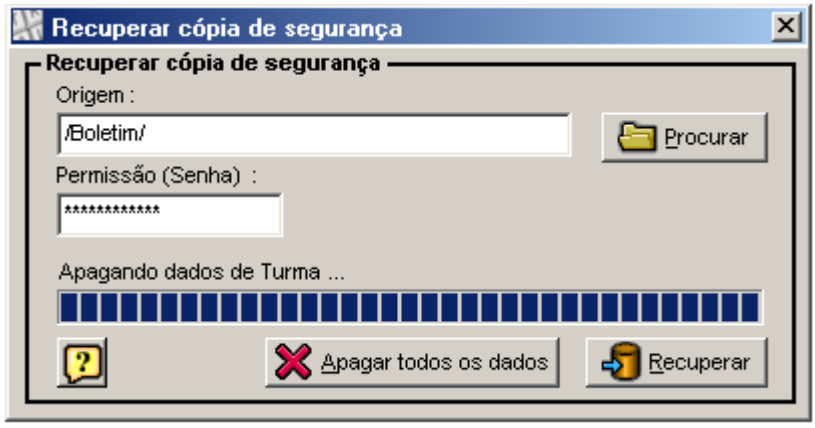


Figura C.12 – Formulário Recuperar Cópia de Segurança