

1. Introdução

Este trabalho relata e oficializa todo o desenvolvimento do projeto criado para o Espaço Cultural Flávio Fernandes de Matos, que visa aperfeiçoar os processos de armazenamento e manipulação de dados.

Esta documentação vem demonstrar todo o conteúdo dos processos que foram passados até a conclusão, desde a estrutura do software, passando pelo desenvolvimento e o manual de utilização do mesmo.

A missão do grupo F11 é facilitar, fornecer agilidade segurança as informações obtidas pelo Espaço Cultural. Esperamos aqui esclarecer de forma completa e coerente todo o planejamento e execução deste projeto.

2. Conhecimento Científico

"Ciência é conhecimento organizado. Sabedoria é vida organizada."

(IMMANUEL KANT, 1973)

Francisco Saiz (2000) retrata que o homem é um ser que faz questionamentos existenciais, e que tem que interpretar a si e ao mundo em que vive, atribuindo-lhes significados. Cria representações significativas da realidade, a qual chamou de conhecimento. O conhecimento, dependendo da forma pela qual se chega a essa representação, pode ser classificado em diversos tipos como, por exemplo, mítico, ordinário, dogmático e científico.

O conhecimento científico é o que é produzido pela investigação científica, através de seus métodos. Surge não apenas da necessidade de encontrar soluções para problemas de ordem prática da vida diária, mas do desejo de fornecer explicações sistemáticas que possam ser testadas e criticadas através de provas empíricas.

A investigação científica se inicia quando se descobre que os conhecimentos existentes, originários querem o senso comum, quer do corpo de conhecimentos existentes na ciência, são insuficientes para explicar os problemas surgidos. O conhecimento prévio que nos lança a um problema pode ser tanto do conhecimento ordinário quanto do científico.

GALLIANO (1979, p. 24-30) refere-se ao mesmo como, o conhecimento racional, sistemático, exato e verificável da realidade. Sua origem está nos procedimentos de verificação baseados na metodologia científica. Podemos então dizer que o Conhecimento Científico:

- É racional e objetivo.
- Atém-se aos fatos.
- Transcende aos fatos.
- É analítico.
- Requer exatidão e clareza.
- É comunicável.
- É verificável.
- Depende de investigação metódica.
- Busca e aplica leis.
- É explicativo.
- Pode fazer previsões.
- É aberto.
- É útil

2.1. Metodologia

A Metodologia é o estudo dos métodos. Ou então as etapas a seguir num determinado processo. Tem como finalidade captar e analisar as características dos vários métodos disponíveis, avaliarem as suas capacidades, potencialidades, limitações ou distorções e criticar os pressupostos ou as implicações de sua utilização. Além de ser uma disciplina que estuda os métodos, a metodologia é também considerada uma forma de conduzir a pesquisa ou um conjunto de regras para ensino de ciência e arte.

A Metodologia é a explicação minuciosa, detalhada, rigorosa e exata de toda ação desenvolvida no método (caminho) do trabalho de pesquisa. É a explicação do tipo

de pesquisa, do instrumental utilizado (questionário, entrevista etc.), do tempo previsto, da equipe de pesquisadores e da divisão do trabalho, das formas de tabulação e tratamento dos dados, enfim, de tudo aquilo que se utilizou no trabalho de pesquisa.

Em Gestão de Projetos, existe a metodologia geral e a metodologia detalhada. A metodologia pode ser dividida em vários métodos até chegar num determinado objetivo.

2.1.1. Metodologia aplicada ao projeto Eclipse

A metodologia do grupo F11 utilizou a análise do funcionamento da instituição como pivô do início do projeto de desenvolvimento. No processo de análise levantamos os problemas e estudamos as possíveis soluções.

A partir do momento que passamos da fase de análise, começamos a fase de elaboração do banco de dados e suas respectivas documentações, no qual nos dá o controle sobre as entidades, depósitos de dados, relações e eventos ocorridos. O banco de dados foi desenvolvido no Access 2000.

Com o banco de dados construído, demos início ao desenvolvimento do sistema, procuramos dar comodidade ao usuário, sem a poluição visual das telas. Foi usado o Delphi 7.0 para a construção do software, assim nós concluímos o projeto.

2.2. Objetivo

"Sempre que você perceber que alguma coisa está sendo realizada, pode saber que haverá por perto um mono-maníaco com uma missão." (Peter Drucker, 2002)

Objetivos são metas que o indivíduo ou o grupo se coloca a assumir sobre sua vida ou determinado projeto que se deseja realizar. O estabelecimento de objetivos é fundamental para se concluir o que se foi proposto de maneira que haja êxito no mesmo.

2.2.1. Objetivo F11

Atender as necessidades do Espaço Cultural Flávio Fernandes de Matos, fazendo com que o usuário tenha interatividade com o sistema Eclipse, retribuindo com segurança, qualidade para associação que nos depositou confiança nos dando oportunidade de desenvolver um sistema para informatizar o seu dia-a-dia.

3. Concepção de Software

Concepção de software são as atividades incluídas no processo de desenvolvimento de software. O desenvolvimento refere-se às fases do ciclo de vida responsáveis pelo projeto de construção de sistemas, incluindo ainda a sua implementação. Excluem-se, por exemplo, os estudos de viabilidade econômica, as tarefas de manutenção e a utilização efetiva do sistema.

4. Analise de Sistemas

De fato, e conforme Luís Gouveia (1996) é possível dizer que a Análise de Sistemas consiste no processo de estudo de uma organização (na sua totalidade ou em parte), em que se procura realizar o levantamento exaustivo de como funciona e desta forma descrever os processos de resolução seguidos para as suas necessidades (que garantam atingir os objetivos da organização). Os profissionais responsáveis por este conjunto de tarefas são designados Analistas de Sistemas.

Antes das atividades de Projeto ser iniciada pelo Analista de Sistemas, é conduzido um estudo do sistema para conhecimento dos detalhes da situação atual da organização. A informação coletada até ao fim do estudo forma a base para a criação de alternativas a considerar na fase de Projeto de construção do Sistema. São os gestores e não os analistas de sistemas que escolhem a alternativa a seguir para a fase de Implementação, face das eventuais opções de projeto para o sistema (estudadas e descritas na fase de análise).

Virtualmente todas as organizações são sistemas. Elas interagem com o exterior através de inputs - entradas de dados - a receber e dos outputs a fornecer - saídas de dados, sistemas com estas características são designados por Sistemas Abertos.

A análise de sistema é essencial para iniciar o processo de desenvolvimento dos sistemas, pois são constituídos por um conjunto de pequenos sistemas agregados, designados por subsistemas. Estes operam para atingir um objetivo específico que contribui para a satisfação geral do objetivo do sistema. É com base nos objetivos encontrados que se especificam as necessidades de armazenamento e as necessidades de controle. Uma classe importante de sistemas são aqueles que interagem

com os ambientes que os rodeiam sendo designados por sistemas abertos e as suas performances são avaliadas comparando com as normas que eventualmente existam para sistemas semelhantes. Os resultados obtidos da comparação e do desempenho do sistema com as suas normas constituem o feedback – é a realimentação de informação necessária para o consciente ajustamento das atividades do sistema e para melhoria do geral de seu desempenho.

Os Sistemas de Informação servem os sistemas empresariais através do processamento de transações e suportando a tomada de decisões. Os Sistemas de Transações assistem as operações de rotina, enquanto que os sistemas de gestão e de decisão ajudam na tomada de decisões. Os componentes destes sistemas incluem hardware, software e sistemas de armazenamento contendo dados e informações. As aplicações dos sistemas de informação são constituídas por procedimentos específicos, conjuntos de programas, ficheiros e equipamentos, tudo cuidadosamente integrado para atingir os objetivos determinados.

Os analistas são os responsáveis pelo desenvolvimento dos Sistemas de Informação sendo também os profissionais responsáveis pelo diálogo com gestores e outros profissionais da empresas, interpretando as suas especificações. O ciclo de vida de desenvolvimento de um sistema é o conjunto de atividades que analistas e conceitores de sistemas fazem para desenvolver e implementar um sistema de informação. A análise de Sistema é conjunto de atividades que inclui as investigações preliminares, a recolha de dados, determinação de objetivos, o desenvolvimento de protótipos para o projeto do sistema, o desenvolvimento do software, o teste do sistema e a de sua implementação.

4.1. A Importância da Análise de Sistema

Assim para Luís Gouveia (1996), a análise de sistemas é uma atividade crítica no processo de desenvolvimento de sistemas, por ser uma etapa inicial e cujas falhas terão efeitos em cadeia nas etapas subseqüentes assim como no produto final. A determinação incorreta dos requisitos levará a obtenção e disponibilização de sistemas informáticos inadequado ao sistema de informação e ao sistema organizacional.

O custo relativo de reparar problemas de requisitos detectados durante a fase de testes finais ou operacionais pode ser de até 100 (cem) vezes superior àqueles que são detectados durante a fase de análise de sistemas.

4.2. A importância dos Utilizadores

Luís Gouveia (1996) Acredita que o envolvimento dos utilizadores é um tópico crítico em todo o processo de desenvolvimento de sistemas de informação. O seu nível de participação tem um impacto direto, positivo e significativo na satisfação dos utilizadores. Quanto maior for sua participação no projeto, maior será sua satisfação como utilizador final do sistema.

4.3. Dificuldades na Análise de Sistemas

As diferenças formativas e culturais dos Analistas face aos Utilizadores provocam barreiras comunicacionais, onde:

- Há o fraco conhecimento e domínio do negócio;
- Há pontos de vista técnico/tecnológico excessivos;
- Há o Julgamento de que todos os requisitos são completamente definidos ao início do projeto e que não mudaram no decorrer do tempo;

- Julgar que não é necessário grande envolvimento e comprometimento dos utilizadores, ou que o ponto de vista de um já é o suficiente;
- Métodos e técnicas tradicionais não promovem trabalho colaborativo entre Analistas e Utilizadores nem a focagem de aspectos sócios organizacionais.

4.4. Análise Estruturada

Segundo Carlos Silvano Valvassori (2004), afirma que como todos os métodos de análise de requisitos de software, a análise estruturada, vêm a ser um a atividade de construção de modelos com fluxo e conteúdo das informações divididas em partições funcionais, onde descreve a essência do que deve ser construído.

A análise estruturada também contém gráficos que possibilita o analista criar modelos de fluxo de informação, com uma heurística para o uso dos símbolos, juntamente com um dicionário de dados e as narrativas de processamento como o complemento aos modelos de fluxo de informação.

Um modelo de fluxo pode ser criado para qualquer sistema baseado em computador, independentemente do tamanho e complexidade. Um meio importante de representar os dados vem a ser através de um DFD (Diagrama de Fluxo de Dados).

4.4.1. Objetivos da Análise Estruturada

É retratado por Carlos Silvano Valvassori (2004), O documento a ser padronizado deve ser:

- Passível de manutenção
- Gráfico
- Lógico
- Rigoroso
- Conciso
- Legível

Tudo isso deve ser um subproduto natural do trabalho. Ou seja, terminada a fase de análise, ninguém deve necessitar de mais tempo para preparar a documentação - ela já deve estar concluída.

4.5. UML

"A UML (Unified Modeling Language)é uma linguagem para especificação, documentação, visualização e desenvolvimento de sistemas orientados a objetos."

(DOUGLAS MARCOS DA SILVA, 2001)

O grande problema de acordo com Douglas Marcos da Silva (2001) do desenvolvimento de novos sistemas utilizando a orientação a objetos nas fases de análise de requisitos, análise de sistemas e design é que não existe uma notação padronizada e realmente eficaz que abranja qualquer tipo de aplicação que se deseje. Cada simbologia existente possui seus próprios conceitos, gráficos e terminologias, resultando numa grande confusão, especialmente para aqueles que querem utilizar a orien-

tação a objetos não só sabendo para que lado aponte a seta de um relacionamento, mas sabendo criar modelos de qualidade para ajudá-los a construir e manter sistemas cada vez mais eficazes.

Quando a (UML) foi lançada, muitos desenvolvedores da área da orientação a objetos ficaram entusiasmados já que essa padronização proposta pela UML era o tipo de força que eles sempre esperaram. A UML é muito mais que a padronização de uma notação. É também o desenvolvimento de novos conceitos não normalmente usados. Por isso e muitas outras razões, o bom entendimento da UML não é apenas aprender a simbologia e o seu significado, mas também significa aprender a modelar orientado a objetos no estado da arte.

UML foi desenvolvida por Grady Booch, James Rumbaugh, e Ivar Jacobson que são conhecidos como "os três amigos". Eles possuem um extenso conhecimento na área de modelagem orientado a objetos já que as três mais conceituadas metodologias de modelagem orientada a objetos foram eles que desenvolveram e a UML é a junção do que havia de melhor nestas três metodologias adicionado novos conceitos e visões da linguagem.

UML aborda o caráter estático e dinâmico do sistema a ser analisado levando em consideração, já no período de modelagem, todas as futuras características do sistema em relação à utilização de "packages" próprios da linguagem a ser utilizada, utilização do banco de dados bem como as diversas especificações do sistema a ser desenvolvido de acordo com as métricas finais do sistema.

4.6. Análise estruturada x UML

Podemos entender análise estruturada como a construção de fluxo e conteúdo das informações descrevendo aquilo que deve ser construído. Também contem gráficos que possibilita o analista criar modelos de fluxo de informações. Um meio importante de representar os dados vem a ser através de um DFD (Diagrama de Fluxo de Dados).

Os usuários obtêm uma idéia mais clara do sistema proposto pelo diagrama de fluxo de dados, do que a obtida através da narrativa e Fluxograma de sistemas físicos, a apresentação em termos de fluxo lógico consegue mostrar mal-entendidos e pontos controversos, As interfaces entre o novo sistema e outros já existentes, são mostrados de modo bem mais claro, o uso de dicionário de dados para guardar os itens do glossário do projeto economiza tempo ao resolver rapidamente os casos em que pessoas chamam as mesmas coisas por diferentes nomes.

Já UML tratasse de uma linguagem de modelagem para documentar e visualizar os artefatos que são especificados e construídos na análise e projeto de um sistema.

Podemos citar como principais vantagens na utilização de UML:

- Fornecer aos usuários uma linguagem de modelagem visual expressiva e pronta para uso, visando o desenvolvimento de modelos de negócios.

Uma linguagem de modelagem visual apresenta três benefícios principais:

- Visualização
- Gerenciamento da Complexidade
- Comunicação

- 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
- Suportar conceitos de desenvolvimento de nível mais elevado tais como colaborações, estrutura de trabalho, padrões e componentes.
- Integrar as melhores práticas

5. Estudo de viabilidade

Trata-se de um levantamento de informações que consistem em analisar a viabilidade do projeto em seu desenvolvimento até sua execução. É o estudo de viabilidade que determinará pontos críticos do seu projeto, diferentes alternativas de soluções para o problema e, até mesmo, se o projeto será levado adiante ou não.

O estudo de viabilidade consiste, na prática, de um documento com formato mais ou menos definido que descreve de maneira geral o problema a ser tratado. A organização para a qual se destina o software, e as mais variadas soluções acompanhadas de análises comparativas entre elas.

O estudo de viabilidade já é um procedimento padrão no processo de design, do qual depende todo o restante do projeto.

6. Processo de desenvolvimento de software

É um conjunto de atividades, ligadas por padrões de relacionamento entre ela, pelas quais se as atividades operarem corretamente e de acordo com os padrões requeridos, o resultado desejado é produzido. O resultado desejado é um software de alta qualidade e baixo custo. Obviamente, um processo que não aumenta a produção (não suporta projetos de software grandes) ou não pode produzir software com boa qualidade não é um processo adequado.

(JALOTE, P. 1997)

O processo de desenvolvimento pode ser considerado como um conjunto de atividades onde se unem métodos, ferramentas e prática para que seja construído de software conhecido também como sistema. Para ser realizado devemos considerar as seguintes informações:

- As atividades a serem realizadas pelo sistema;
- Seus recursos necessários;
- Os artefatos requeridos
- Produção de artefatos;
- Os procedimentos que foram adotados;
- E seu modelo de ciclo de vida;

6.1. Processo de produção de Software

Quando se faz referência ao processo de Produção de software, normalmente se associa um modelo a este processo. Processo esse que deverá ser seguido para a construção, liberação e evolução do produto, desde a concepção de uma idéia até a liberação do produto. Esta modelagem de processo é freqüentemente citada como ciclo de vida de desenvolvimento de software.

6.2. Modelos do Processo de Produção de Software

Com base na teoria de Luís Fernando Garcia (2008) onde diz que alguns modelos de desenvolvimento estruturado de software estabeleceram níveis ou etapas a serem seguidas, com o intuito de aprimorar o processo de produção de software. O modelo "cascata" foi o modelo precursor na modelagem do processo de produção de software e é considerado o modelo tradicional. Outros modelos como o evolutivo, espiral e o transformacional surgiram na tentativa de suprir as deficiências do modelo tradicional.

6.2.1. Modelo Cascata (Waterfall Model)

Para Luis F. Garcia (2008) o modelo cascata é estruturado de forma a parecer uma cascata de fases, como ilustra a figura abaixo, onde o final de uma fase implica no início de outra. Este tipo de comportamento ressalta a qualidade de um modelo rígido e linear, no sentido que uma fase começa após a outra numa direção linear, ou seja, sem o devido feedback para as fases anteriores.

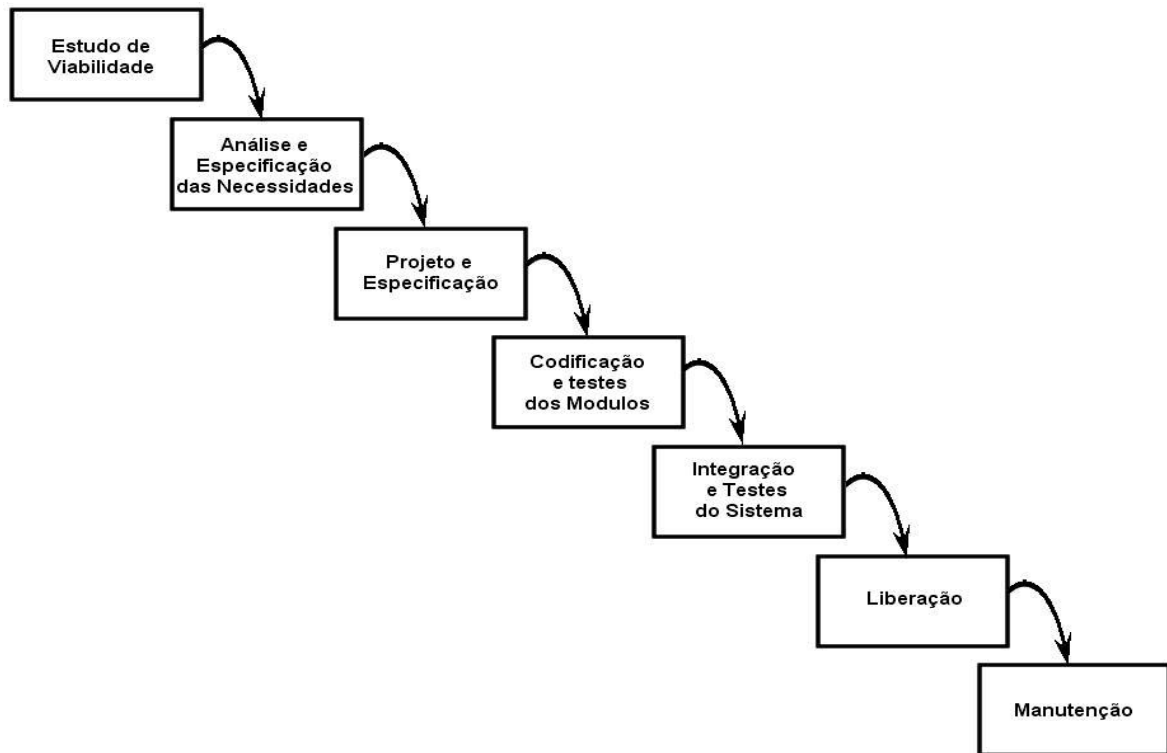


Figura 1 - Modelo Cascata

Os resultados de cada uma destas fases são produtos intermediários que se agregam ao final do processo gerando um produto final. Neste modelo cada fase ou etapa está estruturada com um conjunto de atividades que podem ser executadas concorrentemente por diferentes pessoas. As fases do modelo cascata são basicamente as seguintes:

- Estudo de viabilidade onde é através de pesquisas de mercado, levantamento de informações acerca do tema do projeto e das condições de desenvolvimento, o estudo de viabilidade define os custos, prazos e benefícios do produto para que se verifique a possibilidade ou não da execução do projeto. Em algumas situações pode ser a primeira e a última fase do processo;
- Análise e especificação das necessidades sendo que uma vez que o estudo de viabilidade demonstre os benefícios e a possibilidade de execução, é a fa-

se inicial nos projetos para desenvolvimento de software. Objetiva identificar e documentar exatamente quais são os requisitos do produto. Isto é feito em conjunto com usuários, desenvolvedores e, em algumas situações, com organizações de marketing. Algumas metodologias de engenharia de software sugerem que esta fase deva produzir também os manuais do usuário e os planos de testes do produto;

- Projeto e especificação onde nesta fase são utilizadas as documentações geradas na fase anterior para se projetar o software. Em geral esta fase subdivide-se em outras duas fases: projeto arquitetural (ou projeto de alto nível) e projeto detalhado. O projeto de alto nível trata da organização e estrutura dos módulos no contexto geral do produto enquanto que o projeto detalhado refina o projeto de alto nível a fim de identificar os detalhes individuais de cada módulo;

- Codificação e testes dos módulos assim nesta fase são produzidos o código (programa) do produto. As fases anteriores podem até desenvolver códigos no nível de protótipos, testes, simulações, etc. Contudo, esta fase é exclusiva dos programadores ou desenvolvedores. Os módulos desenvolvidos nesta fase são individualmente testados antes de serem liberados às fases seguintes;

- A integração e testes do sistema assim cada módulo desenvolvido na fase anterior é integrado aos demais módulos durante esta fase. Assim, durante a integração, os módulos relacionados são testados e, após a integração, é testado o produto como um todo;

- Liberação e manutenção onde após os testes de integração e do produto, o mesmo é liberado aos usuários ou ao mercado consumidor e entra na fase de manutenção. Toda e qualquer modificação feita no produto após o mesmo ter sido liberado é atribuída à fase de manutenção.

Existem algumas variações deste modelo dependendo da organização que o utiliza e da especificação do projeto e, embora a variação normalmente ocorra no número e na natureza de suas etapas, a filosofia básica do modelo é a mesma.

O modelo cascata fez duas contribuições fundamentais para a compreensão do processo de desenvolvimento de software. Na primeira ele nos diz que o processo de desenvolvimento de software deveria ser submetido a uma disciplina, planejamento e gerenciamento e, na segunda, a implementação do produto deveria ser adiada até que os objetivos do produto estejam bem compreendidos.

Embora o modelo cascata ser o paradigma mais utilizado no desenvolvimento da engenharia de software, ele recebeu severas críticas por sua passividade e por apresentar um formalismo excessivo. Isto o torna também extremamente burocrático, já que existe uma forte preocupação na elaboração e aprovação de documentos para que seja possível dar continuidade ao processo de produção. A documentação, embora necessária para o bom andamento de um projeto, deve ser flexível para possibilitar um maior ou menor grau de formalismo ao processo de acordo com as necessidades de registro.

Além disso, mesmo tendo introduzido disciplina no processo de desenvolvimento de software, o modelo cascata é mais indicado para software que apresentam seus requisitos bem conhecidos antecipadamente. Isso se deve ao fato da rigidez do mode-

lo dificultar futuras acomodações de mudanças de requisitos e mudanças gerenciais, ambas as ocorrências freqüentes no processo de desenvolvimento de software.

6.2.2. Modelo Evolutivo (Evolutionary Model)

O modelo evolutivo é um modelo no qual os estágios consistem em expandir e incrementar um produto de software operacional, onde a direção da evolução é determinada pela experiência operacional assim conclui Luis F. Garcia (2008).

Este modelo se baseia no princípio de que a primeira versão de um sistema de software deve ser utilizada como um protótipo descartável e, portanto, utilizado temporariamente. A segunda versão é então desenvolvida utilizando-se a filosofia do modelo cascata.

O objetivo deste princípio é de solidificar os requisitos dos usuários, afim de que se possa estruturar uma base sólida para o desenvolvimento real do software. Com isso se conseguiria rapidamente liberar uma versão preliminar do produto com o intuito de avaliar a satisfação dos usuários e se ter conhecimento das alterações ou novas funcionalidades que deverão ser verificadas.

Este modelo permite que os usuários acompanhem o desenvolvimento do produto desde seus estágios iniciais, de forma que o mesmo poderá ser adaptado ou modificado ao longo do processo de produção de acordo com as necessidades dos usuários. Isto é feito ao ponto de que quando existir um protótipo refinado que satisfaça e atenda a todos os requisitos dos usuários, o produto estará pronto.

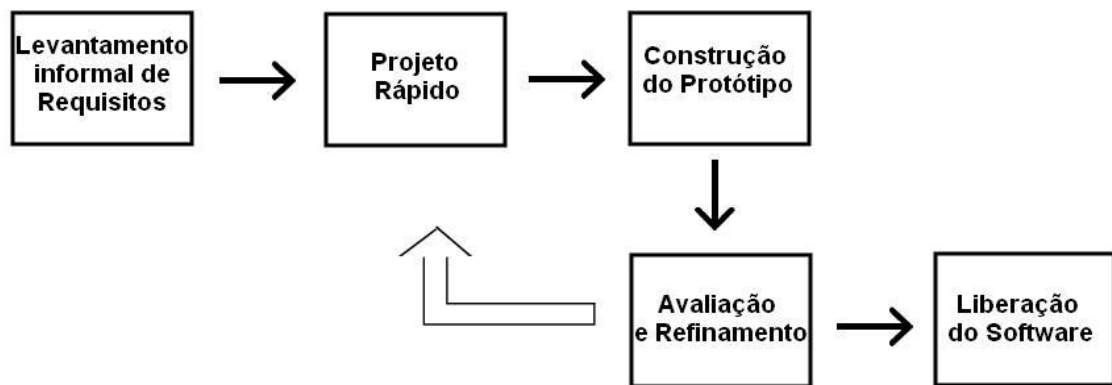


Figura 2 - Modelo Evolutivo

O fato de que cada pequena versão ser considerada como um produto gera um feedback importante entre usuário e engenheiro possibilitando que o usuário receba sempre alguma coisa substancial rapidamente e com isso possa interagir e contribuir para a evolução do produto. Evolução esta que pode ser uma evolução funcional (aumento das funcionalidades do sistema) ou uma evolução rumo ao final do projeto, o que conseqüentemente geraria um produto final.

6.2.3. Modelo Transformacional (Transformation Model)

Este modelo está enraizado profundamente na teoria da especificação formal. Esta idéia é traduzida no fato de que o desenvolvimento de software pode ser visto como uma seqüência de passos que gradativamente transformarão a especificação na implementação.

A idéia básica de Luis F. Garcia (2008) deste modelo consiste de dois estágios principais: análise e especificação de requisitos e otimização. O primeiro provê requisitos formais, os quais são supridos pelo processo de otimização que faz o refinamento do desempenho até atingirmos um resultado satisfatoriamente otimizado.

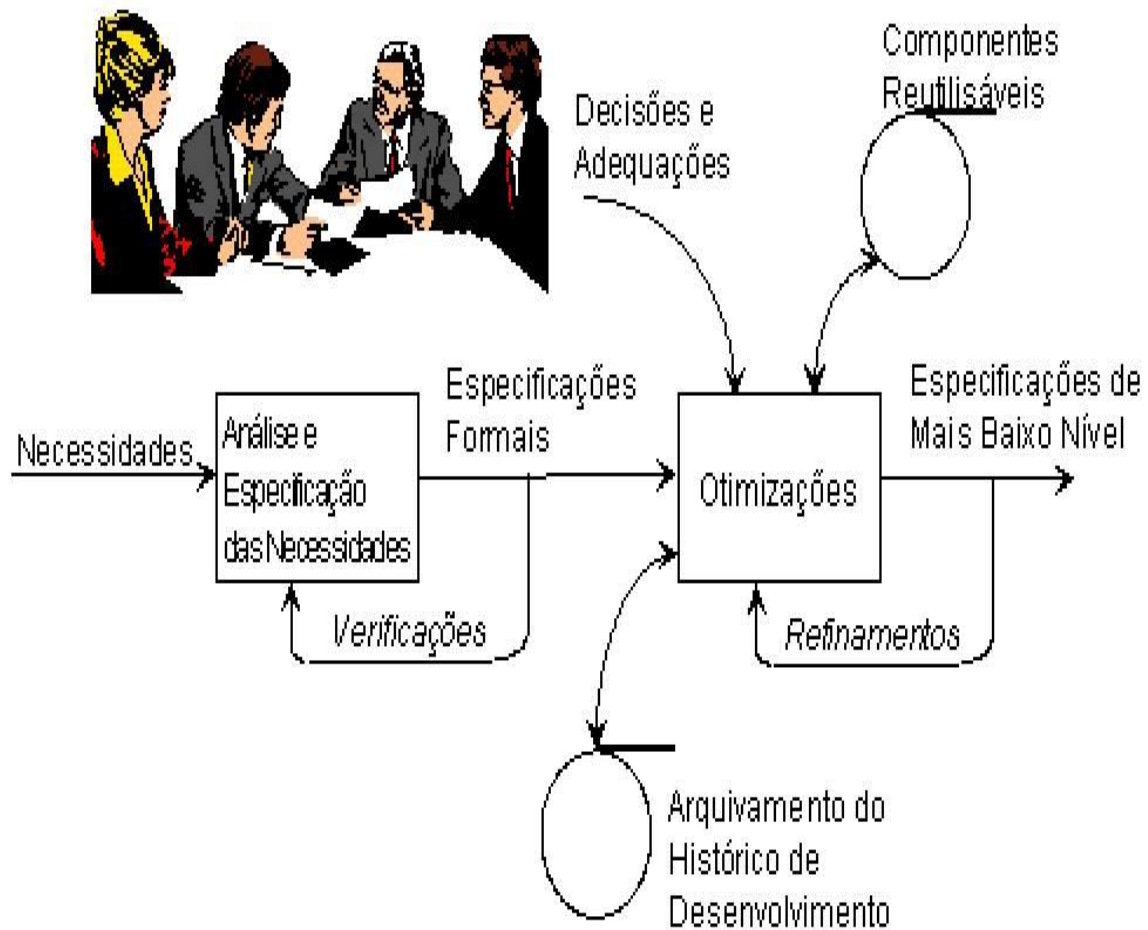


Figura 3 - Modelo Transformacional

Todo o processo de transformação é controlado por um (ou mais) engenheiro e pode aproveitar as vantagens de componentes reutilizáveis disponíveis. Novos componentes reutilizáveis desenvolvidos durante o processo podem ser adicionados/armazenados em biblioteca.

O suporte a evolução dos programas neste modelo contrasta com a prática comumente usada no modelo cascata. No modelo cascata as mudanças são difíceis pela forma como é feito o tratamento do problema desde os requisitos até o código, ou seja, se as mudanças não forem antecipadamente previstas, elas serão sempre tratadas como reparos de emergência.

Em consequência disto, o programador, pela forte pressão que sofre, acaba apenas por mudar o código sem propagar os efeitos destas mudanças para mudanças de especificação. Desta forma especificação e implementação estarão em divergência, fazendo com que futuras mudanças na aplicação se tornem difíceis de serem desenvolvidas.

Já no modelo transformacional as coisas são sensivelmente diferentes. Considerando que um histórico do desenvolvimento do software juntamente com as adequações de cada transformação está armazenado em ambientes de suporte, o programador pode ser forçado a renunciar mudanças diretas no código, e, além disso, ele é obrigado a recompor o histórico do problema fazendo o conserto no lugar apropriado do histórico.

6.2.4. Modelo Espiral (Spiral Model)

O modelo espiral objetiva fornecer uma estruturação técnica para o processo de desenvolvimento de software. Luis F. Garcia (2008) considera também, os custos agregados durante este processo, proporcionando uma ferramenta de análise para fatores de riscos para a gerência do projeto.

A grande característica deste modelo está no fato dele ser cíclico, e não linear como o modelo cascata. Cada ciclo deste modelo consiste de quatro estágios e cada estágio é representado por um quadrante de um diagrama cartesiano. O raio do espiral constitui o custo acumulado do projeto e a dimensão angular representa o progresso no processo.

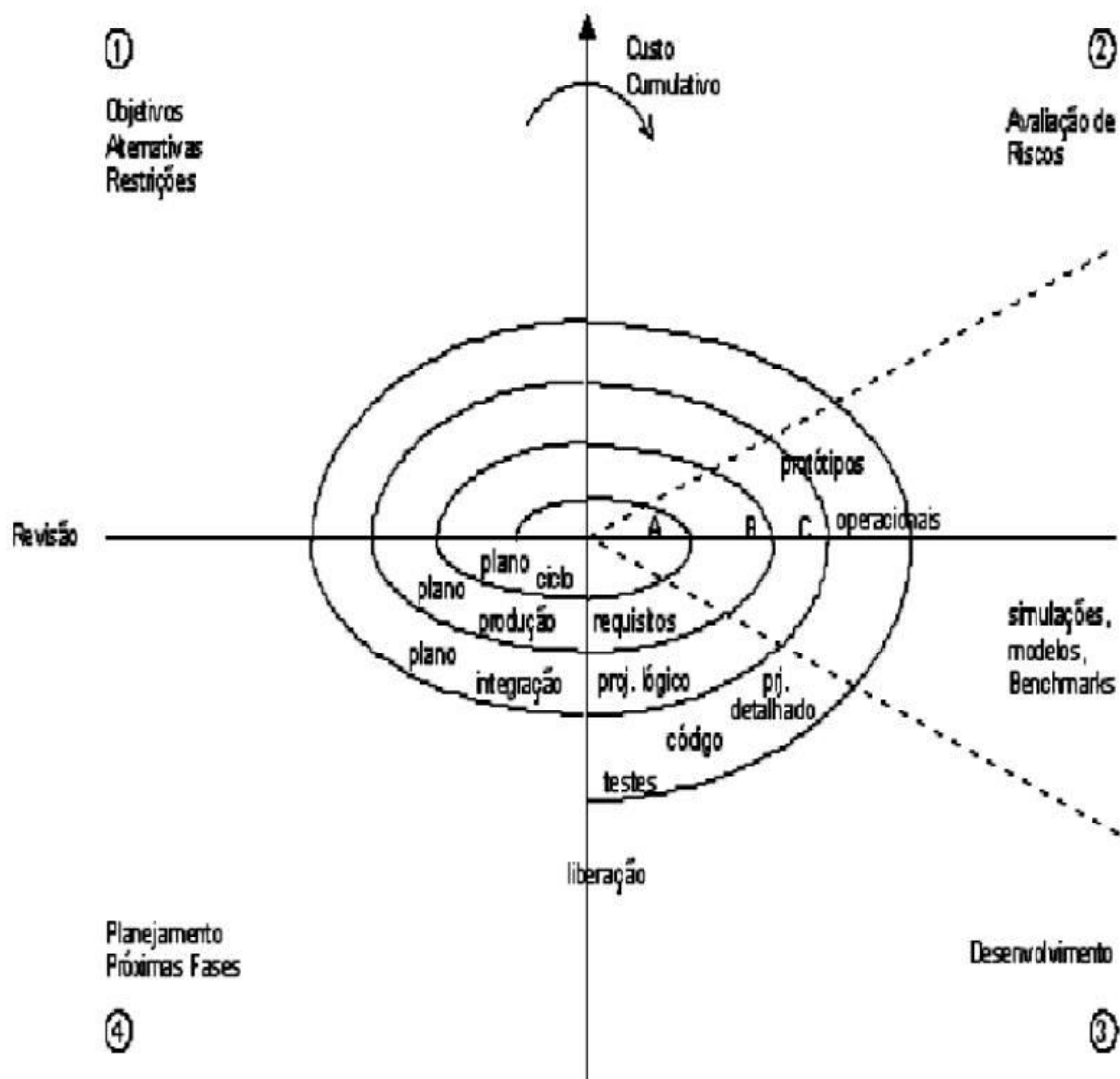


Figura 4 - Modelo Espiral

Como se pode notar, no primeiro estágio o ciclo é composto pela determinação dos objetivos do projeto, que são seguidos, já no segundo estágio, pela avaliação dos riscos envolvidos na escolha das alternativas para execução dos objetivos.

No terceiro estágio o desenvolvimento das atividades previstas é concretizado. Cabe salientar que o nível de dificuldade econômica para a continuidade do projeto é paralelamente analisado neste estágio. A partir disso pode-se evoluir de um projeto lógico para um projeto detalhado ou ainda para a evolução de uma segunda versão.

Caso o ciclo de vida do projeto esteja acordado, no planejamento das fases seguintes é então definido no estágio quatro. É neste estágio também que os avanços atingidos são avaliados e revisados juntamente com a definição das metas seguintes.

7. DFD - Diagrama de fluxo de dados

Segundo Denis Alcides (1999), o DFD é uma técnica usada na programação estruturada de diagramação de software que possui diversos tipos de diagramas, derivando-se em outros diagramas subsequentes.

Assim um DFD representa:

- Imagem do sistema, projeto ou produto;
- Modelo de organização;
- Apresentação em etapas com aumento gradativo de detalhes;
- Utilização dos princípios da modularização e da hierarquização.

Assim, podemos ter diversos níveis de DFD de forma a representar o fluxo de dados da aplicação.

a) DFD nível 0 - Apresenta uma visão clara do produto com todos os macro-processos, com entidades externas, fluxo de dados e depósito de dados principais.


b) DFD nível 1 - É uma expansão do nível zero com mais detalhes e mais completo incluindo o tratamento de exceções.

7.1. Simbologia do DFD

Com a visão de Denis Alcides (1999), logo veremos algumas simbologias do DFD:

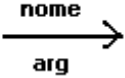
- Entidades Externas

Tabela 1 - Entidades Externas DFD

<p>São categorias lógicas de objetos ou pessoas que representam Origem ou destino de dados, e, que acionam um sistema e/ou recebem informações;</p> <p>Podem ser pessoas, sistemas ou unidades departamentais. Suas regras são:</p> <p>x - letra para identificação;</p> <p>Nome - nome da entidade: Ex: Clientes, Sistema Acesso, Banco, etc.</p> <p>No mínimo temos duas: quem usa o sistema (cliente) e quem opera o sistema (departamento A)</p>	
--	---


- Fluxo de dados

Tabela 2- Fluxo de Dados DFD

<p>É o Meio por onde os dados e as informações trafegam;</p> <p>Regras: Nome: nome do dado. Ex: Pedido, Nota Fiscal, Produto, Item, Arg: argumento de acesso a um depósito. Ex: CGC, CPF, CEP, código, matrícula, Nome, etc.</p> <p>Sempre envolvem processos não sendo possível o fluxo de entidade para entidade, entidade para depósito de dados, depósito de dados para depósito de dados para</p>	
--	---

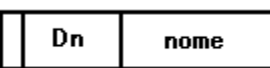
- Processos

Tabela 3 – Processos DFD

<p>Transformam fluxos de dados em uma atividade e são módulos do sistema;</p> <p>Regras: n: número de referência do processo. Ex: 0, 1, 2, 3, 1.1, 1.2</p> <p>Função: descreve o processo no verbo infinitivo. Ex: Cadastrar Cliente, Gerar Arquivo, Imprimir Relatório, etc.</p> <p>Loc: local físico onde se desenvolve o processo. Ex: Almoxarifado; Contabilidade, etc.</p> <p>Dica: Para descobrir um processo relate os requisitos do sistema. (Cadastrar Cliente, Efetuar Logon, etc.)</p>	
---	---

- Depósito de Dados

Tabela 4 - Depósitos de dados DFD

<p>São locais de armazenamento de dados;</p> <p>São arquivos físicos ;</p> <p>Regras:</p> <p>Dn: número do depósito. Ex: 0, 1, 2,3, D1/1, D1/2</p> <p>Nome: nome do depósito. Ex: Clientes, Produtos, Contas, etc.</p> <p>Para tornar mais fácil identificar DD leve em conta dois tipos de arquivos: Cadastral e de Movimento (Movimento de Itens, etc.)</p>	
---	---

7.2. Etapas de elaboração de um DFD

- Identificar e descrever os requisitos funcionais;
- Identificar entidades externas (EE);
- Associar o fluxo de dados que as entidades enviam, consomem ou recebem;
- Identificar consultas

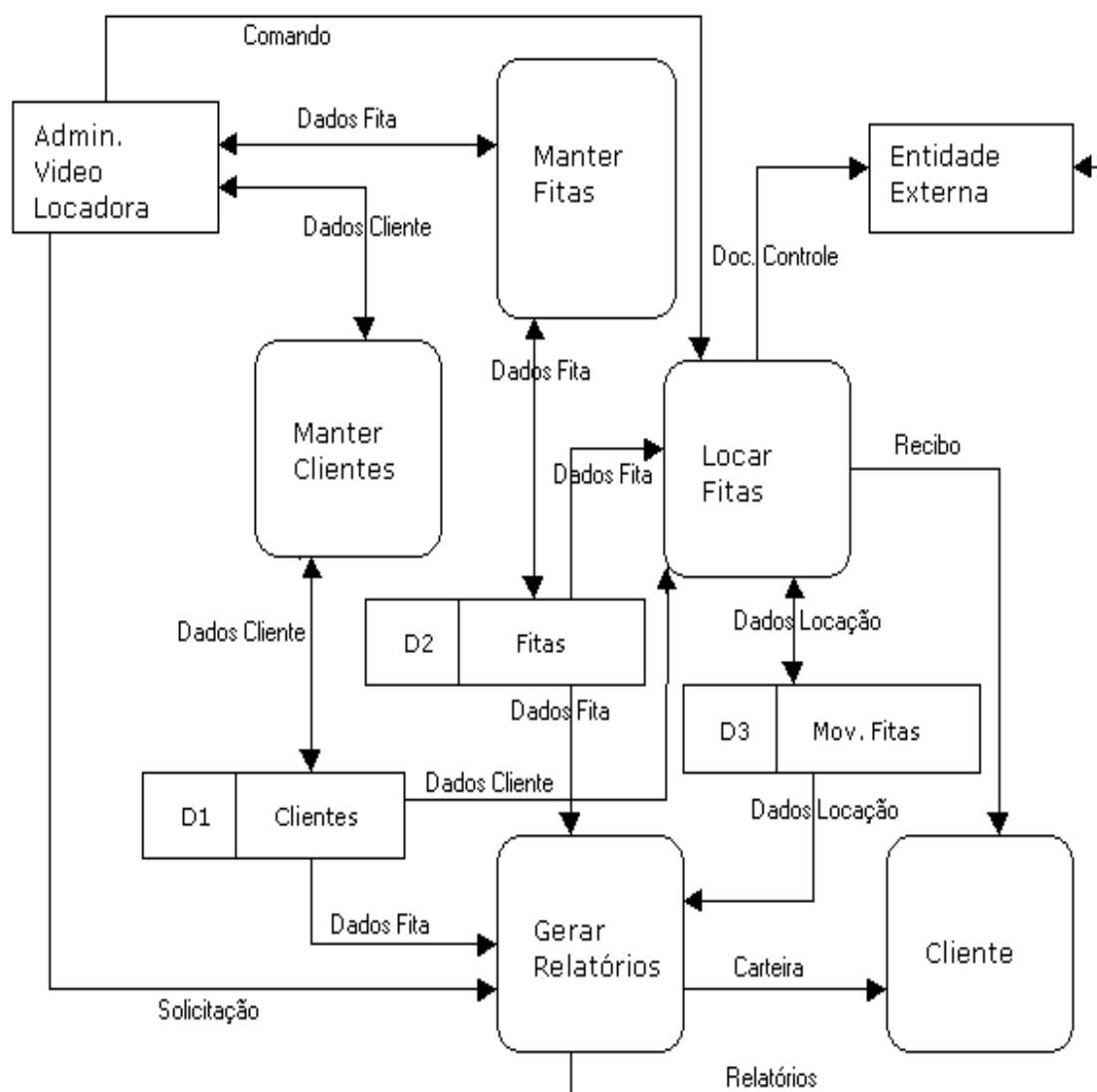


Figura 5 - Modelo DFD

8. Dicionário de Dados

Segundo José V. de Oliveira (2000), O dicionário de dados pode ser visto como um depósito central que descreve e define o significado de toda a informação usada na construção de um sistema. Permite fazer a verificação de consistência entre os vários modelos.

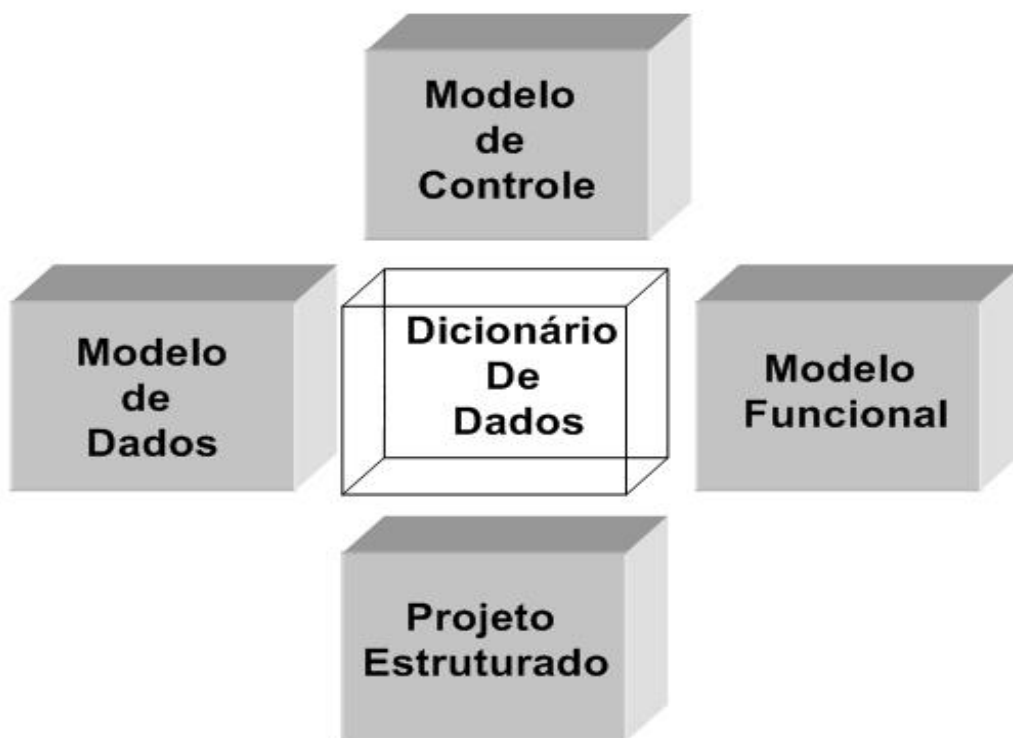


Figura 6 - Dicionário de Dados

8.1. Características dos Dicionários de Dados

Para José V. de Oliveira (2000) O dicionário de dados é relativo ao DFD, o conteúdo de um dicionário de dados é composto pela:

- Especificação dos fluxos de dados;
- Especificação de arquivos;

- Especificação de processos;
- O seu conteúdo deve ser preciso, conciso e não redundante;
- Cada ocorrência deve contemplar, pelo menos, os seguintes aspectos:
 - Nome, e.g., detalhes_de_quarto;
 - Tipo, e.g., fluxo;
 - Descrição, e.g., conjunto de dados que caracterizam um quarto;
 - Pseudónimos (outros nomes possíveis)
 - Especificação, e.g., detalhes_de_quarto = no + tipo + estado + extensão_telefonica + ([no_de_camas | lotação]);
 - Comentários significativos, que poderem incluir ao volume,
 - Frequência, política de partilha, segurança dos dados.

8.2. Especificação de dados

Notação Típica

Símbolo	Descrição	Alternativa
=	Composição	
+	Concatenação	
{ }	Iteração: 0 ou mais ocorrências	
[]	Seleção de 1 entre os presentes	
()	Opção	
“ ”	Valor Discreto	
@	Chave de identificação	Sublinhado
	Separador de alternativas	,
**	Comentário	

Exemplo

Nome = titulo _de_cortesia +(primeiro_nome)+ultimo_ nome

titulo _de_cortesia =[Sr.|Sra.|Prof.|Dr.|Arq.|Eng.]

segundo nome=2{caracter}12

ultimo_nome= {caracter}

caracter = [A-Z| a-z| |-]

8.3. Despite de Inconsistências

Em vistas de Alterações nos modelos sem a manutenção adequada no dicionário de dados José V. de Oliveira (2000) afirma que podem dar origem a:

8.3.1.Pseudônimos

Nomes diferentes para a mesma especificação de dados.

8.3.2.Impostores

Dados com o mesmo nome, mas com especificações diferentes.

8.3.3.Órfãos

Dados que não pertencem a nenhum componente do sistema e desconhece-se a sua origem e o seu destino.

9. Diagrama de Entidade Relacionamento (DER)

O DER é um modelo em rede que descreve a diagramação dos dados armazenados de um sistema em alto nível de abstração. Ele é inteiramente diferente de um diagrama de fluxo de dados que modela as funções executadas por um sistema e é diferente do diagrama de transições de estado, que modela o comportamento tempo-dependente de um sistema assim conclui Maria A. Freire (2006).

9.1. Tipos de objetos

Um tipo de objeto representa uma coleção ou um conjunto de objetos (coisas) do mundo real cujos membros individuais (exemplares ou instâncias) têm as seguintes características:

- Cada um deles só pode ser identificado de uma única forma.
- Cada um exerce um papel no sistema em construção. Isto é, para que o tipo de objeto seja legítimo é necessário que o sistema não possa funcionar sem acesso aos membros desse tipo de objeto.
- Cada um pode ser descrito por um ou mais elementos de dados. Desse modo, um cliente pode ser descrito por esse elemento de dados pelo nome, endereço, limite de crédito e número do telefone.

9.2. Relacionamentos

Os objetos são interligados por relacionamentos. Onde um relacionamento representa um conjunto de conexões entre objetos. É importante reconhecer que o relacionamento representa um conjunto de conexões.

Um relacionamento pode interligar duas ou mais instâncias do mesmo objeto. Observe que o relacionamento representa alguma coisa que deve ser recordada pelo sistema e também pode haver mais de um relacionamento entre dois objetos e, em alguns casos, podemos ter relacionamentos entre diferentes instâncias do mesmo tipo de objeto, mas uma situação mais corriqueira é haver múltiplos relacionamentos entre múltiplos objetos.

9.3. Notação Alternativa Para Relacionamentos

Os relacionamentos no sistema DER são multidirecionais; podem ser lidos em qualquer direção e também já foi dito que o diagrama DER não apresenta cardinalidade, isto é, não mostra o número de objetos que participam de um relacionamento. Uma notação alternativa usada por alguns analistas de sistemas mostra tanto a cardinalidade quanto a ordinalidade. Outra notação de uso comum é a seta que quando tem a ponta dupla é utilizada para indicar um relacionamento um-para-muitos, enquanto a seta de ponta singela é empregada para indicar relacionamentos um-para-um entre objetos.

9.4. Indicadores de tipos de objetos associativos

Uma notação especial em diagramas DER é o indicador de tipos de objetos associativos, ele representa alguma coisa que funciona tanto como um objeto quanto como um relacionamento. Outro modo de encarar o tipo de objeto associativo é considerar que ele representa um relacionamento sobre o qual queremos manter algumas informações.

10. Banco de dados

Gabriel Torres (2007) relata que bancos de dados ou bases de dados são conjuntos de dados com uma estrutura regular que organizam informação.

Essas estruturas costumam ter a forma de tabelas: cada tabela é composta por linhas e colunas. Informações utilizadas para um mesmo fim são agrupadas num banco de dados.

Em sistemas computacionais, bases de dados são geridas por um sistema gestor de bancos de dados, ou SGBD. A apresentação dos dados pode ser semelhante à de uma planilha eletrônica, porém os sistemas de gestão de banco de dados possuem características especiais para o armazenamento, classificação e recuperação dos dados.

Existe uma grande variedade de bancos de dados, desde exemplos simples como uma coleção de tabelas, até um modelo formalmente definido como o relacional. Os bancos de dados são diferenciados por muitas características. A mais útil e usada é o modelo de programação.

O modelo plano, ou tabular, é basicamente uma matriz bi-dimensional de elementos de dados na qual todos os membros de uma dada coluna possuem valores de mesmo tipo, e todos os membros de uma linha estão relacionados entre si. Por exemplo, uma tabela de um banco de dados para segurança do sistema pode ter colunas de nome e de senha; cada linha deve ter a senha específica associada a cada um dos usuários.

O modelo em rede amplia o modelo de tabela permitindo a adição de múltiplas tabelas. Uma coluna de tabela pode ser definida como uma referência a uma ou mais entradas de uma tabela diferente. Assim, as tabelas são relacionadas por meio de referências, o que pode ser visualizado como uma estrutura de rede. Um subconjunto particular do modelo de rede, o modelo hierárquico, limita os relacionamentos a uma estrutura de árvore, ao contrário da estrutura aplicada pelo modelo de rede completo.

De acordo com a arquitetura ANSI / SPARC em três níveis, os bancos de dados relacionais possuem três camadas: um conjunto de visões compondo o nível externo; uma coleção de estruturas de dados, a saber relações, compondo o nível conceitual; um conjunto de índices ou métodos de acesso a dados armazenados, compondo o nível interno.

A teoria relacional de banco de dados define um conjunto de operações lógicas, a saber, a álgebra e o cálculo relacionais. Essas operações são a base da linguagem SQL.

Um dos pontos fortes do modelo relacional de banco de dados é a possibilidade de definição de um conjunto de restrições de integridade. Estas definem os conjuntos de estados e mudanças de estados consistentes do banco de dados, determinando os valores que podem e os que não podem ser armazenados.

Diferentemente dos bancos de dados em rede, nos bancos de dados relacionais os relacionamentos entre as tabelas não são codificados explicitamente na sua definição. Em vez disso, se fazem implicitamente pela presença de atributos chave.

Como resultado, bancos de dados relacionais podem ser reorganizados e utilizados de maneira flexível e de formas não previstas pelos projetistas originais. Por causa dessa flexibilidade, muitos bancos de dados são baseados no modelo relacional, embora imperfeitamente.

Todos os tipos de bancos de dados podem ter seu desempenho melhorado pelo uso de índices. O tipo mais comum de índice é uma lista ordenada dos valores de uma coluna de uma tabela, contendo ponteiros para as linhas associadas a cada valor. Um índice permite que o conjunto das linhas de uma tabela que satisfazem determinado critério sejam localizados rapidamente. Há vários métodos de indexação utilizados comumente, como árvores B, hashes e listas encadeadas.

Em anos recentes, o modelo baseado na orientação a objeto vem sendo aplicado também aos bancos de dados, criando um novo modelo de programação conhecido como bancos de dados orientados a objeto.

Os objetos são valores definidos segundo classes, ou tipos de dados complexos, com seus próprios operadores (métodos). Com o passar do tempo, os sistemas gestores de bancos de dados orientados a objeto e os bancos de dados relacionais baseados na linguagem SQL se aproximaram.

Muitos sistemas orientados a objeto são implementados sobre bancos de dados relacionais baseados em linguagem SQL.

Os bancos de dados são utilizados em muitas aplicações, abrangendo praticamente todo o campo dos programas de computador. Os bancos de dados são o método de armazenamento preferencial para aplicações multiusuário, nas quais é necessário haver coordenação entre vários usuários. Entretanto, são convenientes também pa-

ra indivíduos, e muitos programas de correio eletrônico e organizadores pessoais baseiam-se em tecnologias padronizadas de bancos de dados.

Um banco de dados é um conjunto de informações com uma estrutura regular.

Um banco de dados é normalmente, mas não necessariamente, armazenado em algum formato de máquina lido pelo computador. Há uma grande variedade de bancos de dados, desde simples tabelas armazenadas em um único arquivo até gigantescos bancos de dados com muitos milhões de registros, armazenados em salas cheias de discos rígidos.

Bancos de dados caracteristicamente modernos são desenvolvidos desde os anos da década de 1960. Um pioneiro nesse trabalho foi Charles Bachman. A maneira mais prática de classificar bancos de dados é de acordo com o modelo de programação associado ao banco de dados.

Diversos modelos foram utilizados ao longo da história, por determinados períodos. Historicamente, o modelo de bancos de dados hierárquico foi implementado primeiro; então surgiu o modelo de bancos de dados em rede; daí o modelo de bancos de dados relacional surgiu e ganhou destaque, acompanhado daquilo que é chamado modelo plano (tabular) para fins mais diretos e simples.

Os dois primeiros e o último nunca foram descritos teoricamente, e são classificados como modelos de dados unicamente em contraste com o modelo relacional (que tem uma teoria de suporte). Esses três modelos sem teoria própria surgiram basicamente a partir de estruturas e modelos de programação, não modelos de dados.

10.1.Banco de Dados Sistema Eclipse

O banco de dados ao qual esta sendo utilizado para a construção do Sistema Eclipse é o Microsoft Office Access 2000, devido a termos de planejamento do Espaço Cultural Flávio Fernandes de Matos e visando o custo (vesus) benefício , utilizamos um banco de dados no qual sua instituição já possui licença.

10.1.1. Histórico Access 2000

Em 1999 é lançada a versão 9.0 pela Microsoft, também conhecido como MS Access 2000, com suporte a OLE DB e o relacionamento com bases de dados corporativas.

10.1.1.1. Requisitos de hardware para o Access 2000

De acordo com a Microsoft (2002), para a execução do Access 2000 é necessária a configuração de Hardware igual ou superior aos seguintes requisitos:

- Pentium 75-megahertz (MHz) ou processador superior.
- Microsoft Windows 95 ou sistema operacional posterior ou Microsoft Windows NT Workstation versão 4.0, Service Pack 3 ou posterior.
- Para o Windows 95 ou Windows 98: 16 megabytes (MB) de RAM para o sistema operacional, mais 8 MB adicionais de RAM para o Access.
- Para o Windows NT Workstation: 32 MB de RAM para o sistema operacional, mais 8 MB adicionais de RAM para o Access.
- Para o Windows 2000 Professional: 64 MB de RAM para o sistema operacional, mais 8 MB adicionais de RAM para o Access.

- Observação: Windows 2000 requer um Pentium de 133 megahertz (MHz) ou processador superior.
- Unidade de CD-ROM.
- Resolução VGA ou adaptador de vídeo mais alto. É recomendável VGA (SVGA) super 256 cores.
- Microsoft Mouse, Microsoft IntelliMouse ou dispositivo apontador compatível.

10.1.1.2. Requisitos de software para o access 2000

Microsoft Access 2000 foi projetado para ser executado nos seguintes sistemas operacionais:

- Microsoft Windows 95
- Microsoft Windows 98
- Microsoft Windows NT (r) Workstation versão 4.0, Service Pack 3 ou posterior
- Microsoft Windows 2000
- Microsoft Windows XP
- Redes com suporte. O Microsoft Access 2000 suporta as seguintes redes:
 - Microsoft Windows 95 e 98 / Microsoft Windows NT e Windows 2000 / Novell NetWare / Microsoft Windows XP

10.1.1.3. Espaço em disco rígido Access 2000

Microsoft Access 2000 requer 161 MB de espaço disponível em disco. (Esse número indica uma instalação típica; requisitos de espaço em disco rígido variam dependendo da configuração. Escolhas feitas durante a instalação personalizada podem exigir mais ou menos espaço em disco.)

11. Linguagens de Programação

Definição de LP por Gabriel Andrade (2007), onde uma linguagem de programação é um conjunto finito de símbolos com os quais se escrevem programas de computador. Há várias classificações para as linguagens de programação. Citaremos três delas, que são as mais utilizadas.

A primeira classificação relaciona-se com a proximidade que a linguagem tem do usuário final, isto é, do homem. Baseando-nos neste critério as linguagens podem ser de alto nível, nível intermediário ou baixo nível.

Linguagem de alto nível é uma linguagem que se aproxima mais da linguagem utilizada pelo ser humano. Exemplos típicos são Pascal, COBOL, C e SQL.

É importante lembrar que o Delphi e o Visual Basic são ambientes de programação e não linguagens no sentido estrito da definição. O Delphi tem por linguagem base uma extensão do Pascal chamada Pascal Orientado a Objetos ou Object-Pascal e o Visual Basic tem por base uma linguagem que é uma extensão do Basic de nome Visual Basic.

Linguagem de nível intermediário é uma linguagem que, como o próprio nome diz, fica em um nível intermediário entre a linguagem de alto nível e a linguagem de baixo nível. São códigos chamados de mnemônicos, mais conhecidos como *assembly*. Exemplos típicos são o *assembly* do processador 8085, do Pentium, do AMD K6 e do Motorola.

Linguagem de baixo nível é o código que o computador executa diretamente. É composta de conjunto de 0 e 1 e conhecida como linguagem binária.

Uma segunda classificação, também muito utilizada, é a que se relaciona com o nível de abstração que o usuário utiliza para escrever programas na linguagem. Baseando-nos neste critério as linguagens podem ser procedimentais ou não-procedimentais.

Numa linguagem de programação procedimental o usuário deve descrever, comando a comando, como o programa trabalhará para chegar ao fim desejado. Exemplos típicos são Pascal, COBOL e C.

Uma linguagem de programação não-procedimental o usuário deve descrever o que o programa executará, mas não como fará isso. Exemplo típico é a linguagem SQL.

11.1. Conceitos de Linguagem de Programação

Jessé Bragança (2005) aplica conceitos sobre LP's de forma que esclareça dúvidas que surgem em torno do assunto.

Sintática: Uma linguagem de programação é uma notação utilizada pelo programador para especificar ações a serem executadas por um computador

Semântica: Uma linguagem de programação compreende um conjunto de conceitos que um programador usa para resolver problemas de programação

11.2.Domínios das linguagens de programação

- Aplicações científicas
- Aplicações comerciais
- Inteligência artificial
- Sistemas básicos
- Aplicações Internet

11.3.Critérios de linguagem de programação

- Legibilidade
- Simplicidade
- Expressividade
- Ortogonalidade
- Confiabilidade
- Portabilidade

11.3.1. Característica: legibilidade

- Facilidade de ler e escrever programas
- Legibilidade influi:
- Desenvolvimento e depuração de programas
- Manutenção de programas
- Desempenho de equipes de programação

11.3.1.1. Fatores que melhoram a legibilidade

- Abstração de dados
- Comandos de controle
- Modularização de programas
- Documentação
- Convenções léxicas, sintaxe e semântica
- Exemplo em Java: nomes de classes iniciam por letra maiúscula, nomes de campos usam letras minúsculas

11.3.2. Característica: simplicidade

- Representação de cada conceito seja simples de aprender e dominar
- Simplicidade sintática exige que a representação seja feita de modo preciso, sem ambigüidades
- Contra-exemplo: `A++; A=A+1; A+=1; ++A.`
- Simplicidade semântica exige que a representação possua um significado independente de contexto
- Contra-exemplo: `private: B b; class B: private A`
- Simplicidade não significa concisão
- A linguagem pode ser concisa, mas usar muitos símbolos especiais
- Exemplo: linguagens funcionais

11.3.3. Característica: expressividade

- Representação clara e simples de dados
- procedimentos a serem executados pelo programa
- Exemplo: tipos de dados em Pascal
- Expressividade
- concisão
- Muito concisa: falta expressividade?
- Muito extensa: falta simplicidade?
- Linguagens mais modernas:
- Incorporam apenas um conjunto básico de representações de tipos de dados e comandos
- Aumentam o poder de expressividade com bibliotecas de componentes
- Exemplos: Pascal, C++ e Java

11.3.4. Característica: ortogonalidade

- Possibilidade de combinar entre si, sem restrições, os componentes básicos da LP
- Exemplo: permitir combinações de estruturas de dados, como *arrays* de registros
- Contra exemplo: não permitir que um *array* seja usado como parâmetro de um procedimento
- Componente de primeira ordem: pode ser livremente usado em expressões, atribuições, como argumento e retorno de procedimentos
- Influenciada pelo modelo de LP

- Modelo de Objetos: objeto
- Modelo funcional: funções

11.3.5. Característica: portabilidade

- Multiplataforma:
- Capacidade de um software rodar em diferentes plataformas sem a necessidade de maiores adaptações
- Sem exigências especiais de hardware/software
- Exemplo: aplicação compatível com sistemas Unix e Windows

11.3.5.1. Longevidade

- O Ciclo de vida útil do software e o do hardware não precisa ser síncrono; ou seja, é possível usar o mesmo software após a mudança de hardware
- Característica: confiabilidade
- Mecanismos que facilitem a produção de programas que atendam às suas especificações
- Tipagem forte: o processador da linguagem deve
- Assegurar que a utilização dos diferentes tipos de dados seja compatível com a sua definição
- Evitar que operações perigosas, tal como aritmética de ponteiros, seja permitida
- Tratamento de exceções: sistemas de tratamento de exceções permitem construir programas que
- Possuam definições de como proceder em caso de comportamento não usual

- Possibilitem tanto o diagnóstico quanto o tratamento de erros em tempo de execução
- Evolução de Linguagens de Programação
- Década de 70: engenharia de software
- Abstração de dados: definição de tipos
- Abstração de controle: comandos, procedimentos
- Inicia preocupação com programação em larga escala: módulos e programação estruturada.

11.3.5.1.1. Exemplos de linguagens populares:

- Uso acadêmico: ALGOL (algoritmos), Pascal (tipos de dados)
- Uso comercial: COBOL (arquivos), PL/I (uso amplo)
- Década de 80: modularização
- Ênfase em mecanismos de LP e abstrações
- Correção de programas: verificação de tipos, exceções
- Programação concorrente e distribuída e tempo real
- Programação baseada em objetos (TADs)
- Programação orientada a objetos (herança)
- Exemplos de linguagens
- Uso acadêmico: Pascal / Modula
- Programação de tempo real: Ada 83
- Orientada a objetos: Smalltalk
- Década de 90: base na estrutura
- Estruturação de dados: encapsulamento
- Estruturação da computação: classe

- Estruturação do programa
- classes e objetos
- Programação para Internet
- plataforma neutra
- Exemplos de linguagens
- Pascal / Delphi
- C / C++
- Ada83 / Ada95
- Java
- Visual Basic
- Lua

11.4.Ferramenta de Desenvolvimento do Sistema Eclipse

Visando o processo de análise do grupo F11, chegamos a conclusão que a ferramenta adequada para estarmos desenvolvendo o Sistema Eclipse seria o Delphi 7 da Borland Software Corporation, a escolha dessa ferramenta de desenvolvimento se baseou na análise das configurações de hardware e software dos equipamentos de Informática do Espaço Cultural Flávio Fernandes de Matos, onde se havia a necessidade de produzir um sistema que fosse leve e versátil. Essa ferramenta nos deu o melhor retorno em testes efetuados nas máquinas do Espaço Cultural.

11.4.1. Histórico Delphi 7

11.4.1.1. Projeto de Criação

De acordo com a Borland Software Corporation (1990 á 2002), onde em 1990 o sucesso marcante do Turbo Pascal (tanto para Dos como para Windows), que era a ferramenta de desenvolvimento e carro chefe da Borland, começava a dar sinais de que já não estava mais com esta bola toda. As causas? O Turbo Pascal for Windows não era uma ferramenta RAD e sim apenas um IDE de escrita de linguagem limitadíssimo e com alguns recursos extras, que se comparados aos IDE's atuais, seria visto como um mero bloco de notas incrementado, de tão pouco o que ele oferecia ao desenvolvedor.

Então em 1993, começou o desenvolvimento desta nova ferramenta baseada na linguagem Object Pascal e, portanto, orientada a objetos, e em meados de 1995, foi lançado no mercado à nova aposta da Borland para confrontar o Visual Basic for Windows e assim surgiu o Delphi.

11.4.1.2. Desenvolvedores

Anders Hejlsberg e Danny Thorpe - Borland Software Corporation

11.4.1.3. Lançamento

Ano de 2002

11.4.1.4. Informações Técnicas Delphi 7

- Compatibilidade
- Windows 95, 98, 2000, XP e Linux

- Internet/Intranet – IntraWeb
- Compatível com o Kylix – CLX

11.4.2. Arquivos

O Project que é a coleção de arquivos necessários para compilar o sistema, os Forms, onde será desenvolvida a "cara" do programa. As Units , código do form, sendo que para todo Form temos pelo menos uma Unit , mas temos Units sem form (códigos de procedures, funções, etc).

11.4.3. Prós e Contras

11.4.3.1. Prós

Daniel Silveira(2002) institui que Delphi é uma ferramenta de alto nível, relativamente fácil de usar e inicialmente baseado na linguagem Pascal . Delphi, em contraste com Pascal, não foi concebido essencialmente para fins educativos. Além do alto nível de linguagem, O Delphi também suporta características de baixo nível de programação. A linguagem de orientação a objeto tem funcionalidades de classe e interface baseada em polimorfismo, fazendo programas escritos em Delphi mais claramente definidos do que programas escritos em algumas outras linguagens que permitem utilizar herança múltipla. Metaclasses são a primeira classe de objetos. Os objetos são realmente feitas para os objetos (como em Java), então não há geralmente nenhuma necessidade de manualmente alocar memória para ponteiros e objetos similares ou técnicas necessárias em algumas outras linguagens.

11.4.3.2. Contrás

A cada nova versão do Delphi há tentativas para manter á compatibilidade, tanto quanto possível. Isto permite aos utilizadores em construir código sem se preocupar com qualquer erro na interface ou funcionalidade. No entanto, alguns criadores consideram que a atenção e preocupação para compatibilidade têm travado a evolução da linguagem Delphi, e levou a concepção antiquada e decisões no padrão classe bibliotecas (VCL / RTL).

11.4.3.3. Informações Gerais

Através de Rafael Li (2007), conclui-se que ao contrário do que se fala o Delphi não é uma linguagem de programação, mas sim uma ferramenta de desenvolvimento da ultima geração do Object Pascal. Na verdade, o Delphi nada mais é do que uma evolução natural do "Borland Turbo Pascal for Windows" que marcou época na sua geração.

12. Apresentação da Organização

A organização a qual será realizado o projeto é o Espaço Cultural Flávio Fernandes de Matos, através de seu histórico, organograma, sua visão e missão estaremos conhecendo a fundo esta associação assim dando continuidade a elaboração do projeto Eclipse.

Nome Fantasia:

Arco-Íris – Grupo de assistência a criança carente

Razão Social:

Espaço Cultural Flávio Fernandes de Matos - CNPJ: 07.376.150/0001-43

Endereço:

R. Flório, 582 - Chácara Mafalda – São Paulo – SP – (011)3578-7298

Responsável:

Antônio Fernandes de Matos – Presidente da Associação

12.1.Histórico

O Espaço Cultural Flávio Fernandes de Matos é um grupo de voluntários, criado no ano de 1998 e formalizado no ano de 2002, onde seu objetivo social principal é de assistência as crianças e adolescentes carentes da comunidade.

12.2. Organograma

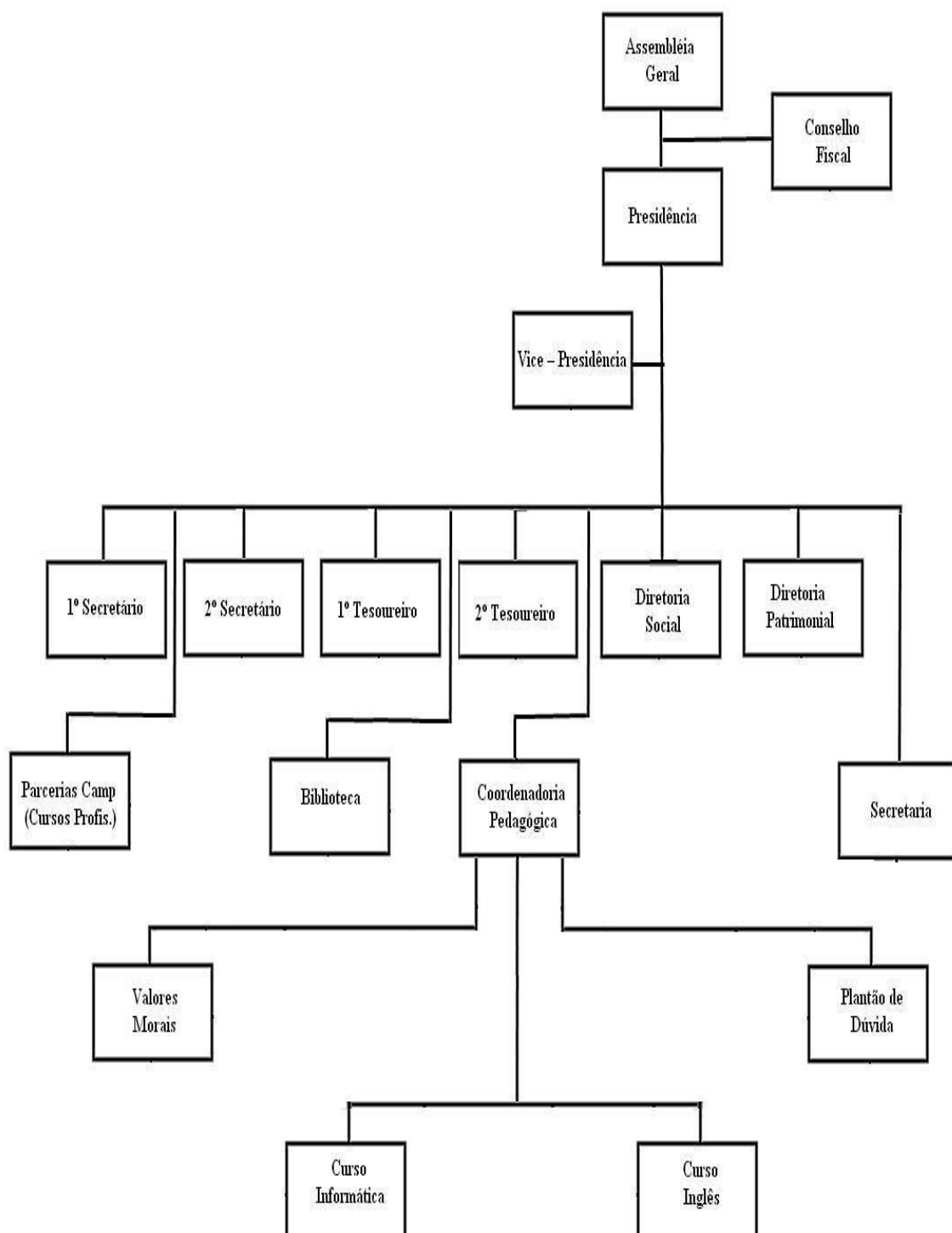


Figura 7 - Organograma Espaço Cultural Flávio Fernandes de Matos

12.3.Missão

A missão é uma declaração ampla e duradoura de propósitos que individualiza a organização e distingue o seu negócio impondo a delimitação de suas atividades dentro do espaço que deseja ocupar em relação às oportunidades de negócios.

(Herrera, 2007)

A missão do Espaço Cultural Flávio Fernandes de Matos é de promover a integração familiar, e seu desenvolvimento individual. Assim trazendo conceitos de comunidade, fraternidade e cidadania através de ações sociais, tendo como foco a criança carente, o público-alvo do grupo é a juventude e infância estabelecendo a responsabilidade social, de educação, arte e cultura.

12.4.Visão

Os grandes navegadores sempre sabem onde fica o norte. Sabem aonde querem ir e o que fazer para chegar a seu destino. Com as grandes empresas acontece a mesma coisa: elas têm visão. É isso que lhes permite administrar a continuidade e a mudança simultaneamente.

(PORRAS, JERRY / COLLINS, JAMES , 1998)

O Espaço Cultural Flávio Fernandes de Matos visa levar oportunidades de crescimento pessoal e de um futuro melhor para a nossa juventude. Tem como objetivo implementar seus cursos de maneira que haja um crescimento favorável para que ocorra um melhor atendimento a comunidade carente.

13. Processo de Análise

O processo de análise consistiu no levantamento de dados do sistema de trabalho manual atual, elaborado sobre os prós e os contras do projeto.

13.1.Descrição do funcionamento do sistema atual

O sistema atual se baseia em trabalho manual, tanto a seção inicial quanto sua fase de armazenamento, sem opções de segurança ocasionando as possíveis perca de dados. O mesmo ocupa a questão de espaço e tempo da associação. Conforme seguimento da análise foi perceptível que o funcionamento deste sistema era impróprio para a visão de expansão do Espaço Cultural Flávio Fernandes de Matos.

13.2.DFD – Sistema de Trabalho Manual Atual

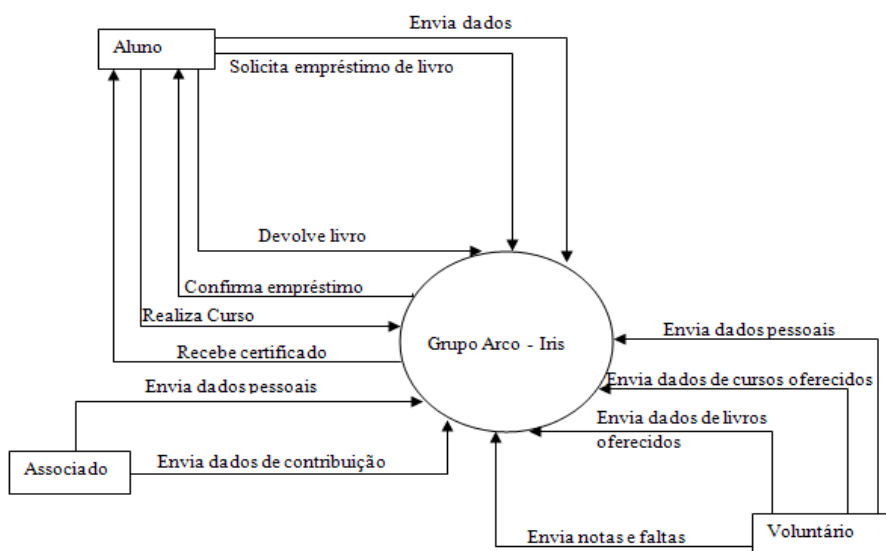


Figura 8 - DFD nível 0 Sistema de trabalho manual

13.2.1. Lista de Eventos – Sistema de Trabalho Manual Atual

Tabela 5 - Lista de Eventos Sistema de Trabalho Manual

Eventos	Estimulo	Ação/Resposta
Aluno envia dados pessoais	Enviar dados do aluno	Cadastrar aluno
Voluntario envia dados pessoais	Enviar dados do voluntario	Cadastrar voluntario
Associado envia dados	Enviar dados de contribuinte	Cadastrar associado
Associado envia dados da contribuição	Cadastrar contribuição	Cadastrar contribuição
Voluntario envia dados dos cursos oferecidos	Cursos oferecidos	Cadastrar cursos oferecidos
Voluntario envia dados do livro	Livros existentes na biblioteca	Cadastrar livro
Aluno solicita empréstimo de livro	Gerenciar empréstimos	Confirmação de empréstimo
Aluno devolve livro	Gerenciar devolução	Confirmação de devolução
Voluntario envia notas e faltas	Controle de faltas e notas para emissão de certificado	Cadastrar notas e faltas
Emissão de certificado	Emissão de certificado	Emite certificado

13.3. Concepção do Sistema Eclipse

A concepção do sistema foi à etapa fundamental para a análise do funcionamento do Espaço Cultural Flávio Fernandes de Matos, assim será elaborado o novo sistema sobre o conceito do que é necessário para a criação do projeto.

13.3.1. Levantamento do Sistema atual (Pontos Críticos)

Atualmente a instituição não possui um programa que armazene os dados necessários, tais como ficha do aluno, ficha de voluntários, diários de classe, empréstimos de livros, suas informações são guardadas em dispositivos de armazenamento, e em planilhas do Excel assim como o cadastro de livros da biblioteca. O controle de

faltas, tanto de alunos quanto os de voluntários são lançados em papéis, não possui cópia de segurança, permitindo perda de dados.

13.3.2. Relatório de Necessidades

O Espaço Cultural Flávio Fernandes de Matos necessita de um software, o qual organize de maneira segura e que viabilize os dados, realize o cadastro de voluntários, interessados, alunos, livros, associados, cursos e locadores. Ajudando o funcionamento no dia-a-dia da fundação onde sejam seguros de forma que faça backup, possuindo níveis de segurança e que trabalhe em rede.

13.4.Cronograma Projeto Eclipse

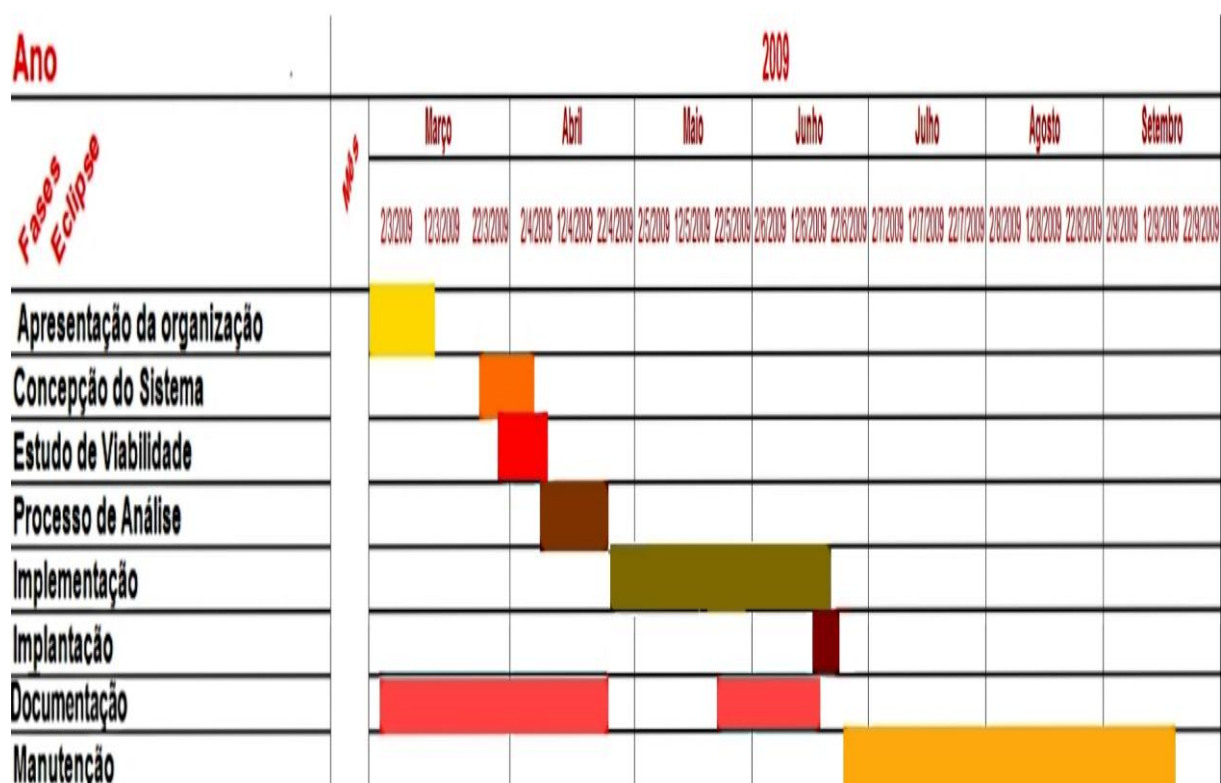


Figura 9 - Cronograma Projeto Eclipse - Grupo F11

14. Projeto do Sistema Eclipse

O projeto Eclipse apresentará soluções consideradas necessárias para suprir as necessidades do Espaço Cultural Flávio Fernandes de Matos

14.1.Proposta do novo Sistema

O sistema fará o cadastro de alunos, voluntários, livros, cursos, associados e as respectivas áreas que irá complementar estes cadastros. Onde também controlará notas, as faltas tanto de alunos como de voluntários, as locações e devoluções de livros, as aberturas e enceramentos da turma, as contribuições entre outros itens. O sistema emitirá relatórios possibilitando o controle dos dados da organização.

14.2.Descrição de fluxo de dados Sistema Eclipse

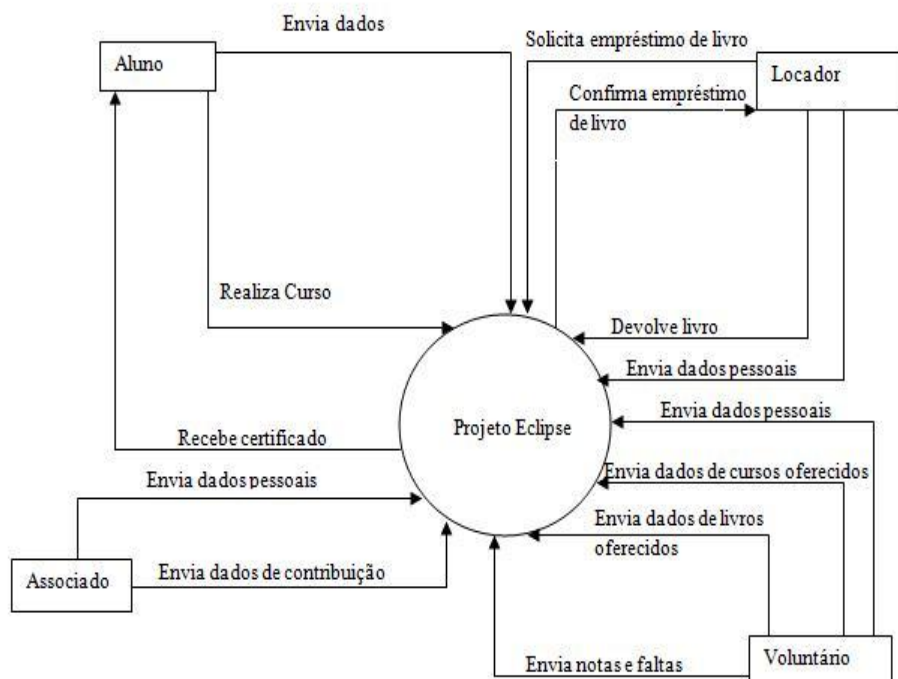


Figura 10 - DFD nível 0 Sistema Eclipse

14.2.1. DFD Ascendente Sistema Eclipse

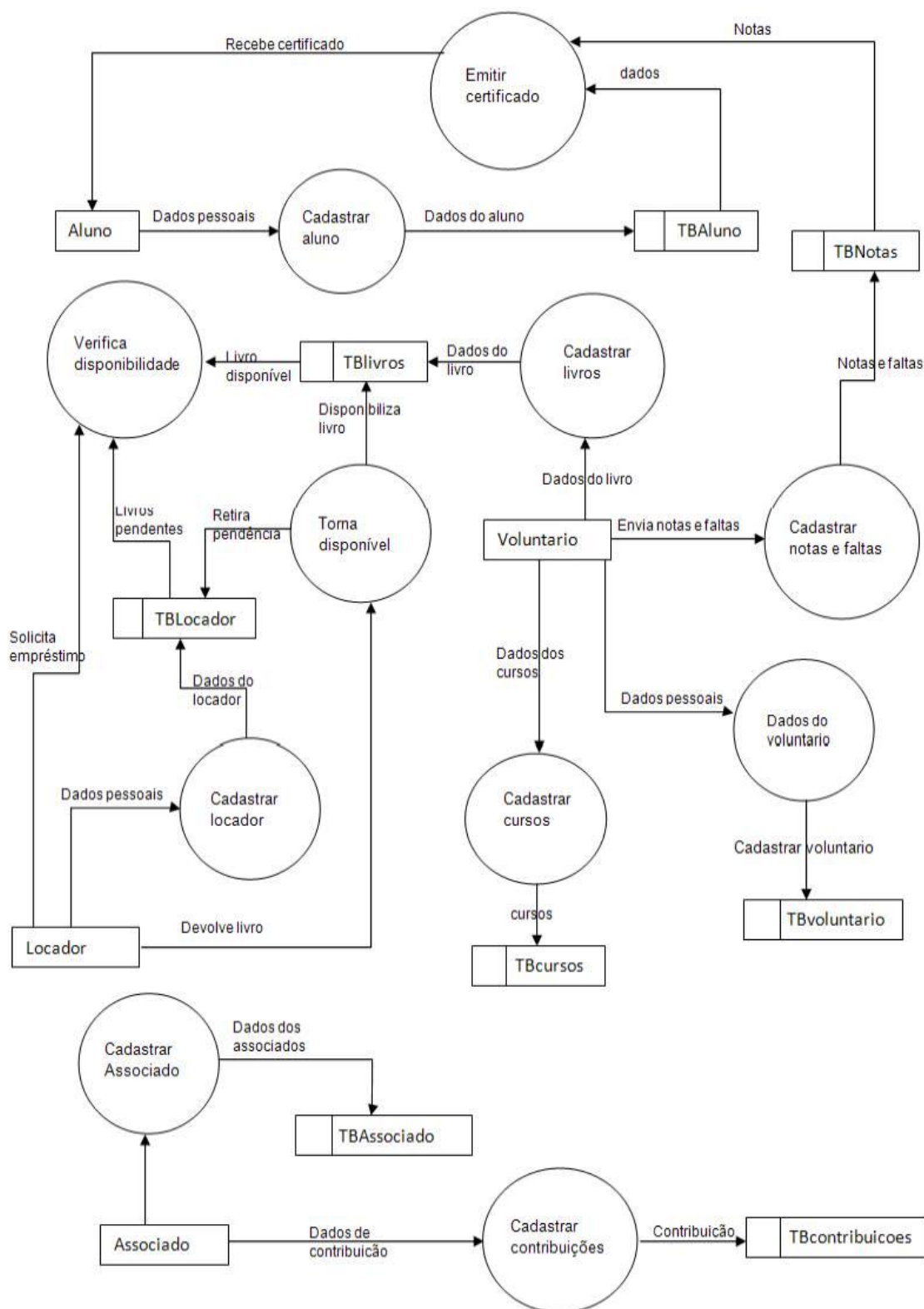
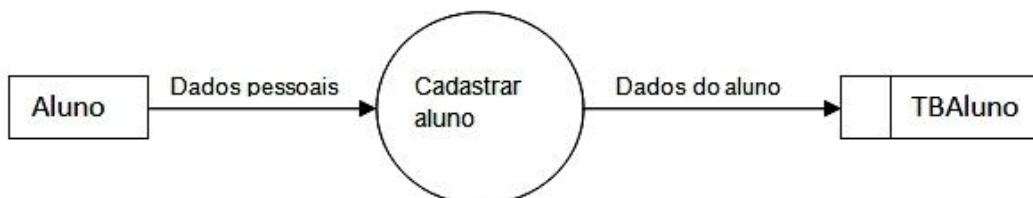


Figura 11 - DFD Ascendente Sistema Eclipse

14.2.2. DFD – Individual Sistema Eclipse

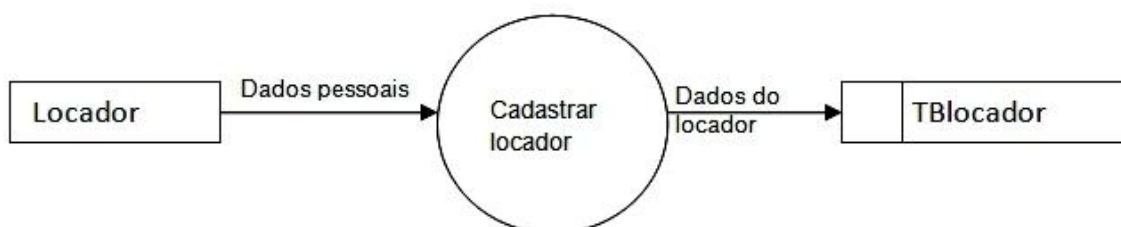
Primeiro Evento: Cadastrar Aluno



Segundo Evento: Cadastrar Voluntário



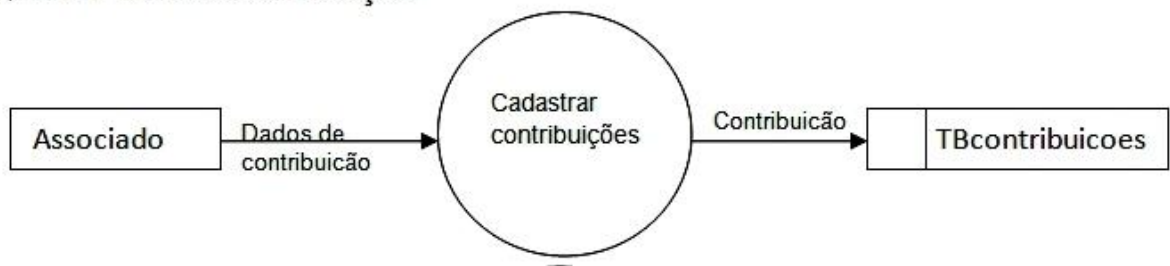
Terceiro Evento: Cadastrar Locador



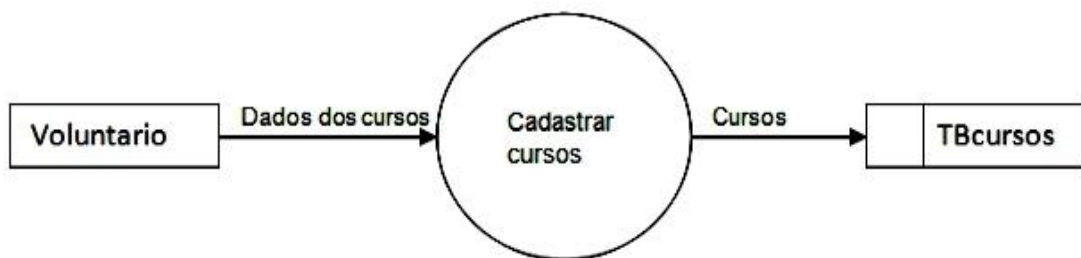
Quarto Evento: Cadastrar Associado



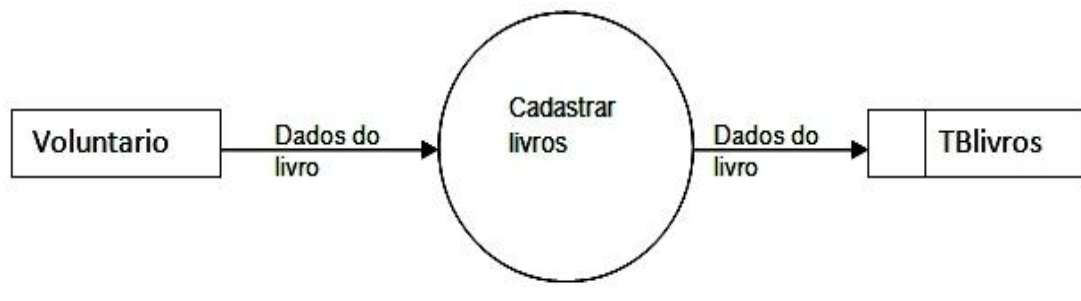
Quinto Evento: Contribuição



Sexto Evento: Cadastrar cursos



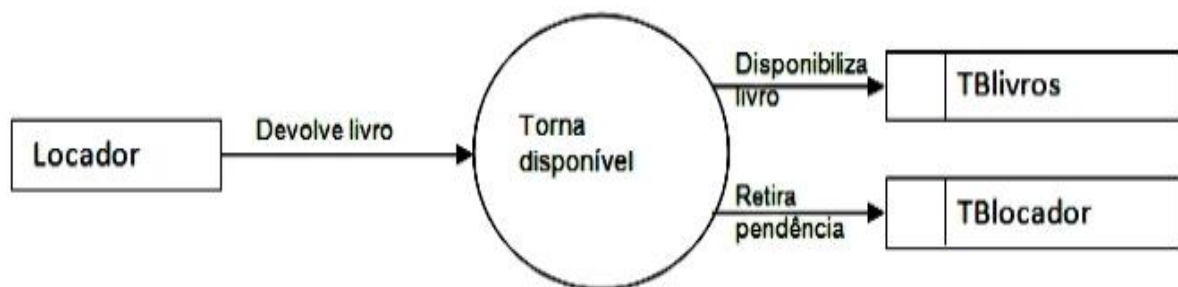
Sétimo Evento: Cadastrar livros



Oitavo Evento: Empréstimo de livro



Nono Evento: Devolução de livro



Décimo Evento: Cadastrar notas e faltas



Décimo Primeiro Evento: Emissão de certificado

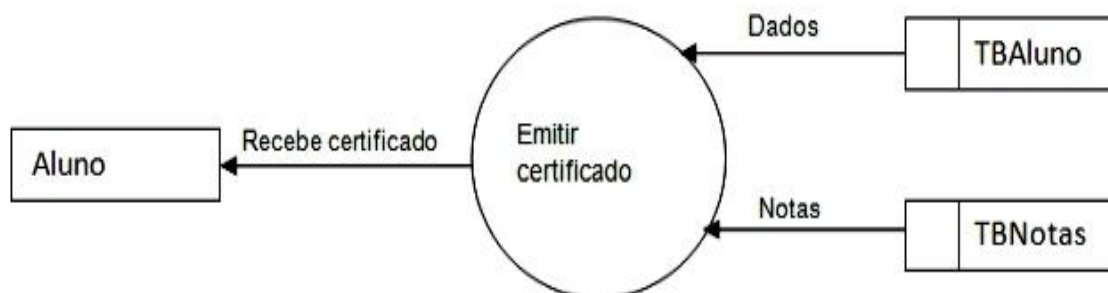


Figura 12- DFD Individual Sistema Eclipse

14.2.3. Lista de Eventos – Sistema a ser implantado

Tabela 6 - Lista de eventos Sistema Eclipse

Eventos	Estimulo	Ação/Resposta
Aluno envia dados pessoais	Enviar dados do aluno	Cadastrar aluno
Voluntario envia dados pessoais	Enviar dados do voluntario	Cadastrar voluntario
Locador envia dados pessoais	Enviar dados do locador	Cadastrar locador
Associado envia dados pessoais	Enviar dados de contribuinte	Cadastrar associado
Associado envia dados da contribuição	Cadastrar contribuição	Cadastrar contribuição
Voluntario envia dados dos cursos oferecidos	Cursos oferecidos	Cadastrar cursos oferecidos
Voluntario envia dados do livro	Livros existentes na biblioteca	Cadastrar livro
Locador solicita empréstimo de livro	Gerenciar empréstimos	Confirmação de empréstimo
Locador devolve livro	Gerenciar devolução	Confirmação de devolução
Voluntario envia notas e faltas	Controle de faltas e notas para emissão de certificado	Cadastrar notas e faltas
Emissão de certificado	Emissão de certificado	Emite certificado

15. Dicionário de Dados Sistema Eclipse

O dicionário de dados é uma ferramenta essencialmente textual que define o significado de toda a informação que entra, sai e é transformada pelo sistema. A sua construção e manutenção é uma das atividades morosa, mas crucial.

15.1.Tabela Voluntário

Tabela 7 - Tabela Voluntário Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_vol	AutoNumeração	Inteiro Longo	Código do voluntário
dtadm_vol	Data/Hora	Data abreviada	Data Admissão do voluntário
nome_vol	Texto	50	Nome do voluntário
sx_vol	Texto	1	Sexo do voluntário
dtnasc_vol	Data/Hora	Data abreviada	Data de Nascimento do voluntário
rg_vol	Texto	15	RG do voluntário
cpf_vol	Texto	15	CPF do voluntário
nacional_vol	Texto	15	Nacionalidade do voluntário
natural_vol	Texto	25	Naturalidade do voluntário
end_vol	Texto	35	Endereço do voluntário
num_vol	Numero	Inteiro Longo	Número da Residência do voluntário
compl_vol	Texto	35	Complemento do voluntário
bairro_vol	Texto	20	Bairro do voluntário
cep_vol	Texto	10	CEP do voluntário
cid_vol	Texto	20	Cidade do voluntário
uf_vol	Texto	2	Estado do voluntário
email_vol	Texto	50	Endereço de e-mail do voluntário
escolar_vol	Texto	15	Escolaridade do voluntário
frmcao_vol	Texto	30	Formação do voluntário
profs_vol	Texto	25	Profissão do voluntário
aptid_vol	Texto	50	Talentos e aptidão do voluntário
trabassoci_vol	Texto	3	Interesse do voluntário em trabalhar na associação
area_vol	Texto	15	Área em que o voluntário Deseja Trabalhar
dias_vol	Texto	15	Dias Disponíveis Para o trabalho
hra_vol	Texto	15	Horário Disponível
indicacao_vol	Texto	25	Quem indicou o voluntário

15.2.Tabela Telefone Voluntário

Tabela 8 - Tabela Telefone Voluntário Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_vol	Numero	Inteiro Longo	Código do voluntário
tpfon_vol	Texto	15	Tipo de telefone do voluntário
fone_vol	Texto	15	Número de telefone do voluntário

15.3.Tabela Cargo

Tabela 9 - Tabela Cargo Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_vol	Numero	Inteiro Longo	Código do voluntário
cargo_vol	Texto	25	Cargo do voluntário

15.4.Tabela Aluno

Tabela 10 - Tabela Aluno Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_alu	AutoNumeração	Inteiro Longo	Código do aluno
nome_alu	Texto	50	Nome do aluno
sx_alu	Texto	1	Sexo do aluno
dtnasc_alu	Data/Hora	Data abreviada	Data de nascimento do aluno
rg_alu	Texto	15	RG do aluno
nacional_alu	Texto	15	Nacionalidade do aluno
natural_alu	Texto	25	Naturalidade do aluno
end_alu	Texto	35	Endereço do aluno
num_alu	Numero	Inteiro Longo	Número da residência do aluno
compl_alu	Texto	35	Complemento do aluno
bairro_alu	Texto	20	Bairro do aluno
cep_alu	Texto	10	CEP do aluno
cid_alu	Texto	20	Cidade do aluno
uf_alu	Texto	2	Estado do aluno
mae_alu	Texto	35	Nome da mãe

profmae_alu	Texto	25	Profissão da mãe
rgmae_alu	Texto	15	RG da mãe
cpfmae_alu	Texto	15	CPF da mãe
pai_alu	Texto	35	Nome do pai
profpai_alu	Texto	25	Profissão do pai
rgpai_alu	Texto	15	RG do pai
cpfpai_alu	Texto	15	CPF do pai
renda_alu	Numero	Inteiro Longo	renda da família
csap_alu	Texto	3	Casa própria da família?
vlalug_alu	Numero	Inteiro Longo	Se casa não for própria, valor aluguel
qt_resid_alu	Numero	Inteiro Longo	Quantidades de pessoas na residência
conhece_alu	Texto	100	Como Obteve conhecimento da associação
estuda_alu	Texto	255	Estuda?
tpesc_alu	Texto	12	Tipo de Escola
nomeesc_alu	Texto	25	Nome da Escola
serieesc_alu	Texto	15	Serie em que estuda
endesc_alu	Texto	35	Endereço da escola
bairroesc_alu	Texto	20	bairro da escola
cepesc_alu	Texto	10	CEP da escola
cidesc_alu	Texto	20	Cidade da escola
ufesc_alu	Texto	2	Estado da escola
dtcad_alu	Data/Hora	Data abreviada	Data do cadastro
codigo_vol	Numero	Inteiro Longo	Código do voluntário
foto	Objeto OLE		Foto do aluno
status	Texto	1	Status do aluno

15.5.Tabela telefone aluno

Tabela 11 - Tabela Telefone Aluno Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_alu	Numero	Inteiro Longo	Código do aluno
tpstel_fnalu	Texto	15	Tipo de telefone ex. residencial, celular
fone_fnalu	Texto	15	Número de telefone do aluno

15.6.Tabela Turma

Tabela 12 - tabela Turma Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_tur	AutoNumeração	Inteiro Longo	Código da turma
codigo_cur	Numero	Inteiro Longo	Código do curso
codigo_vol	Numero	Inteiro Longo	Código do voluntário
nome_tur	Texto	35	Nome da turma
dt_ini	Data/Hora	Data abreviada	Data de inicio da turma
dt_prev	Data/Hora	Data abreviada	Data prevista para o término da turma
dt_fin	Data/Hora	Data abreviada	Data de fim da turma
numvagas_tur	Numero	Inteiro	Número de vagas da turma
numalu_tur	Numero	Inteiro	Número de alunos da turma
obs_tur	Memorando		Observação sobre turma
status_tur	Texto	20	Status da turma

15.7.Tabela Lista de Espera

Tabela 13 - Tabela Lista de Espera Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_alu	Numero	Inteiro Longo	Código do aluno
codigo_dis	Numero	Inteiro Longo	Código da disciplina
dtficha_esp	Data/Hora	Data abreviada	Data que entrou para ficha de espera

15.8.Tabela Histórico

Tabela 14 - Tabela Histórico Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_alu	Numero	Inteiro Longo	Código do aluno
codigo_tur	Numero	Inteiro Longo	Código da turma
nfaltas	Texto	6	Número de faltas
status_alu	Texto	2	Status do aluno
obs_alu	Memorando		Observação sobre o aluno
mediafinal_alu	Texto	40	Média final do aluno

15.9.Tabela Menção

Tabela 15 - Tabela Menção Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
nota_men	Texto	10	Valor da Nota ex. A, B, C

15.10. Tabela Curso

Tabela 16 - Tabela Curso Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_cur	AutoNumeração	Inteiro Longo	Código do associado
nome_cur	Texto	25	Nome do curso

15.11. Tabela Associado

Tabela 17 - Tabela Associado Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_ass	AutoNumeração	Inteiro Longo	Código do associado
dtadm_ass	Data/Hora	Data abreviada	Data de admissão
nome_ass	Texto	50	Nome do associado
sx_ass	Texto	1	sexo do associado
dtnasc_ass	Data/Hora	Data abreviada	data de nascimento associado
rg_ass	Texto	15	RG associado
cpf_ass	Texto	15	CPF do associado
nacional_ass	Texto	15	Nacionalidade associado
natural_ass	Texto	25	Naturalidade do associado
end_ass	Texto	35	Endereço do associado
num_ass	Numero	Inteiro Longo	Número residência do associado
compl_ass	Texto	35	Complemento residência do associado
bairro_Ass	Texto	20	Bairro do associado
cep_ass	Texto	10	CEP do associado
cid_ass	Texto	20	Cidade do associado
uf_ass	Texto	2	Estado do associado
email_ass	Texto	50	E-mail do associado
escolar_ass	Texto	15	Escolaridade do associado

frmcao_ass	Texto	30	Formação do associado
aptid_ass	Texto	255	Aptidão do associado
conhece_ass	Texto	255	Como obteve conhecimento da associação
profis_ass	Texto	25	profissão do associado
indicacao_vol	Texto	35	Quem indicou?

15.12. Tabela Tipo Telefone

Tabela 18 - Tabela Tipo Telefone Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
sigla_tf	Texto	2	Sigla do Telefone
tpfone_tf	Texto	30	Tipo do telefone
status_tf	Texto	30	Status telefone

15.13. Tabela contribuição

Tabela 19 - - Tabela Contribuição Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_con	AutoNumeração	Inteiro Longo	Código contribuição
codigo_ass	Numero	Inteiro Longo	Código associado
vlr_con	Numero	Inteiro Longo	Valor contribuição
dtpagto_con	Data/Hora	Data abreviada	Data de pagamento da contribuição

15.14. Tabela Fone Associado

Tabela 20 - Tabela Fone Associado Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_ass	Numero	Inteiro Longo	Código Associado
tpfone_fnass	Texto	15	Tipo de Telefone Ex. Residencial
fone_fnass	Texto	15	Número de Telefone

15.15. Tabela Usuário

Tabela 21 - Tabela Usuário Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_usu	AutoNumeração	Inteiro longo	Código do usuário
nome_usu	Texto	15	Nome do usuário
senha_usu	Texto	20	Senha de usuário
nacesso_usu	Texto	2	numero de acesso do usuário
frase_usu	Texto	50	frase de usuário

15.16. Tabela Locador

Tabela 22 - Tabela Locador Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_loc	AutoNumeração	Inteiro Longo	Código do locador
nome_loc	Texto	50	Nome do locador
sx_loc	Texto	1	Sexo do locador
rg_loc	Texto	15	RG do locador
cpf_loc	Texto	15	CPF do locador
end_loc	Texto	35	Endereço do locador
cid_loc	Texto	20	Cidade do locador
uf_loc	Texto	2	Estado do locador
email_loc	Texto	50	Email do locador
block_loc	Texto	1	Bloqueio de Locação

15.17. Tabela Telefone Locador

Tabela 23 - Tabela Telefone Locador Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_loc	Numero	Inteiro Longo	Código do locador
tpfone_fnloc	Texto	15	Tipo de telefone
fone_fnloc	Texto	15	Número do telefone

15.18. Tabela Livros

Tabela 24 - Tabela Livros Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_tombo	Numero	Inteiro Longo	Código do tombo
nome_liv	Texto	50	Nome do livro
autor_liv	Texto	35	Autor da obra
sinopse_liv	Memorando		Sinopse da obra (breve descrição)
codigo_gen	Numero	Inteiro Longo	Código gênero
sub_cod_gen	Numero	Inteiro Longo	Código do sub gênero
disp_liv	Texto	20	Disponibilidade do livro

15.19. Tabela Gênero

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_gen	AutoNumeração	Inteiro Longo	Código genero
nome_gen	Texto	25	Nome do gênero
cor_gen	Texto	20	Cor do gênero

15.20. Tabela Locação

Tabela 25 - Tabela Locação Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_locacao	AutoNumeração	Inteiro Longo	Código de locação
codigo_tombo	Numero	Inteiro Longo	Código do tombo
codigo_loc	Numero	Inteiro Longo	Código do locador
dt_locado	Data/Hora	Data abreviada	Data de locação
dt_prev_locac	Data/Hora	Data abreviada	Data prevista para entrega
dtentg_locac	Data/Hora	Data abreviada	Data que o locador fez a devolução
pendencia_locac	Texto	100	Especifica se há ou não pendência de devolução
obs_locac	Memorando		Observações do locador ou livro em seu processo de locação

15.21. Tabela Multas

Tabela 26 - Tabela Multas Sistema Eclipse

NOME DO CAMPO	TIPO	TAMANHO	DESCRIÇÃO
codigo_mul	AutoNumeração	Inteiro Longo	Código da multa
codigo_loca	Numero	Inteiro Longo	Código do locador
multad_mul	Numero	Inteiro	Valor Multa dia
datrazo_mul	Numero	Inteiro	Dias de atraso
total_mul	Numero	Inteiro Longo	Total das multas

16. DER Sistema Eclipse

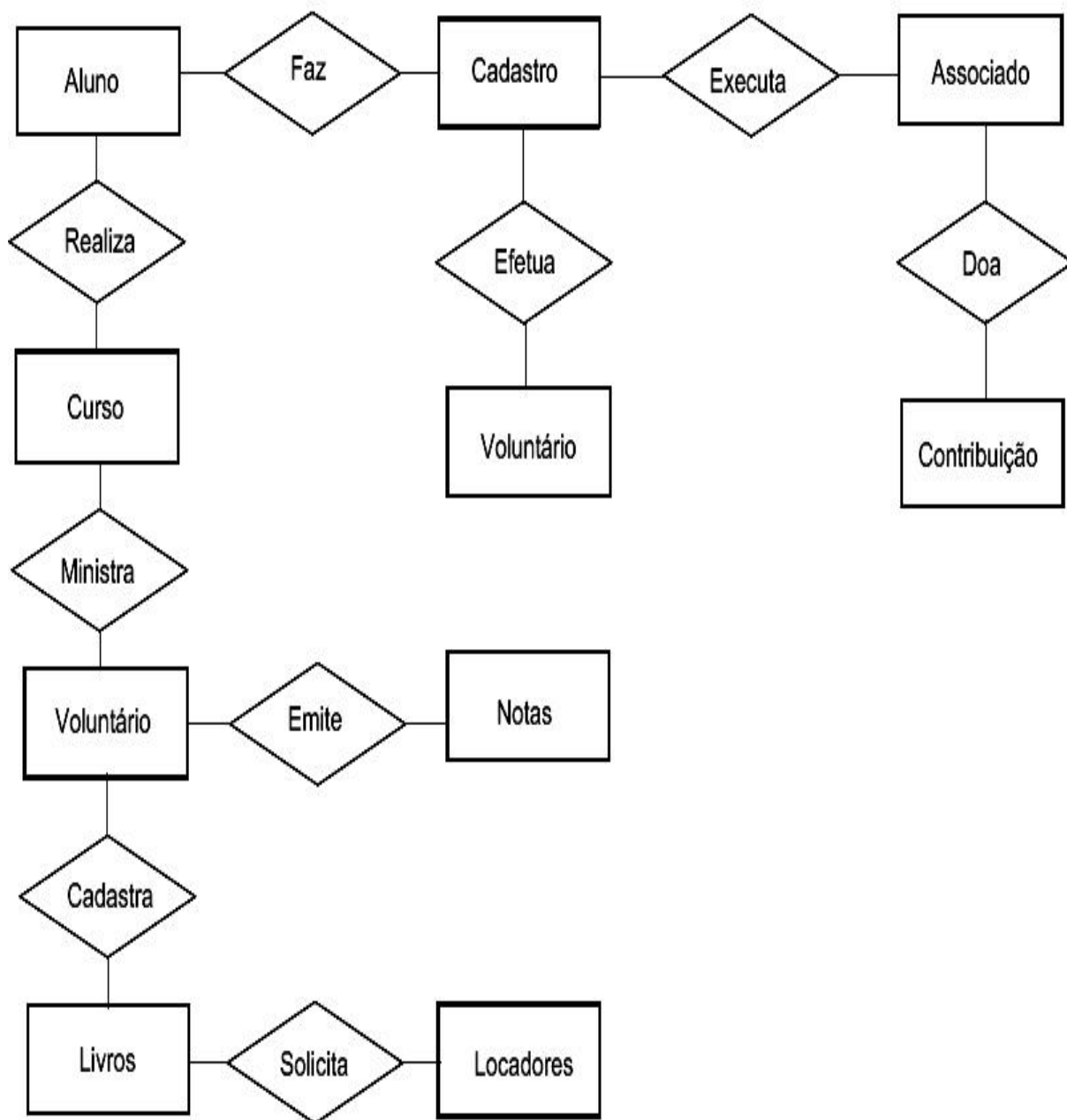


Figura 13 - DER Sistema Eclipse

17. Requisitos de Hardware e Software Sistema Eclipse

Realizamos um sistema que atende as necessidades da Fundação Flávio Fernandes de Matos, analisamos a configuração dos equipamentos de informática no qual seria executado o Sistema Eclipse , os quais possuem a seguinte configurações:

- Processador: Pentium 3, 434 Mhz
- Memoria Ram : 256 Mb
- HardDisk Capacity : 20Gb
- Placa de video : 32 Mb On-Board
- Placa de Rede: 10 Mb On-Board
- SO: Windows 98 / Windows XP
- Monitor: 15 polegadas

O Sistema Eclipse é leve e funcional sendo que pode se executado em sistemas que tenham as seguintes configurações de hardware e software:

- Processador: Pentium 3, 434 Mhz
- Memoria Ram : 128 Mb
- HardDisk Capacity : 20Gb
- SO: Windows 98 /2000/ Windows XP

18. Protocolos de comunicação

No domínio das redes de computação, um protocolo é um conjunto de especificações objetivas que os computadores entendem. Tecnicamente, é um conjunto de regras-padrão que caracterizam o formato, a sincronização, a seqüência e, ainda, a detecção de erros e falhas na comutação de pacotes, isto é, na transmissão de informação entre computadores.

Assim, dois ou mais computadores, para comunicarem numa rede, têm de falar a mesma linguagem, ou seja, usar o mesmo protocolo. Para existir comunicação é necessário existir pelo menos um canal, um emissor e um receptor e garantir que ambos tenham a faculdade de utilizar um protocolo comum.

18.1.Topologia Utilizada no Espaço Cultural Flávio Fernandes de Matos

A topologia utilizada no Espaço Cultural Flávio Fernandes de Matos é a topologia estrela devido aos benefícios que a mesma fornece. O servidor é interligado a toda a rede por cabos de par trançado conectando com os computadores do laboratório, recepção, administração e biblioteca.

18.1.1. Topologia Estrela

As redes em estrela são aquelas utilizam cabos de par trançado e um hub como ponto central da rede. O hub se encarrega de retransmitir todos os dados para todas as estações, mas com a vantagem de tornar mais fácil a localização dos problemas, já que se um dos cabos, uma das portas do hub ou uma das placas de rede estiver com problemas, apenas o PC ligado ao componente defeituoso ficará fora da rede,

ao contrário do que ocorre nas redes 10base2, onde se havendo um mal contato em qualquer um dos conectores e derrubada a rede inteira, prejudicando seus usuários. Claro que esta topologia se aplica apenas a pequenas redes, já que os hubs costumam ter apenas 8 ou 16 portas. Em redes maiores é utilizada a topologia de árvore, onde temos vários hubs interligados entre si por switches ou roteadores.

19. Testes

Paulo Tozelli (2008) esclarece que o teste do software é um processo realizado pelo testador de software que permeia outros processos da Engenharia de Software, e envolve ações que vão do levantamento de requisitos (necessidades) até a execução do teste propriamente dito.

O objetivo, por paradoxal que pareça, é encontrar defeitos nos produtos, para que estes possam ser corrigidos pela equipe de programadores, antes da entrega final. A maioria das pessoas pensa que o teste de software serve para demonstrar o correto funcionamento de um programa, quando na verdade ele é utilizado como um processo da engenharia de software para encontrar defeitos.

O processo de teste de software é voltado para o alcance de um nível de qualidade de produto, que durante o processo de desenvolvimento de software muda conforme avanço das atividades - requisitos, protótipos, modelo de dados lógico, modelo de dados físico, código-fonte, módulos funcionais e finalmente um sistema.

19.1. Teste de Unidade

Também conhecida como Teste Unitário. É a fase do processo de teste em que se testam as menores unidades de software desenvolvidas (pequenas partes ou unidades do sistema). O universo alvo desse tipo de teste são os métodos dos objetos ou mesmo pequenos trechos de código. Assim, o objetivo é o de encontrar falhas de funcionamento dentro de uma pequena parte do sistema funcionando independentemente do todo.

19.1.1. Teste de unidade Projeto eclipse

Aplicamos o teste unitário em procedures e funções criadas para facilitar o processo de desenvolvimento do sistema. Assim na hora da compilação podemos testar se cada um destes itens encontrava-se funcionais para que fosse dada a continuidade a aplicação no restante do sistema.

As procedures e funções criadas foram fundamentais para o Sistema Eclipse onde deram agilidade no processo criação de software

19.2. Teste de Sistema

Na fase de Teste de Sistema o objetivo é executar o sistema sob ponto de vista de seu usuário final, varrendo as funcionalidades em busca de falhas. Os testes são executados em condições similares - de ambiente, interfaces sistêmicas e massas de dados - àquelas que um usuário utilizará no seu dia-a-dia de manipulação do sistema. De acordo com a política de uma organização podem ser utilizadas condições reais de ambiente, interfaces sistêmicas e massas de dados.

19.2.1. Teste de Sistema Projeto Eclipse

Ao executar o teste de Sistema, escolhemos os analistas para representarem o papel dos clientes devido aos mesmos estarem mais aptos ao funcionamento da Fundação Flávio Fernandes de Matos. No decorrer do teste foram achados erros no qual os programadores deixaram passar despercebidos. Assim como a opção de restauração do backup, pois no ato que o usuário solicitava o cancelamento o programa não aceitava assim dando continuidade à restauração do arquivo.

19.3. Teste de Operação

Fase de Teste em que o teste é conduzido pelos administradores do ambiente final onde o sistema ou software entrará em ambiente produtivo. Vale ressaltar que essa fase é aplicável somente a sistemas de informação próprios de uma organização, cujo acesso pode ser feito interna e/ou externamente a essa organização. Nessa fase de teste devem ser feitas simulações para garantir que a entrada em produção do sistema será bem sucedida. Envolve testes de instalação, simulações com backup e restore das bases de dados, etc. Em alguns casos um sistema entrará em produção para substituir outro e é necessário garantir que o novo sistema continuará garantindo o suporte ao negócio.

19.3.1. Teste de Operação Sistema Eclipse

No teste de operação o sistema eclipse apresentou ao cliente os seguintes erros:

- O servidor estava indisponível, porém, foi solucionado com a reinicialização do mesmo.
- Não apresenta a foto tirada com a WebCam e quando era apresentada a foto trazia cortes.
- Não atualiza os dados nas consultas

Sendo que no próprio local os programadores com o auxílio de um notebook corrigiram os erros apresentados no teste. Assim no processo de teste de operação do Software coube aos analistas e programadores usarem os seguintes procedimentos:

- Levantar erros trazendo as soluções;
- Adequar ao cliente o sistema propostos fornecendo explicações minuciosas.

Através deste teste podemos garantir a funcionalidade do Sistema Eclipse.

20. Instalador

Utilizamos O Mep Installer 2.1 é um ótimo gerador de programas de instalação além de ser gratuito para programas voltados a plataforma Windows, ele é baseado no Inno Setup. Ele é fácil de usar, e gera instalações profissionais, com um novo visual diferenciado. Os instaladores gerados por ele se encontram em diversas linguagens inclusive Português (Brasil).

20.1. Código Fonte do Instalador Eclipse

A seguir o código fonte gerado pelo Mep Installer 2.1 :

; Projeto gerado pelo Assistente de Projeto do Mep Installer.

; VEJA A DOCUMENTAÇÃO PARA OBTER DETALHES SOBRE A CRIAÇÃO
DE PROJETOS DO MEP INSTALLER!

[Setup]

AppName=Eclipse

AppVerName=Eclipse 1.7

AppId=Eclipse ID

AppPublisher=F11

DefaultDirName={pf}\Eclipse System

DefaultGroupName=Eclipse

OutputDir=C:\Users\william\Desktop

OutputBaseFilename=setup_eclipse_1-7

SetupIMG=C:\Program Files\Mep Installer 2\setup.img

Compression=lzma/max

SolidCompression=yes

PrivilegesRequired=none

[Languages]

Name: "english"; MessagesFile: "compiler:Default.isl"

Name: "portuguesebr"; MessagesFile: "compiler:Languages\PortugueseBR.isl"

[Tasks]

Name: "desktopicon"; Description: "{cm:CreateDesktopIcon}"; GroupDescription: "{cm:AdditionalIcons}"; Flags: unchecked

Name: "quicklaunchicon"; Description: "{cm:CreateQuickLaunchIcon}"; GroupDescription: "{cm:AdditionalIcons}"; Flags: unchecked

[Files]

Source: "C:\Users\william\Desktop\intalador\programa\eclipse.exe"; DestDir: "{app}"; Flags: ignoreversion

Source: "C:\Users\william\Desktop\intalador\programa\BD*"; DestDir: "{app}\BD";
Flags: ignoreversion recursesubdirs createallsubdirs

Source: "C:\Users\william\Desktop\intalador\programa\help*"; DestDir: "{app}\help"; Flags: ignoreversion recursesubdirs createallsubdirs

Source: "C:\Users\william\Desktop\intalador\programa\Imagens*"; DestDir: "{app}\imagenes"; Flags: ignoreversion recursesubdirs createallsubdirs

[Icons]

Name: "{group}\Eclipse"; WorkingDir: "{app}"; Filename: "{app}\eclipse.exe"

Name: "{group}\{cm:UninstallProgram,Eclipse}"; WorkingDir: "{app}"; Filename: "{uninstallexe}"

Name: "{commondesktop}\Eclipse"; WorkingDir: "{app}"; Filename: "{app}\eclipse.exe"; Tasks: desktopicon

Name: "{userappdata}\Microsoft\Internet Explorer\Quick Launch\Eclipse"; WorkingDir: "{app}"; Filename: "{app}\eclipse.exe"; Tasks: quicklaunchicon

[Run]

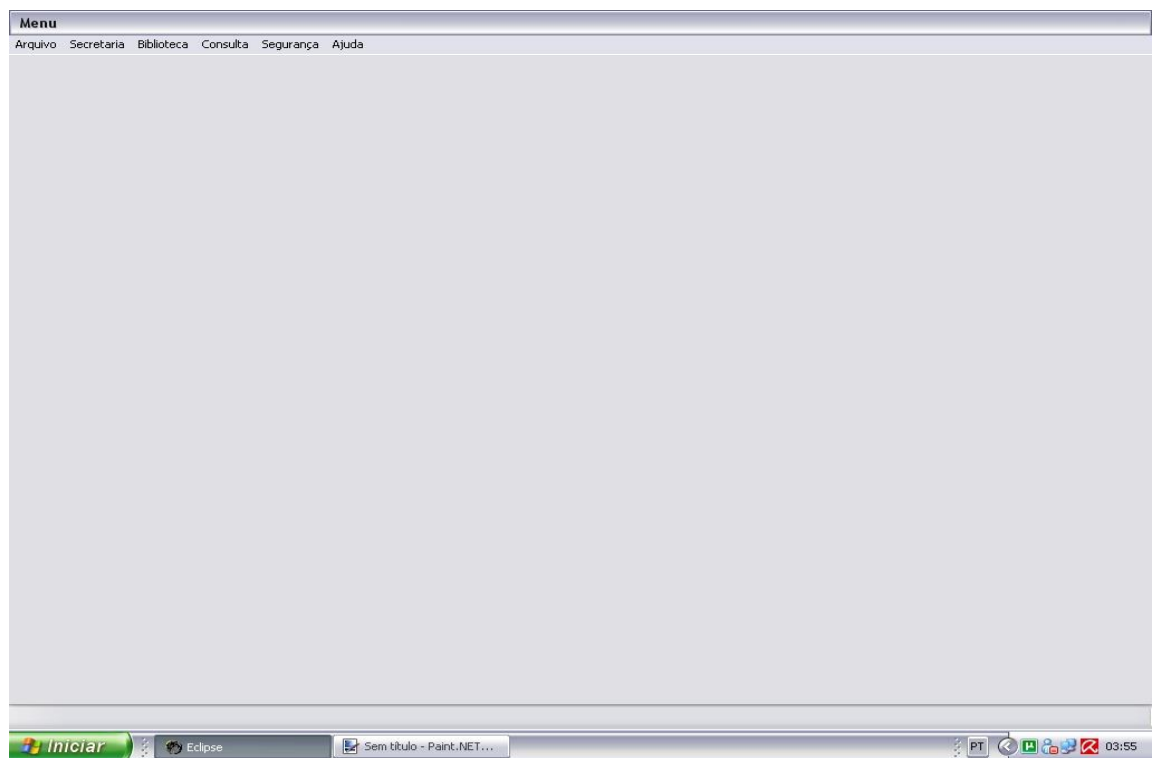
Filename: "{app}\eclipse.exe"; Description: "{cm:LaunchProgram,Eclipse}"; Flags: nowait postinstall skipifsilent

21. Layouts

21.1. Tela de Logon



21.2. Menu Tradicional



21.3.Menu Compacto



21.4.Cadastro de Curso

A screenshot of a software window titled "Cadastro de Curso". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. Below the title bar is a toolbar containing several icons: a green plus sign, a pencil, a trash can, a grey X, a grey checkmark, a magnifying glass, and a red arrow pointing to a document. The main content area of the window contains two text input fields. The first field is labeled "Código do curso:" and contains the text "1". The second field is labeled "Nome do curso" and contains the text "teste".

21.5.Cadastro de Turmas

Cadastro de Turma

Código: 2 Curso: teste Professor: joao

Nome da turma: teste

Data de início: 3/3/2009 Data final prevista: 9/9/2009 Data final: 9/9/2009

Número de vagas: 30 Número de alunos: 12

Obs: 123

Status:
☐ Turma Aberta
☐ Turma Fechada

21.6.Cadastro de Associados

Cadastro de Associado

Código do Associado: 1

Data de Admissão: 1/1/2009 Sexo: ☐ Fem ☒ Masc

Nome: Miguel Data de nascimento: 3/3/2009

RG: 3030 CPF: 383008118 Nacionalidade: 0303 Naturalidade: 30

Endereço: 103 Número: 30 Complemento: 30 Bairro: 30


CEP: 30 Cidade: 3030 UF: SP

Email: 03

Escolaridade: 30 Formação: 30 Aptidão: wq

Como conheceu a instituição: qwd Profissão: qdw Indicado por:





Cadastro de contato

Associado 

Tipo Número

	Cod.	Tipo	Número
*			

Contato

Consulta

Nome

	codigo_ass	Associado
▶		

Sair

21.7.Cadastro de Voluntário

The screenshot shows the 'Cadastro de Voluntários' window with the 'Dados Pessoais' tab selected. The window has a title bar with standard OS controls and a toolbar with icons for adding, editing, deleting, and saving records, as well as buttons for 'Cargos' and 'Contatos'. The form fields are as follows:

Dados Pessoais		
Código	1	
Nome	joao	
Sexo	<input type="radio"/> Mas <input type="radio"/> Fem	
Data de Nascimento	Nacionalidade	Naturalidade
14/ 6 /2009	345	3
RG	CPF	
0303313	24544345	

The screenshot shows the 'Cadastro de Voluntários' window with the 'Endereço' tab selected. The window has the same title bar and toolbar as the previous screenshot. The form fields are as follows:

Endereço			
Endereço	Número	Complemento	
5	4	5	
Bairro	CEP	Cidade	UF
3	4	5	

Cadastro de Voluntários







 Cargos Contatos

Dados Pessoais Endereço **Informações Gerais**

Escolaridade
 Formação

Profissão
 Aptidão


Indicação
 Deseja trabalhar na associação ☐ Sim ☐ Não

Dias da semana
 Horários disponíveis

Área
 Data de Admissão

E-mail





Cadastro de Cargos

Voluntario: 

Cargo:

	Código	Voluntario	Cargo
*			

Contato

Consulta

Nome

codigo_vol	nome_vol
▶	

Cadastro de contato

Voluntario

Tipo Número

Cod.	Tipo	Número
*		

Contato

21.8.Cadastro Alunos

Cadastro de Aluno

+ ✎ 🗑️ ✕ ✓ 🔍 Interesses Contatos ➡

Dados do aluno | Dados paternos | Informações gerais

Código do Aluno: 1

Nome do aluno: Sexo: ☐ Fem ☒ Masc

Data de nascimento: RG: Nacionalidade: Naturalidade:

Endereço: Número: Complemento:

Bairro: CEP: Cidade: UF:

Email: Status do aluno: ☐ Cursando ☐ Finalizado ☐ Lista de Espera

Foto do Aluno:

Cadastro de Aluno

+ ✎ 🗑️ ✕ ✓ 🔍 Interesses Contatos ➡

Dados do aluno | Dados paternos | Informações gerais

Nome da mãe: Profissão da mãe: RG da mãe: CPF da mãe:

Nome do Pai: Profissão do pai: RG do pai: CPF do pai:

Cadastro de Aluno

Interesses Contatos

Dados do aluno Dados paternos **Informações gerais**

Estudante? ☒ Sim ☐ Não Série 3 Nome da escola onde estuda CTZL Tipo da escola a qual estuda Técnica

Endereço da escola ase CEP da escola 03013233 Bairro da escola asdas

Cidade da escola asd UF da escola SP

Como conheceu o curso? asd Renda familiar 4000 Pessoas na residência 5 Valor Aluguel Casa Própria? ☐ Sim ☐ Não

Código Voluntário Data de cadastro 8/ 6 /2009

Cadastro de interesse

Aluno:

Curso Data

	Cod. Alu.	Nome	Cod. Cur.	Curso	Data Inscrição
*					

Contato

+ -

Cadastro de contato

Aluno

Tipo Número

Cod.	Tipo	Número
*		

Contato

+

-

+

-

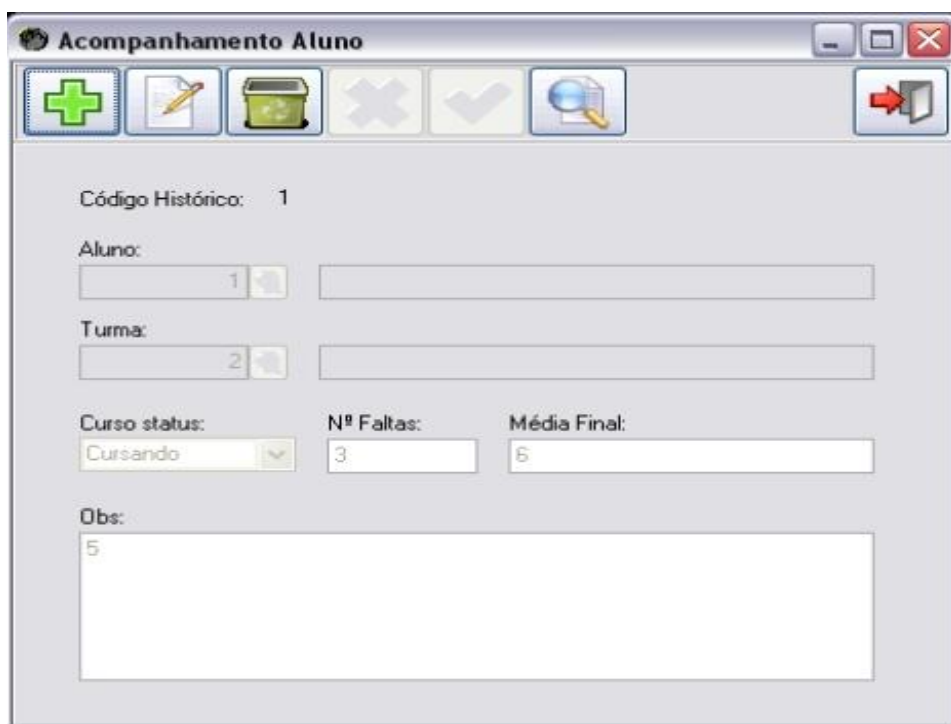
Consulta

Nome

codigo_alu	nome_alu
1	diego
3	matheus
19	carol
29	teste

Sair

21.9.Consulta Histórico do Aluno



Acompanhamento Aluno

Código Histórico: 1

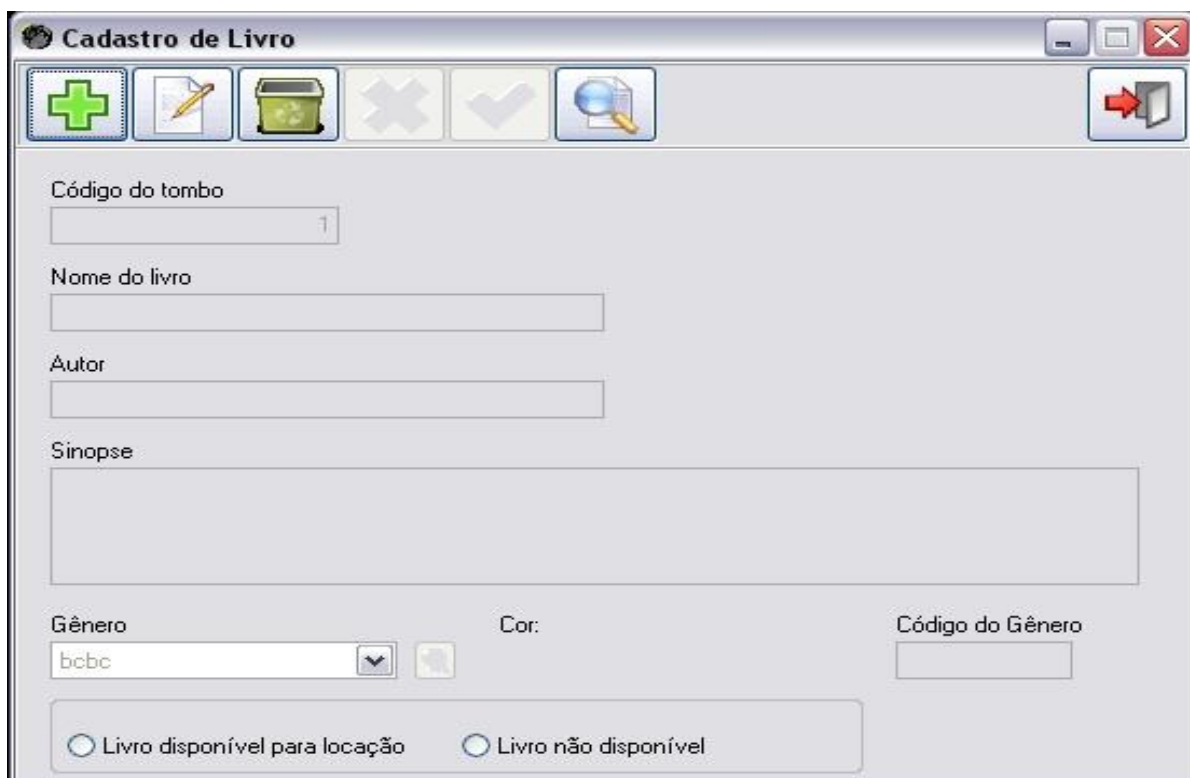
Aluno:

Turma:

Curso status: Nº Faltas: Média Final:

Obs:

21.10. Cadastro Livros



Cadastro de Livro

Código do tomo:

Nome do livro:

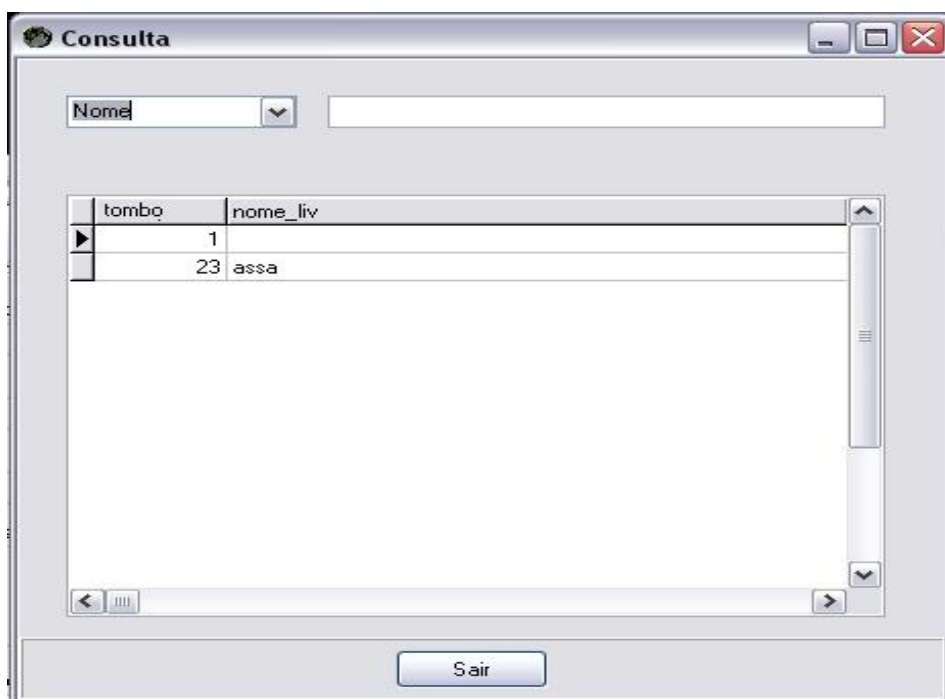
Autor:

Sinopse:

Gênero: Cor: Código do Gênero:

☐ Livro disponível para locação ☐ Livro não disponível

21.10.1. Consulta Livro



The 'Consulta' window displays a table with two columns: 'tombo' and 'nome_liv'. The table contains two rows of data. Below the table is a 'Sair' button.

tombo	nome_liv
1	
23	assa

Sair

21.10.2. Cadastro de Gênero



The 'Cadastro de Gênero' window features a toolbar with icons for adding, editing, deleting, and saving. The form includes fields for 'Código do Gênero', 'Nome do Gênero', and 'Cor'.

Código do Gênero: 1

Nome do Gênero

bcbe

Cor

21.11. Cadastro de locador

Cadastro de Locador

Toolbar: Add, Edit, Delete, Undo, Redo, Search, Print

Código do locador: 1

Nome do locador: asdas

Sexo: ☐ Fem

RG: CPF:

Endereço: Cidade:

Email:

Bloqueado: ☐ Sim ☐ NÃO

21.12. Cadastro de Locação

Locação de livros

Toolbar: Add, Edit, Delete, Undo, Redo, Search, Print

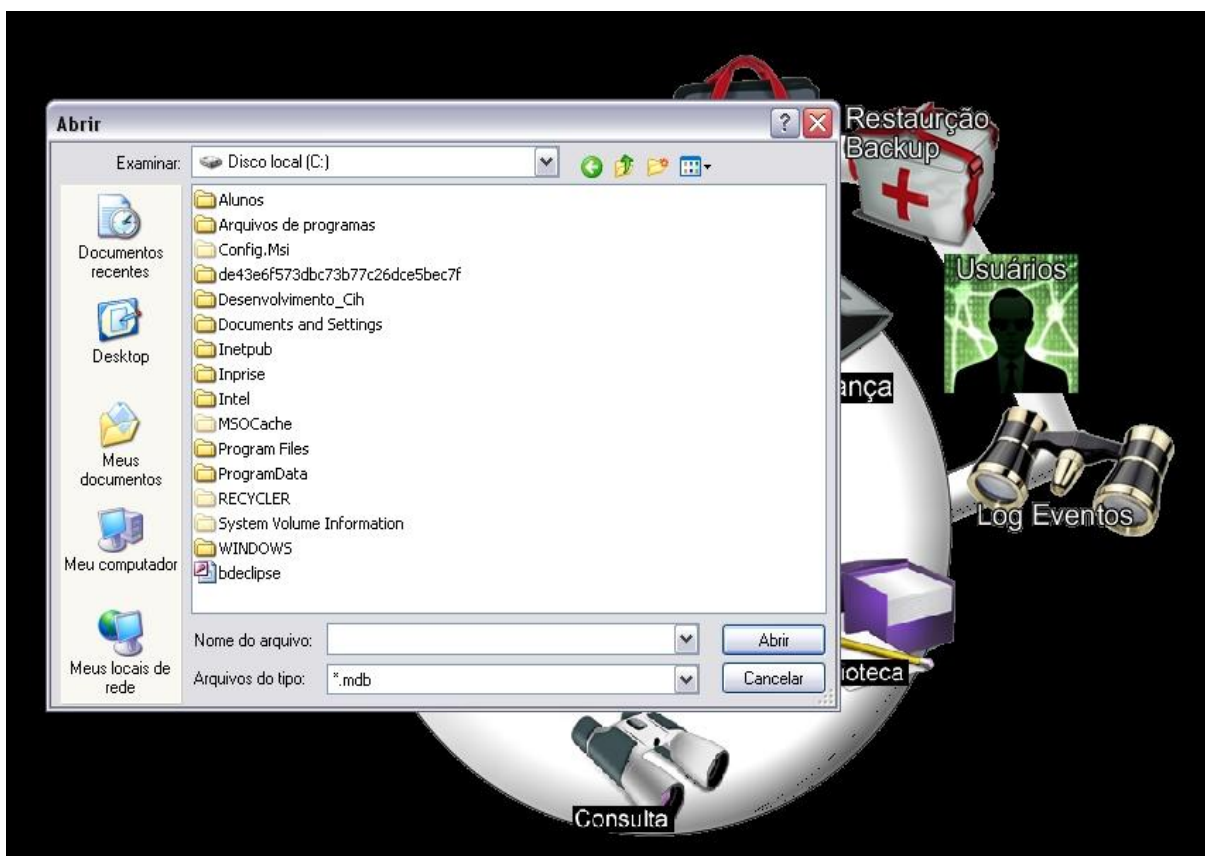
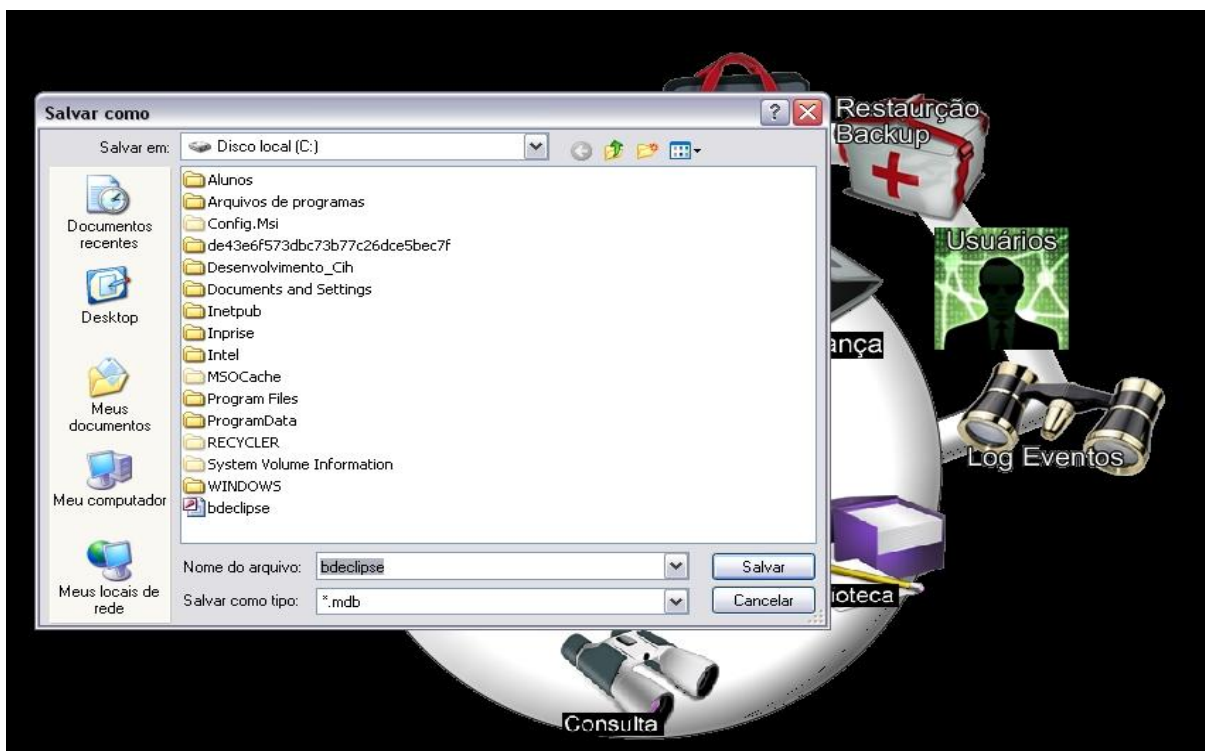
Código locação: 1 Tombo do livro: Locador:

Nome do Livro: Nome do Locador: asdas

Data de locação: Data prevista: Data de entrega: Pendente?

Observação da locação:

21.13. Backup e Restauração



21.14. Cadastro de Usuário

21.15. Gerenciador de Log de Eventos

21.16. Sobre



22. Codificação

```
unit API_F11;

interface

uses

  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DBCtrls, StdCtrls, ExtCtrls, Mask, ComCtrls, DB, ToolWin,
  Buttons, Grids, DBGrids, ADODB, Menus;

type

  TUser_id = class

    nome : String;

    nível : integer;

    senha : string;

    frase:string;

  end;

type

  TParam = record

    Param : Array of String;

  end;

var

  p:textfile;

  menup:string;

  User_id : TUser_id;
```

```

Params : TParam;

Getmaxcodi : TADOQuery;

function data(vdata:string):boolean;

function cpf(vcpf:string):boolean;

function Crypt(Action, Src: String): String;

procedure trims(form:tform);

procedure cor(grade:tdbgrid;color:tcolor);

procedure mudaedit(form : TForm; resu : boolean);

procedure botoes(Form : TForm);

procedure abrirpesq(Form : TForm);

procedure pesq (Form : TForm ; tabelapesq : TADOQuery ;nometb, camponome,
campocodi : String);

function atribfield(Cfield :TAutoIncField):integer;

procedure niveldeacesso(n:integer);

procedure pesq_val(Form : TForm; tabelapesq : TADOQuery; msg :String; edtcam-
popesq :TDBEdit; edtcamponome :TEdit; camponome : TWideStringField);

// procedure param_user(Form : TForm; result : boolean; params : TParam);

// function Getmax_codi(Form : TForm; tabela, campo : String; dataSource : TData-
Source):integer;

implementation

uses UpesqGeral, UDMeclipse, ComObj, UPrincipal;

procedure trims(form:tform);

```

```

var x:integer;

begin
  with form do
    begin
      for x:=0 to ComponentCount - 1 do
        begin
          if Components[x].ClassName = 'TDBEdit' then
            begin
              (Components[x] as TDBEdit).text :=trim((Components[x] as TDBEdit).text);
            end;
          end;
        end;
      end;
    end;
  end;

function cpf(vcpf:string):boolean;

var a :array[1..13] of byte;

    i:integer;

begin
  if (Length(vcpf)<11) or (Length(vcpf)>11) then
    begin
      result:=false;
      exit;
    end;
  end;

```

```
for i:=1 to 11 do
```

```
begin
```

```
    a[i]:=strtoint(Copy(vcpf,i,1));
```

```
end;
```

```
a[12]:=((a[1]*1)+(a[2]*2)+(a[3]*3)+(a[4]*4)+(a[5]*5)+(a[6]*6)+(a[7]*7)+(a[8]*8)+(a[9]*9)  
) mod 11;
```

```
a[13]:=((a[1]*0)+(a[2]*1)+(a[3]*2)+(a[4]*3)+(a[5]*4)+(a[6]*5)+(a[7]*6)+(a[8]*7)+(a[9]*8)  
+(a[10]*9)) mod 11;
```

```
if a[12]>=10 then
```

```
    a[12]:=0;
```

```
if a[13]>=10 then
```

```
    a[13]:=0;
```

```
if (a[12]=a[10])and(a[13]=a[11]) then
```

```
    result:=true
```

```
else
```

```
    result:=false;
```

```
end;
```

```
function Crypt(Action, Src: String): String;
```

```
Label Fim;
```

```
var KeyLen : Integer;
```

```
    KeyPos : Integer;
```

```
    OffSet : Integer;
```



```

Dest, Key : String;

SrcPos : Integer;

SrcAsc : Integer;

TmpSrcAsc : Integer;

Range : Integer;

begin

  if (Src = "") Then

    begin

      Result:= "";

      Goto Fim;

    end;

    Key
:= 'YUQL23KL23DF90WI5E1JAS467NMCXXL6JAOAUWWMCL0AOMM4A4VZYW9K
HJUI2347EJHJKDF3424SKL K3LAKDJSL9RTIKJ';

    Dest := "";

    KeyLen := Length(Key);

    KeyPos := 0;

    SrcPos := 0;

    SrcAsc := 0;

    Range := 256;

    if (Action = UpperCase('C')) then

      begin

        Randomize;

```

```

OffSet := Random(Range);

Dest := Format('%1.2x',[Offset]);

for SrcPos := 1 to Length(Src) do

begin

    Application.ProcessMessages;

    SrcAsc := (Ord(Src[SrcPos]) + OffSet) Mod 255;

    if KeyPos < KeyLen then KeyPos := KeyPos + 1 else KeyPos := 1;

    SrcAsc := SrcAsc Xor Ord(Key[KeyPos]);

    Dest := Dest + Format('%1.2x',[SrcAsc]);

    OffSet := SrcAsc;

end;

end

Else if (Action = UpperCase('D')) then

begin

    OffSet := StrToInt('$'+ copy(Src,1,2));

    SrcPos := 3;

repeat

    SrcAsc := StrToInt('$'+ copy(Src,SrcPos,2));

    if (KeyPos < KeyLen) Then KeyPos := KeyPos + 1 else KeyPos := 1;

    TmpSrcAsc := SrcAsc Xor Ord(Key[KeyPos]);

    if TmpSrcAsc <= OffSet then TmpSrcAsc := 255 + TmpSrcAsc - OffSet

    else TmpSrcAsc := TmpSrcAsc - OffSet;

```

```

    Dest := Dest + Chr(TmpSrcAsc);

    OffSet := SrcAsc;

    SrcPos := SrcPos + 2;

until (SrcPos >= Length(Src));

end;

Result:= Dest;

Fim:

end;

procedure niveldeacesso(n:integer);

begin {

    if n=1 then

        begin

            FrmPrincipal.Cadastro1.Enabled:=true;

            FrmPrincipal.Cadastro1.Visible:=true;

            FrmPrincipal.Biblioteca1.Enabled:=false;

            FrmPrincipal.Biblioteca1.Visible:=false;

            frmprincipal.Segurana1.Enabled:=false;

            FrmPrincipal.Segurana1.Visible:=false;

        end;

        if n=2 then

            begin

```

```

    FrmPrincipal.Cadastro1.Enabled:=false;

    FrmPrincipal.Cadastro1.Visible:=false;

    FrmPrincipal.Biblioteca1.Enabled:=true;

    FrmPrincipal.Biblioteca1.Visible:=true;

    frmprincipal.Segurana1.Enabled:=false;

    FrmPrincipal.Segurana1.Visible:=false;

end;

if n=3 then

begin

    FrmPrincipal.Cadastro1.Enabled:=true;

    FrmPrincipal.Cadastro1.Visible:=true;

    FrmPrincipal.Biblioteca1.Enabled:=true;

    FrmPrincipal.Biblioteca1.Visible:=true;

    frmprincipal.Segurana1.Enabled:=true;

    FrmPrincipal.Segurana1.Visible:=true;

end; }

end;

{function Getmax_codi(Form : TForm; tabela, campo : String; dataSource : TData-
Source):integer;

begin

    with Form do

        begin

            with Getmaxcodi do

```

```

begin

    Getmaxcodi := TADOQuery.Create(Form);

    DataSource := dataSource;

    Close;

    SQL.Add('SELECT SUM('+ campo +' ) as MAX FROM TB_' + tabela);

    Open;

end;

end;

Result := 2;

end; }

//procedure param_user(Form : TForm; result : boolean; params : TParam);

//var x, y : integer ;

//begin

{ with form do

begin

    for x:=0 to ComponentCount - 1 do

begin

    if (Components[x].ClassName = 'TMenuItem') then

begin

    for y:=0 to TMenuItem.

        if((Components[x] as TMenuItem).Name = params.Param[] )then

```

```

        (Components[x] as TMenuItem).Enabled := resu;

    end;

end;

    }

//end;

procedure pesq_val(Form : TForm; tabelapesq : TADOQuery; msg :String; edtcampopesq :TDBEdit; edtcamponome :TEdit; camponome : TWideStringField);

begin

    with form do

    begin

        try

            if(trim(edtcampopesq.Text) <> '')then

                begin

                    tabelapesq.Close;

                    tabelapesq.Parameters.ParamByName('CODI').Value := edtcampopesq.Text;

                    tabelapesq.Open;

                    if not(tabelapesq.IsEmpty)then

                        edtcamponome.Text := camponome.AsString

                    else

                        begin

                            ShowMessage(msg);

                            edtcampopesq.SetFocus;

                            edtcampopesq.Clear;

```

```

        edtcamponome.Clear;

    end;

end;

except

    on EOleException do

        ShowMessage('Valor inválido');

    end;

end;

end;

end;

function atribfield(Cfield :TAutoIncField):integer;

begin

    with DMeclipse do

        begin

            with Cfield do

                begin

                    Result := AsInteger;

                end;

            end;

        end;

    end;

end;

```

```

procedure pesq (Form : TForm ; tabelapesq : TADOQuery ;nometb, camponome,
campocodi : String);

begin

```

```

with form do

begin

    with DMeclipse do

begin

    with tabelapesq do

begin

        Close;

        if(Frmpesq.ComboBox1.ItemIndex = 1)then

            begin

                SQL.Text := 'SELECT * FROM ' + 'TB_'+ nometb + ' WHERE ' + campocodi +
' = ' + ':CODI';

                Parameters.ParamByName('CODI').Value := Frmpesq.Edit1.Text;

            end

        else

            begin

                if(Frmpesq.ComboBox1.ItemIndex = 0)then

                    begin

                        SQL.Text := 'SELECT * FROM ' + ' TB_'+ nometb + ' WHERE ' + campo-
nome + ' like ' + ':NOME';

                        Parameters.ParamByName('NOME').Value := Frmpesq.Edit1.Text;

                    end;

                end;

            end;

        Open;

```



```

        end;

    end;

end;

end;

procedure abrirpesq(Form : TForm);

begin

    with Form do

        begin

            If(Frmpesq = nil)then

                begin

                    Application.CreateForm(TFrmpesq, Frmpesq);

                    Frmpesq.ShowModal;

                end;

            end;

        end;

    end;

    //essa procedure so pode ser usada caso vc queira modificar o enable do button 1
    ao 5 e 7

    procedure botoes(Form : TForm);

    var x :integer;

    begin

        with Form do

            begin

                for x:=0 to ComponentCount - 1 do

```

```

begin

  if Components[x].ClassName = 'TBitBtn' then

    begin

      if((Components[x] as TBitBtn).Name = 'BitBtn1') or ((Components[x] as
TBitBtn).Name = 'BitBtn2') or ((Components[x] as TBitBtn).Name = 'BitBtn3') or
((Components[x] as TBitBtn).Name = 'BitBtn4') or ((Components[x] as TBitBtn).Name
= 'BitBtn5') or ((Components[x] as TBitBtn).Name = 'BitBtn7'))then

        (Components[x] as TBitBtn).Enabled := not (Components[x] as
TBitBtn).Enabled;

      end;

      if Components[x].ClassName = 'TSpeedButton' then

        begin

          if((Components[x] as TSpeedButton).Name = 'SpeedButton1') or ((Compo-
nents[x] as TSpeedButton).Name = 'SpeedButton2'))then

            (Components[x] as TSpeedButton).Enabled := not (Components[x] as
TSpeedButton).Enabled;

          end;

        end;

      end;

    end;

  function data(vdata:string):boolean;

  begin

    try

      StrToDate(vdata);

```

```

    data:=true;

except

    MessageDlg('Data Inválida !!' , mtInformation, [mbOk], 0);

    data:=false;

end;

end;

procedure cor(grade:tdbgrid;color:tcolor);

var

    i:integer;

    numcampos:integer;

begin

    numcampos:=grade.FieldCount;

    for I := 0 to numcampos-1 do

        grade.columns[i].font.color:=color;

    end;

procedure mudaedit(form : TForm ;resu : boolean);

var x :integer;

begin

    with form do

        begin

            for x:=0 to ComponentCount - 1 do

                begin

```

```

if (Components[x].ClassName = 'TDBEdit') then
begin
  (Components[x] as TDBEdit).Enabled := resu;

  if(resu)then
  begin
    (Components[x] as TDBEdit).Color := clWhite;

    (Components[x] as TDBEdit).ReadOnly := false;
  end
else
begin
  (Components[x] as TDBEdit).Color := clBtnFace;

  (Components[x] as TDBEdit).ReadOnly := true;
end;
end;

if(Components[x].ClassName = 'TDBComboBox')then
begin
  (Components[x] as TDBComboBox).Enabled := resu;

  if(resu)then
  begin
    (Components[x] as TDBComboBox).Color := clWhite;

    (Components[x] as TDBComboBox).ReadOnly := false;
  end
end

```

```

else

begin

    (Components[x] as TDBComboBox).Color := clBtnFace;

    (Components[x] as TDBComboBox).ReadOnly := true;

end;

end;

if(Components[x].ClassName = 'TDBRadioGroup')then

begin

    (Components[x] as TDBRadioGroup).Enabled := resu;

    if(resu)then

begin

    (Components[x] as TDBRadioGroup).Color := clWhite;

    (Components[x] as TDBRadioGroup).ReadOnly := false;

end

else

begin

    (Components[x] as TDBRadioGroup).Color := clBtnFace;

    (Components[x] as TDBRadioGroup).ReadOnly := true;

end;

end;

if(Components[x].ClassName = 'TDBImage')then

begin

```

```

(Components[x] as TDBImage).Enabled := resu;

if(resu)then

begin

    (Components[x] as TDBImage).Color := clWhite;

    (Components[x] as TDBImage).ReadOnly := false;

end

else

begin

    (Components[x] as TDBImage).Color := clBtnFace;

    (Components[x] as TDBImage).ReadOnly := true;

end;

end;

if (Components[x].classname = 'TDateTimePicker') then

    (Components[x] as TDateTimePicker).Enabled:=not((Components[x] as TDate-
TimePicker).Enabled);

if(Components[x].ClassName = 'TDBGrid')then

begin

    (Components[x] as TDBGrid).Enabled := resu;

    if(resu)then

begin

    (Components[x] as TDBGrid).Color := clWhite;

    (Components[x] as TDBGrid).ReadOnly := false;

end

end

```

```

    else

    begin

        (Components[x] as TDBGrid).Color := clBtnFace;

        (Components[x] as TDBGrid).ReadOnly := true;

    end;

end;

end;

end;

end;

end.

unit UPrincipal;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

    Dialogs, StdCtrls, Menus, XPMAN, ComCtrls, Buttons, ToolWin, ExtCtrls,

    ShellCtrls, shellapi, pngimage, JvGIF;

type

    TFrmPrincipal = class(TForm)

        SaveDialog1: TSaveDialog;

        OpenDialog1: TOpenDialog;

        Image1: TImage;

        isecretaria: TImage;

```

iseguranca: TImage;
ibiblioteca: TImage;
iconsulta: TImage;
iarquivo: TImage;
ivoluntario: TImage;
iassociado: TImage;
ialuno: TImage;
icurso: TImage;
iturma: TImage;
ibackup: TImage;
irestauracao: TImage;
ilogeventos: TImage;
iusuarios: TImage;
igenero: TImage;
ilocador: TImage;
ilocacao: TImage;
ilivros: TImage;
iconsultageral: TImage;
iconsultaturma: TImage;
ilogoff: TImage;
isair: TImage;
icontrolbiblioteca: TImage;


```
icontrolarquivo: TImage;

icontrolsecretaria: TImage;

icontrolseguranca: TImage;

icontrolconsulta: TImage;

ihelp: TImage;

isobre: TImage;

iajuda: TImage;

icontrolhelp: TImage;

iview: TImage;

procedure Sair1Click(Sender: TObject);

procedure Aluno1Click(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure Associado1Click(Sender: TObject);

procedure Cargo1Click(Sender: TObject);

procedure Curso1Click(Sender: TObject);

procedure Gnero1Click(Sender: TObject);

procedure Livro1Click(Sender: TObject);

procedure Locador1Click(Sender: TObject);

procedure Turma1Click(Sender: TObject);

procedure Geral1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Voluntario1Click(Sender: TObject);
```

```

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Sair2Click(Sender: TObject);

procedure HistoricoAluno1Click(Sender: TObject);

procedure Locao1Click(Sender: TObject);

procedure Usurio1Click(Sender: TObject);

procedure Backup1Click(Sender: TObject);

procedure RestauraodeBackup1Click(Sender: TObject);

procedure isecretariaClick(Sender: TObject);

procedure iarquivoClick(Sender: TObject);

procedure iconsultaClick(Sender: TObject);

procedure ibibliotecaClick(Sender: TObject);

procedure isegurancaClick(Sender: TObject);

procedure tamanho;

procedure ihelpClick(Sender: TObject);

procedure iaajudaClick(Sender: TObject);

procedure iviewClick(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

```

```

FrmPrincipal: TFrmPrincipal;

i:timage;

centrov,centroh:integer;

r:boolean;

implementation

uses API_F11, Math, UCadAluno, UCadAssociado, UCadCargo, UCadCurso,
    UCadGenero, UCadLivro, UCadLocador, UCadTurma, UpesqGeral, Ucadvol,
    UcadHistorico, ULocLivro, Ucadusu, Registry, ULogin, Uconf, UPrincipal2;

{$R *.dfm}

procedure TFrmPrincipal.FormClose(Sender: TObject;
    var Action: TCloseAction);

begin
    action:= cafree;

    FrmPrincipal:= nil;

end;

procedure TFrmPrincipal.Sair2Click(Sender: TObject);

begin
    Application.Terminate;

end;

procedure TFrmPrincipal.Sair1Click(Sender: TObject);

begin
    Close;

```

```

    FrmLogin.Show;

end;

procedure TFrmPrincipal.Aluno1Click(Sender: TObject);

begin

    If(FrmAluno = nil)then

        begin

            Application.CreateForm(TFrmAluno, FrmAluno);

            FrmAluno.Show;

            close;

        end;

    end;

end;

procedure TFrmPrincipal.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

end;

procedure TFrmPrincipal.Associado1Click(Sender: TObject);

begin

    If(FrmAssociado = nil)then

```

```

begin

    Application.CreateForm(TFrmAssociado, FrmAssociado);

    FrmAssociado.Show;

    close;

end;

end;

procedure TFrmPrincipal.Cargo1Click(Sender: TObject);

begin

    If(FrmCargo = nil)then

    begin

        Application.CreateForm(TFrmCargo, FrmCargo);

        FrmCargo.Show;

        close;

    end;

end;

procedure TFrmPrincipal.Cursor1Click(Sender: TObject);

begin

    If(FrmCurso = nil)then

    begin

        Application.CreateForm(TFrmCurso, FrmCurso);

        FrmCurso.Show;

        close;

```

```

end;

end;

procedure TFrmPrincipal.Gnero1Click(Sender: TObject);

begin

    If(FrmGenero = nil)then

        begin

            Application.CreateForm(TFrmGenero, FrmGenero);

            FrmGenero.Show;

            close;

        end;

    end;

end;

procedure TFrmPrincipal.Livro1Click(Sender: TObject);

begin

    If(FrmLivro = nil)then

        begin

            Application.CreateForm(TFrmLivro, FrmLivro);

            FrmLivro.Show;

            close;

        end;

    end;

end;

procedure TFrmPrincipal.Locador1Click(Sender: TObject);

begin

```

```

If(FrmLocador = nil)then

begin

    Application.CreateForm(TFrmLocador, FrmLocador);

    FrmLocador.Show;

    close;

end;

end;

procedure TFrmPrincipal.Turma1Click(Sender: TObject);

begin

    If(FrmTurma = nil)then

    begin

        Application.CreateForm(TFrmTurma, FrmTurma);

        FrmTurma.Show;

        close;

    end;

end;

procedure TFrmPrincipal.Geral1Click(Sender: TObject);

begin

    If(Frmpesq = nil)then

    begin

        Application.CreateForm(TFrmpesq, Frmpesq);

        Frmpesq.show;

```

```

        close;

    end;

end;

procedure TFrmPrincipal.FormCreate(Sender: TObject);

var x:integer;

begin

    String_tb := TString_tb.Create;

    r:=true;

    Image1.Picture.LoadFromFile('imagens\menu\roda.png');

    icontrolhelp.Picture.LoadFromFile('imagens\menu\controlhelp.png');

    icontrolarquivo.Picture.LoadFromFile('imagens\menu\controlarquivo.png');

    icontrolsecretaria.Picture.LoadFromFile('imagens\menu\controlsecretaria.png');

    icontrolconsulta.Picture.LoadFromFile('imagens\menu\controlconsulta.png');

    icontrolseguranca.Picture.LoadFromFile('imagens\menu\controlseguranca.png');

    icontrolbiblioteca.Picture.LoadFromFile('imagens\menu\controlbiblioteca.png');

    iverview.Picture.LoadFromFile('imagens\menu\view.png');

    isecretaria.Picture.LoadFromFile('imagens\menu\secretaria.png');

    ialuno.Picture.LoadFromFile('imagens\menu\secretaria\aluno.png');

    iassociado.Picture.LoadFromFile('imagens\menu\secretaria\associado.png');

    ivoluntario.Picture.LoadFromFile('imagens\menu\secretaria\voluntario.png');

    iturma.Picture.LoadFromFile('imagens\menu\secretaria\turma.png');

```



```

icurso.Picture.LoadFromFile('imagens\menu\secretaria\curso.png');

iseguranca.Picture.LoadFromFile('imagens\menu\seguranca.png');

ibackup.Picture.LoadFromFile('imagens\menu\seguranca\backup.png');

irestaura-
cao.Picture.LoadFromFile('imagens\menu\seguranca\restauracaobackup.png');

iusuarios.Picture.LoadFromFile('imagens\menu\seguranca\usuarios.png');

ilogeventos.Picture.LoadFromFile('imagens\menu\seguranca\logeventos.png');

ibiblioteca.Picture.LoadFromFile('imagens\menu\biblioteca.png');

ilivros.Picture.LoadFromFile('imagens\menu\biblioteca\livros.png');

ilocador.Picture.LoadFromFile('imagens\menu\biblioteca\locador.png');

igenero.Picture.LoadFromFile('imagens\menu\biblioteca\genero.png');

ilocacao.Picture.LoadFromFile('imagens\menu\biblioteca\locacao.png');

iarquivo.Picture.LoadFromFile('imagens\menu\arquivo.png');

isair.Picture.LoadFromFile('imagens\menu\arquivo\sair.png');

ilogoff.Picture.LoadFromFile('imagens\menu\arquivo\logoff.png');

ihelp.Picture.LoadFromFile('imagens\menu\help.png');

iajuda.Picture.LoadFromFile('imagens\menu\help\ajuda.png');

isobre.Picture.LoadFromFile('imagens\menu\help\sobre.png');

iconsulta.Picture.LoadFromFile('imagens\menu\consulta.png');

iconsultageral.Picture.LoadFromFile('imagens\menu\consulta\consultageral.png');

iconsultatur-
ma.Picture.LoadFromFile('imagens\menu\consulta\consultaturma.png');

centrov:=image1.Top;

```

```

centroh:=image1.Left;

Image1.left:=(screen.Width div 2)-(image1.Width div 2);
image1.top:=(screen.Height div 2)-(image1.height div 2);
centroh:=centroh-image1.Left;
centrov:=centrov-image1.Top;

for x:=0 to ComponentCount - 1 do
begin
    if Components[x].ClassName = 'TImage' then
    begin
        if((Components[x] as TImage).Name <> 'Image1') then
        begin
            (Components[x] as Timage).left := (Components[x] as Timage).left-centroh;
            (Components[x]as TImage).top:=(Components[x]as TImage).top-centrov;
        end;
    end;
end;

end;

procedure TFrmPrincipal.Volunrio1Click(Sender: TObject);
begin
    If(frmcadvol = nil)then
    begin
        Application.CreateForm(Tfrmcadvol, frmcadvol);
    end;
end;

```

```

    frmcadvol.Show;

    close;

end;

end;

procedure TFrmPrincipal.HistoricoAluno1Click(Sender: TObject);

begin

    If(frmCadHistorico = nil)then

    begin

        Application.CreateForm(TfrmCadHistorico, frmCadHistorico);

        frmCadHistorico.Show;

        close;

    end;

end;

procedure TFrmPrincipal.Locao1Click(Sender: TObject);

begin

    If(FrmLocLivro = nil)then

    begin

        Application.CreateForm(Tfrmloclivro, frmloclivro);

        frmloclivro.Show;

        close;

    end;

end;

end;

```

```
procedure TFrmPrincipal.Usurio1Click(Sender: TObject);
```

```
begin
```

```
  If(Frmcadusuário = nil)then
```

```
  begin
```

```
    Application.CreateForm(TFrmcadusuário, Frmcadusuário);
```

```
    Frmcadusuário.Show;
```

```
  close;
```

```
  end;
```

```
end;
```

```
procedure TFrmPrincipal.Backup1Click(Sender: TObject);
```

```
begin
```

```
  SaveDialog1.Execute;
```

```
  copyfile(pchar('bd\bdeclipse.mdb'),pchar(SaveDialog1.filename),true);
```

```
end;
```

```
procedure TFrmPrincipal.RestauraodeBackup1Click(Sender: TObject);
```

```
begin
```

```
  OpenFileDialog1.Execute;
```

```
  If messagedlg('Tem certeza que deseja substituir o Banco de dados?'+#13+'Essa  
operação é irreversível!', mtconfirmation, [mbyes, mbno], 0) = mryes then
```

```
  begin
```

```
    Application.CreateForm(Tfrmconfirmsenha,frmconfirmsenha);
```

```
    frmconfirmsenha.ShowModal;
```

```

if conf=true then

begin

    copyfile(pchar(OpenDialog1.filename),pchar('bd\bdeclipse.mdb'),false);

end

else if cancel<> true then

begin

    showmessage('Senha Incorreta'+#13+'A seção será encerrada.');
```

close;

```

end;

end;

end;

procedure TFrmPrincipal.isecretariaClick(Sender: TObject);

var w:boolean;

begin

    if User_id.nivel=2 then

        showmessage('Você não tem permissão para acessar essa área')

    else

        begin

            w:=true;

            i:=isecretaria;

            if i.Width=112 then

                w:=false;
```

```

tamanho;

if w=false then

begin

    ialuno.Visible:=true;

    ivoluntario.Visible:=true;

    iassociado.Visible:=true;

    icurso.Visible:=true;

    iturma.Visible:=true;

    icontrolsecretaria.Visible:=true;

    i.width:=132;

    i.Height:=121;

end;

end;

end;

procedure TFrmPrincipal.iarquivoClick(Sender: TObject);

var w:boolean;

begin

    w:=true;

    i:=iarquivo;

    if i.Width=112 then

        w:=false;

    tamanho;

```

```

if w=false then

begin

    i.width:=132;

    i.Height:=111;

    isair.Visible:=true;

    ilogoff.Visible:=true;

    icontrolarquivo.visible:=true;

end;

end;

procedure TFrmPrincipal.iconultaClick(Sender: TObject);

var w:boolean;

begin

    w:=true;

    i:=iconulta;

    if i.Width=112 then

        w:=false;

    tamanho;

    if w=false then

        begin

            i.width:=132;

            i.Height:=111;

            iconultageral.Visible:=true;

```

```

        icontrolconsulta.Visible:=true;

        iconsultaturma.Visible:=true;

    end;

end;

procedure TFrmPrincipal.ibibliotecaClick(Sender: TObject);

var w:boolean;

begin

    if User_id.nivel=1 then

        showmessage('Você não tem permissão para acessar essa área')

    else

        begin

            w:=true;

            i:=ibiblioteca;

            if i.Width=112 then

                w:=false;

            tamanho;

            if w=false then

                begin

                    i.width:=132;

                    i.Height:=111;

                    ilocador.Visible:=true;

                    ilocacao.Visible:=true;

```



```

        ilivros.Visible:=true;

        igenero.Visible:=true;

        icontrolbiblioteca.Visible:=true;

    end;

end;

end;

procedure TFrmPrincipal.isegurancaClick(Sender: TObject);

var w:boolean;

begin

    if User_id.nivel<>3 then

        showmessage('Você não tem permissão para acessar essa área')

    else

        begin

            w:=true;

            i:=iseguranca;

            if i.Width=112 then

                w:=false;

            tamanho;

            if w=false then

                begin

                    i.width:=132;

                    i.Height:=111;

```

```

    iusuarios.Visible:=true;

    ilogeventos.Visible:=true;

    icontrolseguranca.Visible:=true;

    ibackup.Visible:=true;

    irestauracao.Visible:=true;

end;

end;

end;

procedure TFrmPrincipal.tamanho;

var x:integer;

begin

    for x:=0 to ComponentCount - 1 do

        begin

            if Components[x].ClassName = 'TImage' then

                begin

                    if (((Components[x] as TImage).Name = 'ihelp')or((Components[x] as TImage).Name = 'iarquivo')or((Components[x] as TImage).Name = 'isecretaria')or((Components[x] as TImage).Name = 'ibiblioteca')or((Components[x] as TImage).Name = 'iconsulta')or((Components[x] as TImage).Name = 'iseguranca'))and((Components[x] as TImage).Width<>100) then

                        begin

                            if ((Components[x] as TImage).Name = 'ihelp') then

                                begin;

                                    (Components[x] as TImage).width := 100;

```

```

        (Components[x]as TImage).Height:=100;

        (Components[x] as TImage).left:=(screen.Width div 2)-((Components[x] as
TImage).Width div 2);

        (Components[x] as TImage).top:=(screen.Height div 2)-((Components[x] as
TImage).height div 2);

    end

    else

    begin

        (Components[x] as TImage).width := 112;

        (Components[x]as TImage).Height:=91;

    end;

end

    else if ((Components[x] as TImage).Name <> 'iview')and((Components[x] as
TImage).Name <> 'ihelp')and((Components[x] as TImage).Name <> 'Image1')and
((Components[x] as TImage).Name <> 'iarquivo')and((Components[x] as TI-
mage).Name <> 'isecretaria')and((Components[x] as TImage).Name <>
'ibiblioteca')and((Components[x] as TImage).Name <>
'iconsulta')and((Components[x] as TImage).Name <> 'iseguranca') then

    begin

        (Components[x]as TImage).Visible:=false;

    end;

end;

end;

end;

end;

procedure TFrmPrincipal.ihelpClick(Sender: TObject);

```

```

var w:boolean;

begin

    w:=true;

    i:=ihelp;

    if i.Width=100 then

        w:=false;

    tamanho;

    if w=false then

        begin

            i.width:=150;

            i.Height:=150;

            iajuda.Visible:=true;

            isobre.Visible:=true;

            icontrólhelp.Visible:=true;

        end;

        i.left:=(screen.Width div 2)-(i.Width div 2);

        i.top:=(screen.Height div 2)-(i.height div 2);

    end;

procedure TFrmPrincipal.iajudaClick(Sender: TObject);

begin

    ShellExecute(Handle, 'open','help\eclipsehelp.htm', "", "", 1);

```

```

    ihelpClick(ihelp);

end;

procedure TFrmPrincipal.iviewClick(Sender: TObject);

begin

    rewrite(p);

    writeln(p,crypt('C','1900'));

    closefile(p);

    menup:='1900';

    application.CreateForm(tfrmprincipal2,frmprincipal2);

    frmprincipal2.show;

    close;

end;

end.

unit UPrincipal2;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

    Dialogs, StdCtrls, Menus, XPMan, ComCtrls, Buttons, ToolWin, shellapi,ExtCtrls,

    JvGIF, jpeg;

type

```

TFrmPrincipal2 = class(TForm)

StatusBar1: TStatusBar;

MainMenu1: TMainMenu;

Arquivo1: TMenuItem;

Sair1: TMenuItem;

Cadastro1: TMenuItem;

Aluno1: TMenuItem;

Associado1: TMenuItem;

Curso1: TMenuItem;

Livro1: TMenuItem;

Locador1: TMenuItem;

Gnero1: TMenuItem;

Usurio1: TMenuItem;

Turma1: TMenuItem;

Voluntrio1: TMenuItem;

Segurana1: TMenuItem;

Backup1: TMenuItem;

Logdoprograma1: TMenuItem;

Consulta1: TMenuItem;

Geral1: TMenuItem;

Sair2: TMenuItem;

HistoricoAluno1: TMenuItem;

```
Biblioteca1: TMenuItem;  
  
Locao1: TMenuItem;  
  
RestauraodeBackuo1: TMenuItem;  
  
OpenDialog1: TOpenDialog;  
  
SaveDialog1: TSaveDialog;  
  
Help1: TMenuItem;  
  
Help2: TMenuItem;  
  
SobreoEclipse1: TMenuItem;  
  
Mudarmododeexibio1: TMenuItem;  
  
procedure Sair1Click(Sender: TObject);  
  
procedure Aluno1Click(Sender: TObject);  
  
procedure FormKeyPress(Sender: TObject; var Key: Char);  
  
procedure Associado1Click(Sender: TObject);  
  
procedure Cargo1Click(Sender: TObject);  
  
procedure Curso1Click(Sender: TObject);  
  
procedure Gnero1Click(Sender: TObject);  
  
procedure Livro1Click(Sender: TObject);  
  
procedure Locador1Click(Sender: TObject);  
  
procedure Turma1Click(Sender: TObject);  
  
procedure Geral1Click(Sender: TObject);  
  
procedure FormCreate(Sender: TObject);  
  
procedure Voluntrio1Click(Sender: TObject);
```

```

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Sair2Click(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure HistoricoAluno1Click(Sender: TObject);

procedure Locao1Click(Sender: TObject);

procedure Usurio1Click(Sender: TObject);

procedure Backup1Click(Sender: TObject);

procedure RestauraodeBackuo1Click(Sender: TObject);

procedure Help2Click(Sender: TObject);

procedure Mudarmododeexibio1Click(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmPrincipal2: TFrmPrincipal2;

    r:boolean;

implementation

uses API_F11, Math, UCadAluno, UCadAssociado, UCadCargo, UCadCurso,

    UCadGenero, UCadLivro, UCadLocador, UCadTurma, UpesqGeral, Ucadvol,

```



```

    UcadHistorico, ULocLivro, Ucadusu, ULogin, Uconf, UPrincipal;

{$R *.dfm}

procedure TFrmPrincipal2.FormClose(Sender: TObject;
    var Action: TCloseAction);
begin
    action:= cafree;

    FrmPrincipal2:= nil;
end;

procedure TFrmPrincipal2.Sair2Click(Sender: TObject);
begin
    Application.Terminate;
end;

procedure TFrmPrincipal2.Sair1Click(Sender: TObject);
begin
    frmlogin.show;

    Close;
end;

procedure TFrmPrincipal2.Aluno1Click(Sender: TObject);
begin
    If(FrmAluno = nil)then
    begin
        Application.CreateForm(TFrmAluno, FrmAluno);
    end;
end;

```

```

    FrmAluno.Show;

    FrmPrincipal2.Enabled:=false;

end;

end;

procedure TFrmPrincipal2.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

procedure TFrmPrincipal2.Associado1Click(Sender: TObject);

begin

    If(FrmAssociado = nil)then

        begin

            Application.CreateForm(TFrmAssociado, FrmAssociado);

            FrmAssociado.Show;

            FrmPrincipal2.Enabled:=false;

        end;

    end;

procedure TFrmPrincipal2.Cargo1Click(Sender: TObject);

```

```

begin

  If(FrmCargo = nil)then

    begin

      Application.CreateForm(TFrmCargo, FrmCargo);

      FrmCargo.Show;

      FrmPrincipal2.Enabled:=false;

    end;

  end;

procedure TFrmPrincipal2.Curso1Click(Sender: TObject);

begin

  If(FrmCurso = nil)then

    begin

      Application.CreateForm(TFrmCurso, FrmCurso);

      FrmCurso.Show;

      FrmPrincipal2.Enabled:=false;

    end;

  end;

procedure TFrmPrincipal2.Gnero1Click(Sender: TObject);

begin

  If(FrmGenero = nil)then

    begin

      Application.CreateForm(TFrmGenero, FrmGenero);

```

```

    FrmGenero.Show;

    FrmPrincipal2.Enabled:=false;

end;

end;

procedure TFrmPrincipal2.Livro1Click(Sender: TObject);

begin

    If(FrmLivro = nil)then

    begin

        Application.CreateForm(TFrmLivro, FrmLivro);

        FrmLivro.Show;

        FrmPrincipal2.Enabled:=false;

    end;

end;

procedure TFrmPrincipal2.Locador1Click(Sender: TObject);

begin

    If(FrmLocador = nil)then

    begin

        Application.CreateForm(TFrmLocador, FrmLocador);

        FrmLocador.Show;

        FrmPrincipal2.Enabled:=false;

    end;

end;

```

```
procedure TFrmPrincipal2.Turma1Click(Sender: TObject);
```

```
begin
```

```
  If(FrmTurma = nil)then
```

```
  begin
```

```
    Application.CreateForm(TFrmTurma, FrmTurma);
```

```
    FrmTurma.Show;
```

```
    FrmPrincipal2.Enabled:=false;
```

```
  end;
```

```
end;
```

```
procedure TFrmPrincipal2.Geral1Click(Sender: TObject);
```

```
begin
```

```
  If(Frmpesq = nil)then
```

```
  begin
```

```
    Application.CreateForm(TFrmpesq, Frmpesq);
```

```
    Frmpesq.Show;
```

```
    FrmPrincipal2.Enabled:=false;
```

```
  end;
```

```
end;
```

```
procedure TFrmPrincipal2.FormCreate(Sender: TObject);
```

```
begin
```

```
  String_tb := TString_tb.Create;
```

```
  r:=true;
```

```

end;

procedure TFrmPrincipal2.Volunrio1Click(Sender: TObject);
begin
    If(frmcadvol = nil)then
    begin
        Application.CreateForm(Tfrmcadvol, frmcadvol);
        frmcadvol.Show;
        FrmPrincipal2.Enabled:=false;
    end;
end;

procedure TFrmPrincipal2.FormShow(Sender: TObject);
begin
    if(User_id.nível = 1)then
    begin
        Cadastro1.Enabled:=true;
        Biblioteca1.Enabled:=false;
        Segurana1.Enabled:=false;
    end;

    if(User_id.nível = 2)then
    begin
        Cadastro1.Enabled:=false;
        Biblioteca1.Enabled:=true;
    end;
end;

```

```

    Segurana1.Enabled:=false;

end;

if(User_id.nível = 3)then

begin

    Cadastro1.Enabled:=true;

    Biblioteca1.Enabled:=true;

    Segurana1.Enabled:=true;

end;

FrmPrincipal2.Enabled := true;

end;

procedure TFrmPrincipal2.Button2Click(Sender: TObject);

//var x : integer;

begin

    // for x:=0 To Cadastro1.Count do

    //  showmessage(IntToStr(Cadastro1.Count));

end;

procedure TFrmPrincipal2.HistoricoAluno1Click(Sender: TObject);

begin

    If(frmCadHistorico = nil)then

    begin

        Application.CreateForm(TfrmCadHistorico, frmCadHistorico);

```

```

    frmCadHistorico.Show;

    FrmPrincipal2.Enabled:=false;

end;

end;

procedure TFrmPrincipal2.Locao1Click(Sender: TObject);

begin

    If(FrmLocLivro = nil)then

    begin

        Application.CreateForm(Tfrmloclivro, frmloclivro);

        frmloclivro.Show;

        FrmPrincipal2.Enabled:=false;

    end;

end;

procedure TFrmPrincipal2.Usurio1Click(Sender: TObject);

begin

    If(Frmcadusuario = nil)then

    begin

        Application.CreateForm(TFrmcadusuario, Frmcadusuario);

        Frmcadusuario.Show;

        FrmPrincipal2.Enabled:=false;

    end;

end;

```



```

procedure TFrmPrincipal2.Backup1Click(Sender: TObject);

begin

    SaveDialog1.Execute;

    copyfile(pchar('bd\bdeclipse.mdb'),pchar(SaveDialog1.filename),true);

end;

procedure TFrmPrincipal2.RestauraodeBackuo1Click(Sender: TObject);

begin

    OpenFileDialog1.Execute;

    If messagedlg('Tem certeza que deseja substituir o Banco de dados?'+#13+'Essa
operação é irreversível!', mtconfirmation, [mbyes, mbno], 0) = mryes then

begin

    Application.CreateForm(Tfrmconfirmsenha,frmconfirmsenha);

    frmconfirmsenha.ShowModal;

    if conf=true then

begin

        copyfile(pchar(OpenDialog1.filename),pchar('bd\bdeclipse.mdb'),false);

end

    else if cancel<> true then

begin

        showmessage('Senha Incorreta'+#13+'A seção será encerrada.');
```

close;

end;

end;

```

end;

procedure TFrmPrincipal2.Help2Click(Sender: TObject);

begin

    ShellExecute(Handle, 'open','help\eclipsehelp.htm', "", "", 1);

end;

procedure TFrmPrincipal2.Mudarmododeexibio1Click(Sender: TObject);

begin

    rewrite(p);

    writeln(p,crypt('C','2900'));

    closefile(p);

    menup:='2900';

    Application.CreateForm(tfrmprincipal,frmprincipal);

    frmprincipal.show;

    close;

end;

end.

unit UDMeclipse;

interface

uses

    SysUtils, Classes, DB, ADODB;

type

    TDMeclipse = class(TDataModule)

```

ADOCBD: TADOConnection;
TBvoluntario: TADOQuery;
TBfnvoluntario: TADOQuery;
TBcarga: TADOQuery;
TBusuario: TADOQuery;
TBlistaespera: TADOQuery;
TBcontribuicao: TADOQuery;
TBlocador: TADOQuery;
TBlocacao: TADOQuery;
TBgenero: TADOQuery;
TBfnlocador: TADOQuery;
TBaluno: TADOQuery;
TBfnaluno: TADOQuery;
TBcurso: TADOQuery;
TBturma: TADOQuery;
TBhistorico: TADOQuery;
TBassociado: TADOQuery;
TBfnassociado: TADOQuery;
TBmulta: TADOQuery;
TBtpfone: TADOQuery;
TBmencao: TADOQuery;
TBvoluntariocodigo_vol: TAutoIncField;

TBvoluntariodtadm_vol: TDateTimeField;
TBvoluntarionome_vol: TWideStringField;
TBvoluntariosx_vol: TWideStringField;
TBvoluntariodtnasc_vol: TDateTimeField;
TBvoluntariorg_vol: TWideStringField;
TBvoluntariocpf_vol: TWideStringField;
TBvoluntarionacional_vol: TWideStringField;
TBvoluntarionatural_vol: TWideStringField;
TBvoluntarioend_vol: TWideStringField;
TBvoluntarionum_vol: TIntegerField;
TBvoluntariocompl_vol: TWideStringField;
TBvoluntariobairro_vol: TWideStringField;
TBvoluntariocep_vol: TWideStringField;
TBvoluntariocid_vol: TWideStringField;
TBvoluntariouf_vol: TWideStringField;
TBvoluntarioemail_vol: TWideStringField;
TBvoluntarioescolar_vol: TWideStringField;
TBvoluntariofrmcao_vol: TWideStringField;
TBvoluntarioprofis_vol: TWideStringField;
TBvoluntarioaptid_vol: TWideStringField;
TBvoluntariotrabassoci_vol: TWideStringField;
TBvoluntarioarea_vol: TWideStringField;

TBvoluntariodias_vol: TWideStringField;
TBvoluntariohora_vol: TWideStringField;
TBvoluntarioindicacao_vol: TWideStringField;
TBfnvoluntariocodigo_vol: TIntegerField;
TBfnvoluntariotpfon_vol: TWideStringField;
TBfnvoluntariofone_vol: TWideStringField;
TBlistaesperacodigo_alu: TIntegerField;
TBlistaesperadtfecha_esp: TDateTimeField;
TBcargocodigo_vol: TIntegerField;
TBcargocargo_vol: TWideStringField;
TBusuariocodigo_usu: TAutoIncField;
TBusuarionome_usu: TWideStringField;
TBusuariosenha_usu: TWideStringField;
TBusuarionacesso_usu: TWideStringField;
TBusuariofrase_usu: TWideStringField;
TBcursocodigo_cur: TAutoIncField;
TBcursonome_cur: TWideStringField;
TBturmacodigo_tur: TAutoIncField;
TBturmacodigo_cur: TIntegerField;
TBturmacodigo_vol: TIntegerField;
TBturmanome_tur: TWideStringField;
TBturmadt_ini: TDateTimeField;

TBturmadt_prev: TDateTimeField;
TBturmadt_fin: TDateTimeField;
TBturmanumvagas_tur: TSmallintField;
TBturmanumalu_tur: TSmallintField;
TBturmaobs_tur: TMemoField;
TBturmastatus_tur: TWideStringField;
TBassociadocodigo_ass: TAutoIncField;
TBassociadodtadm_ass: TDateTimeField;
TBassociadonome_ass: TWideStringField;
TBassociadosx_ass: TWideStringField;
TBassociadodtnasc_ass: TDateTimeField;
TBassociadorg_ass: TWideStringField;
TBassociadocpf_ass: TWideStringField;
TBassociadonacional_ass: TWideStringField;
TBassociadonatural_ass: TWideStringField;
TBassociadoend_ass: TWideStringField;
TBassociadonum_ass: TIntegerField;
TBassociadocompl_ass: TWideStringField;
TBassociadobairro_ass: TWideStringField;
TBassociadocep_ass: TWideStringField;
TBassociadocid_ass: TWideStringField;
TBassociadouf_ass: TWideStringField;

TBassociadoemail_ass: TWideStringField;
TBassociadoescolar_ass: TWideStringField;
TBassociadofrmcao_ass: TWideStringField;
TBassociadoaptid_ass: TWideStringField;
TBassociadoconhece_ass: TWideStringField;
TBassociadoprofis_ass: TWideStringField;
TBassociadoindicacao_vol: TWideStringField;
TBfnassociadocodigo_ass: TIntegerField;
TBfnassociadotpfone_fnass: TWideStringField;
TBfnassociadofone_fnass: TWideStringField;
TBcontribuicaocodigo_con: TAutoIncField;
TBcontribuicaocodigo_ass: TIntegerField;
TBcontribuicaovlr_con: TIntegerField;
TBcontribuicaodtpagto_con: TDateTimeField;
TBlocacaocodigo_locacao: TAutoIncField;
TBlocacaocodigo_loc: TIntegerField;
TBlocacaodt_locado: TDateTimeField;
TBlocacaodtprev_locac: TDateTimeField;
TBlocacaodtentg_locac: TDateTimeField;
TBlocacaopendencia_locac: TWideStringField;
TBlocacaoobs_locac: TMemoField;
TBlocadorcodigo_loc: TAutoIncField;

TBlocadornome_loc: TWideStringField;
TBlocadorsx_loc: TWideStringField;
TBlocadorrg_loc: TWideStringField;
TBlocadorcpf_loc: TWideStringField;
TBlocadorend_loc: TWideStringField;
TBlocadorcid_loc: TWideStringField;
TBlocadoruf_loc: TWideStringField;
TBlocadoremail_loc: TWideStringField;
TBlocadorBlock_loc: TWideStringField;
TBgenerocodigo_gen: TAutoIncField;
TBgeneronome_gen: TWideStringField;
TBgenerocor_gen: TWideStringField;
TBfnlocadorcodigo_loc: TIntegerField;
TBfnlocadortpfone_fnloc: TWideStringField;
TBfnlocadorfone_fnloc: TWideStringField;
TBmultacodigo_mul: TAutoIncField;
TBmultacodigo_loca: TIntegerField;
TBmultamultad_mul: TSmallintField;
TBmultadatrazo_mul: TSmallintField;
TBmultatotal_mul: TIntegerField;
TBtpfonesigla_tf: TWideStringField;
TBtpfonetpfone_tf: TWideStringField;

TBtpfonestatus_tf: TWideStringField;
TBmencaonota_men: TWideStringField;
DStbaluno: TDataSource;
DStbassociado: TDataSource;
DStbcargo: TDataSource;
DStbcontribuicao: TDataSource;
DStbcurso: TDataSource;
DStbfnaluno: TDataSource;
DSfnassociado: TDataSource;
DStbfnlocador: TDataSource;
DSfnvoluntario: TDataSource;
DStbgenero: TDataSource;
DStbhistorico: TDataSource;
DStblistaespera: TDataSource;
DStblivro: TDataSource;
DStblocacao: TDataSource;
DStblocador: TDataSource;
DStbmencao: TDataSource;
DStbmulta: TDataSource;
DStbtpfone: TDataSource;
DStbturma: TDataSource;
DStbusuario: TDataSource;

DStbvoltario: TDataSource;

TBlivro: TADOQuery;

TBlocacaotombo_liv: TIntegerField;

TBlistaesperacodigo_cur: TIntegerField;

TBfnalunocodigo_alu: TIntegerField;

TBfnalunotptel_fnalu: TWideStringField;

TBfnalunofone_fnalu: TWideStringField;

TBalunocodigo_alu: TAutoIncField;

TBalunonome_alu: TWideStringField;

TBalunosx_alu: TWideStringField;

TBalunodtnasc_alu: TDateTimeField;

TBalunorg_alu: TWideStringField;

TBalunonacional_alu: TWideStringField;

TBalunonatural_alu: TWideStringField;

TBalunoend_alu: TWideStringField;

TBalunonum_alu: TIntegerField;

TBalunocompl_alu: TWideStringField;

TBalunobairro_alu: TWideStringField;

TBalunocep_alu: TWideStringField;

TBalunocid_alu: TWideStringField;

TBalunouf_alu: TWideStringField;

TBalunomae_alu: TWideStringField;

TBalunoprofmae_alu: TWideStringField;
TBalunorgmae_alu: TWideStringField;
TBalunocpfmae_alu: TWideStringField;
TBalunopai_alu: TWideStringField;
TBalunoprofpai_alu: TWideStringField;
TBalunorgpai_alu: TWideStringField;
TBalunocfppai_alu: TWideStringField;
TBalunorenda_alu: TIntegerField;
TBalunocsap_alu: TWideStringField;
TBalunovlalug_alu: TIntegerField;
TBalunoqt_resid_alu: TIntegerField;
TBalunoconhece_alu: TWideStringField;
TBalunoestuda_alu: TWideStringField;
TBalunotpesc_alu: TWideStringField;
TBalunonomeesc_alu: TWideStringField;
TBalunoserieesc_alu: TWideStringField;
TBalunoendesc_alu: TWideStringField;
TBalunobairroesc_alu: TWideStringField;
TBalunocepesc_alu: TWideStringField;
TBalunocidesc_alu: TWideStringField;
TBalunoufesc_alu: TWideStringField;
TBalunodtcad_alu: TDateTimeField;

```

TBalunocodigo_vol: TIntegerField;

TBalunofoto: TBlobField;

TBalunoStatus: TWideStringField;

TBalunoemail: TWideStringField;

TBlivrotombo_liv: TIntegerField;

TBlivronome_liv: TWideStringField;

TBlivroautor_liv: TWideStringField;

TBlivrosinopse_liv: TMemoField;

TBlivrocodigo_gen: TIntegerField;

TBlivrosub_cod_gen: TIntegerField;

TBlivrodisp_liv: TWideStringField;

TBhistoricocodigo_alu: TIntegerField;

TBhistoricocodigo_tur: TIntegerField;

TBhistoriconfaltas: TWideStringField;

TBhistoricostatus_alu: TWideStringField;

TBhistoricoobs_alu: TMemoField;

TBhistoricomediafinal_alu: TWideStringField;

TBhistoricocodigo_hist: TAutoIncField;

private

{ Private declarations }

public

{ Public declarations }

```

```

end;

var
    DMeclipse: TDMeclipse;

implementation

{$R *.dfm}

end.

unit Unit1;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Grids, DBGrids, ExtCtrls, DB, ADODB, Buttons;

type

    TfrmPesq_hist = class(TForm)

        Panel1: TPanel;

        DBGrid1: TDBGrid;

        ComboBox1: TComboBox;

        Edit1: TEdit;

        ADOpesq_hist: TADOQuery;

        ADOpesq_histnome_alu: TWideStringField;

        ADOpesq_histnome_cur: TWideStringField;

        ADOpesq_histnfaltas: TWideStringField;

        ADOpesq_histstatus_alu: TWideStringField;

```

```

ADOPesq_histobs_alu: TMemoField;

ADOPesq_histmediafinal_alu: TWideStringField;

ADOPesq_histdt_ini: TDateTimeField;

ADOPesq_histnome_tur: TWideStringField;

ADOPesq_histdt_fin: TDateTimeField;

ADOPesq_histdt_prev: TDateTimeField;

ADOPesq_histhorario_tur: TWideStringField;

dspesq_hist: TDataSource;

ADOPesq_histcodigo_alu: TAutoIncField;

Button1: TButton;

ADOPesq_histcodigo_hist: TAutoIncField;

ComboBox2: TComboBox;

ADOCURSO: TADOQuery;

ADOCURSOcodigo_cur: TAutoIncField;

ADOCURSOnome_cur: TWideStringField;

Btnloc: TBitBtn;

CheckBox1: TCheckBox;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure DBGrid1DblClick(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure FormShow(Sender: TObject);

```

```

    procedure BtnlocClick(Sender: TObject);

    procedure CheckBox1Click(Sender: TObject);

private
    { Private declarations }

public
    { Public declarations }

end;

var
    frmPesq_hist: TfrmPesq_hist;

implementation

uses API_F11, UpesqGeral, UDMeclipse, Uatualizar;

{$R *.dfm}

procedure TfrmPesq_hist.FormClose(Sender: TObject;
    var Action: TCloseAction);

begin
    ADOpesq_hist.Close;

    ADOpesq_hist.Active := false;

    action:= cafree;

    frmPesq_hist:= nil;

end;

procedure TfrmPesq_hist.DBGrid1DblClick(Sender: TObject);

begin

```

```

DMeclipse.TBhistorico.Close;

DMeclipse.TBhistorico.SQL.Text := 'SELECT * FROM TB_HISTORICO WHERE
CODIGO_HIST = :CODI';

DMeclipse.TBhistorico.Parameters.ParamByName('CODI').Value      :=      ADO-
pesq_histcodigo_hist.AsInteger;

DMeclipse.TBhistorico.Open;

//          DMeclipse.TBhistoricocodigo_tur.AsInteger      :=      ADO-
pesq_histcodigo_hist.AsInteger;

Button1.Click;

end;

procedure TfrmPesq_hist.Button1Click(Sender: TObject);

begin

    Close;

end;

procedure TfrmPesq_hist.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

end;

```



```

procedure TfrmPesq_hist.FormShow(Sender: TObject);

begin

    ADOpesq_hist.Active := true;

    ADOCURSO.Open;

    if not(ADOCURSO.IsEmpty)then

    begin

        ADOCURSO.First;

        while not ADOCURSO.Eof do

        begin

            ComboBox2.Items.Add(ADOCURSOname_cur.AsString);

            ADOCURSO.Next;

        end;

        ComboBox2.ItemIndex := 0;

    end;

end;

procedure TfrmPesq_hist.BtnlocClick(Sender: TObject);

begin

    with ADOpesq_hist do

    begin

        Close;

        SQL.Clear;

        SQL.Add('SELECT tb_aluno.codigo_alu,

```

```

SQL.Add('      tb_aluno.nome_alu,                ');
SQL.Add('      tb_curso.nome_cur,                ');
SQL.Add('      tb_historico.codigo_hist,         ');
SQL.Add('      tb_historico.nfaltas,              ');
SQL.Add('      tb_historico.status_alu,           ');
SQL.Add('      tb_historico.obs_alu,              ');
SQL.Add('      tb_historico.mediafinal_alu,       ');
SQL.Add('      tb_turma.dt_ini,                   ');
SQL.Add('      tb_turma.nome_tur,                 ');
SQL.Add('      tb_turma.dt_fin,                   ');
SQL.Add('      tb_turma.dt_prev,                  ');
SQL.Add('      tb_turma.horario_tur               ');

SQL.Add('FROM (tb_curso INNER JOIN tb_turma ON tb_curso.codigo_cur =
tb_turma.codigo_cur)                ');

SQL.Add('      INNER JOIN (tb_aluno INNER JOIN tb_historico ON
tb_aluno.codigo_alu = tb_historico.codigo_alu)');

SQL.Add('      ON tb_turma.codigo_tur =
tb_historico.codigo_tur ');

SQL.Add('WHERE                ');

if(ComboBox1.ItemIndex = 1)then

SQL.Add('tb_aluno.codigo_alu = :codi                ');

if(ComboBox1.ItemIndex = 0)then

SQL.Add('tb_aluno.nome_alu Like :nome                ');

```

```

if(ComboBox1.ItemIndex = 2)then

    SQL.Add('tb_turma.dt_ini > :dt                ');

if(ComboBox2.Enabled)then

begin

    SQL.Add('and tb_curso.nome_cur like :Ntur');

    Parameters.ParamByName('Ntur').Value := ComboBox2.Text;

end;

if(ComboBox1.ItemIndex = 1)then

    Parameters.ParamByName('codi').Value := Edit1.Text;

if(ComboBox1.ItemIndex = 0)then

    Parameters.ParamByName('nome').Value := Edit1.Text;

if(ComboBox1.ItemIndex = 2)then

    Parameters.ParamByName('dt').Value := Edit1.Text;

Open;

end;

end;

procedure TfrmPesq_hist.CheckBox1Click(Sender: TObject);

begin

    ComboBox2.Enabled := CheckBox1.Checked;

end;

end.

```

```

unit Uconf;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls;

type

    Tfrmconfirmsenha = class(TForm)

        Edit1: TEdit;

        Button1: TButton;

        Button2: TButton;

        Label1: TLabel;

        Label3: TLabel;

        procedure Label3Click(Sender: TObject);

        procedure Button1Click(Sender: TObject);

        procedure Button2Click(Sender: TObject);

        procedure FormCreate(Sender: TObject);

        procedure FormClose(Sender: TObject; var Action: TCloseAction);

    private

        { Private declarations }

    public

        { Public declarations }

    end;

```

```

var

    frmconfirmsenha: Tfrmconfirmsenha;

    conf,cancel:boolean;

implementation

uses API_F11;

{$R *.dfm}

procedure Tfrmconfirmsenha.Label3Click(Sender: TObject);

begin

    showmessage(User_id.frase);

end;

procedure Tfrmconfirmsenha.Button1Click(Sender: TObject);

begin

    if Edit1.text= User_id.senha then conf:=true;

    close;

end;

procedure Tfrmconfirmsenha.Button2Click(Sender: TObject);

begin

    cancel:=true;

    close;

end;

procedure Tfrmconfirmsenha.FormCreate(Sender: TObject);

begin

```

```

cancel:=false;

conf:=false;

end;

procedure Tfrmconfirmsenha.FormClose(Sender: TObject;
var Action: TCloseAction);

begin

action:=cafree;

frmconfirmsenha:=nil;

end;

end.

unit ULogin;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, DB, ADODB, jpeg, ExtCtrls;

type

TFrmLogin = class(TForm)

Edit1: TEdit;

Edit2: TEdit;

Label1: TLabel;

Label2: TLabel;

```

```

Button1: TButton;

Button2: TButton;

Label3: TLabel;

val_Usu: TADOQuery;

val_Usucodigo_usu: TAutoIncField;

val_Usunome_usu: TWideStringField;

val_Ususenha_usu: TWideStringField;

val_Usunacesso_usu: TWideStringField;

val_Usufrase_usu: TWideStringField;

Image1: TImage;

procedure FormShow(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure Label3Click(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmLogin: TFrmLogin;

```

implementation

uses API_F11, UDMeclipse, UPrincipal, UPrincipal2, StrUtils;

{ \$R *.dfm }

procedure TFrmLogin.FormShow(Sender: TObject);

begin

 User_id := TUser_id.Create;

 if not (FileExists('parametros.txt')) then

 begin

 Assignfile(p,'parametros.txt');

 Rewrite(p);

 Writeln(p,crypt('C','2900'));

 closefile(p);

 end;

 Assignfile(p,'parametros.txt');

 reset(p);

 readln(p,menup);

 menup:=Crypt('D',menup);

 closefile(p);

end;

procedure TFrmLogin.Button1Click(Sender: TObject);

var senha:string;

begin


```

with val_Usu do

begin

    Close;

    SQL.Clear;

    SQL.Add('SELECT *
            ');

    SQL.Add(' FROM TB_USUARIO T
            ');

    SQL.Add(' WHERE T.NOME_USU = :NOME ');

    Parameters.ParamByName('NOME').Value := Edit1.Text;

    Open;

    if not(IsEmpty)then

begin

    senha:= Crypt('D',val_Ususenha_usu.asstring);

    if senha<>Edit2.Text then

begin

    showmessage('Senha Incorreta');

    edit2.SetFocus;

end

else

begin

    User_id.nome := val_Usunome_usu.AsString;

    User_id.nível := val_Usunacesso_usu.AsInteger;

    user_id.senha:=Crypt('D',val_Ususenha_usu.AsString);

```

```

User_id.frase:=val_Usufrase_usu.AsString;

If(FrmPrincipal = nil)then

begin

    Edit1.Text:="";

    edit2.Text:="";

    frmlogin.Hide;

    if menup='1900' then

        begin

            application.CreateForm(TFrmPrincipal2,FrmPrincipal2);

            FrmPrincipal2.Show;

        end

    else

        begin

            application.CreateForm(TFrmPrincipal,FrmPrincipal);

            FrmPrincipal.Show;

        end;

        niveldeacesso(user_id.nivel);

    end;

end;

end

else

begin

```

```

    ShowMessage('Login inválido');

    Edit1.Clear;

    Edit2.Clear;

    edit1.SetFocus;

end;

end;

end;

procedure TFrmLogin.Button2Click(Sender: TObject);

begin

    Application.Terminate;

end;

procedure TFrmLogin.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

end;

procedure TFrmLogin.Label3Click(Sender: TObject);

begin

```

```

with val_Usu do

begin

    Close;

    SQL.Clear;

    SQL.Add('SELECT *
            ');

    SQL.Add(' FROM TB_USUARIO T
            ');

    SQL.Add(' WHERE T.NOME_USU = :NOME
            ');

    Parameters.ParamByName('NOME').Value := Edit1.Text;

    Open;

    if not (isempty) then

begin

    showmessage('Frase:'+#13+ val_Usufrase_usu.AsString);

end

else

    showmessage('Usuário não encontrado.')

end;

end;

end.

unit pesq_t;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

```

Dialogs, StdCtrls, Grids, DBGrids, ExtCtrls, DB, ADODB;

type

TfrmPesq_tur = class(TForm)

Panel1: TPanel;

DBGrid1: TDBGrid;

ComboBox1: TComboBox;

Edit1: TEdit;

Button1: TButton;

dspesq_t: TDataSource;

ADOpesq_t: TADOQuery;

ADOpesq_tcodigo_tur: TAutoIncField;

ADOpesq_tnome_tur: TWideStringField;

ADOpesq_tnome_cur: TWideStringField;

ADOpesq_tstatus_tur: TWideStringField;

ADOpesq_tnome_vol: TWideStringField;

ADOpesq_tdt_ini: TDateTimeField;

procedure Edit1Exit(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure DBGrid1DblClick(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure Button1Click(Sender: TObject);

procedure FormShow(Sender: TObject);

```

private

    { Private declarations }

public

    { Public declarations }

end;

var

    frmPesq_tur: TfrmPesq_tur;

implementation

uses API_F11, UpesqGeral, UDMecclipse, Uatualizar;

{$R *.dfm}

procedure TfrmPesq_tur.Edit1Exit(Sender: TObject);

begin

    with ADOpesq_t do

    begin

        Close;

        SQL.Clear;

        SQL.Add('SELECT tb_turma.codigo_tur,                ');

        SQL.Add('    tb_turma.nome_tur,                    ');

        SQL.Add('    tb_curso.nome_cur,                    ');

        SQL.Add('    tb_turma.status_tur,                  ');

        SQL.Add('    tb_voluntario.nome_vol,                ');

        SQL.Add('    tb_turma.dt_ini                        ');
    end;

```

```

SQL.Add('      FROM tb_voluntario      ');
SQL.Add('      INNER JOIN (tb_curso      ');
SQL.Add('      INNER JOIN tb_turma      ');
SQL.Add('      ON tb_curso.codigo_cur = tb_turma.codigo_cur));
SQL.Add('      ON tb_voluntario.codigo_vol = tb_turma.codigo_vol ');
SQL.Add('WHERE      ');

if(ComboBox1.ItemIndex = 1)then

    SQL.Add('tb_turma.codigo_tur = :codi      ');

if(ComboBox1.ItemIndex = 0)then

    SQL.Add('tb_turma.nome_tur Like :nome      ');

if(ComboBox1.ItemIndex = 2)then

    SQL.Add('tb_turma.dt_ini > :dt      ');


if(ComboBox1.ItemIndex = 1)then

    Parameters.ParamByName('codi').Value := Edit1.Text;

if(ComboBox1.ItemIndex = 0)then

    Parameters.ParamByName('nome').Value := Edit1.Text;

if(ComboBox1.ItemIndex = 2)then

    Parameters.ParamByName('dt').Value := Edit1.Text;

Open;

end;

end;

```

```

procedure TfrmPesq_tur.FormClose(Sender: TObject;
    var Action: TCloseAction);
begin
    ADOpesq_t.Close;

    ADOpesq_t.Active := false;

    action:= cafree;

    frmPesq_tur:= nil;

end;

procedure TfrmPesq_tur.DBGrid1DbClick(Sender: TObject);
begin
    DMeclipse.TBturma.Close;

    DMeclipse.TBturma.SQL.Text := 'SELECT * FROM TB_TURMA WHERE CODI-
GO_TUR = :CODI';

    DMeclipse.TBturma.Parameters.ParamByName('CODI').Value      :=      ADO-
pesq_tcodigo_tur.AsInteger;

    DMeclipse.TBturma.Open;

    Button1.Click;

end;

procedure TfrmPesq_tur.FormKeyPress(Sender: TObject; var Key: Char);
begin
    if(Key = #13)then
    begin
        Key := #13;
    end;
end;

```



```

        SelectNext(ActiveControl,true,true);

    end;

end;

procedure TfrmPesq_tur.Button1Click(Sender: TObject);

begin

    Close;

end;


procedure TfrmPesq_tur.FormShow(Sender: TObject);

begin

    ADOpesq_t.Active := true;

end;

end.

unit Uatualizar;

interface

uses classes;

type

    Tversao = class

        versao : String;

    end;

var

    vers : Tversao;

```

```

implementation

// 13/03/2009

begin

    vers := Tversao.Create;

    vers.versao := ' 0.03';

end.

unit Ucadusu;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

    Dialogs, StdCtrls, Buttons, ExtCtrls, DBCtrls, Mask, DB, ADODB;

type

    TFrmcadusuuario = class(TForm)

        Label1: TLabel;

        Label2: TLabel;

        Label3: TLabel;

        DBEdit3: TDBEdit;

        Label5: TLabel;

        DBEdit5: TDBEdit;

        DBText1: TDBText;

        Edit1: TEdit;

        Panel2: TPanel;

```

```

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

BitBtn5: TBitBtn;

BitBtn6: TBitBtn;

BitBtn7: TBitBtn;

DBRadioGroup1: TDBRadioGroup;

DBEdit2: TDBEdit;

usu_pesq: TADOQuery;

Label4: TLabel;

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure BitBtn6Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure DBEdit2Enter(Sender: TObject);

procedure DBEdit2Exit(Sender: TObject);

```

```

    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private
    { Private declarations }

public
    { Public declarations }

end;

var
    Frmcadusuario: TFrmcadusuario;

implementation

uses UDMeclipse, UpesqGeral, API_F11, ULogin, UPrincipal, UPrincipal2;

{$R *.dfm}

procedure TFrmcadusuario.BitBtn1Click(Sender: TObject);

begin
    botoes(self);

    Edit1.Enabled:=true;

    mudaedit(self, true);

    DMeclipse.tbusuario.Insert;

end;

procedure TFrmcadusuario.BitBtn2Click(Sender: TObject);

begin
    botoes(self);

    Edit1.Enabled:=true;

```

```

mudaedit(self, true);

DMeclipse.TBusuario.edit;

end;

procedure TFrmcadusuario.BitBtn3Click(Sender: TObject);

begin

    If messagedlg('Tem certeza que deseja excluir esse usuário?'+#13+'Essa operação
    é irreversível', mtconfirmation, [mbytes, mbno], 0) = mryes then

    begin

        if dbedit2.text='admin' then

            showmessage('Você não pode excluir o cadastro de Administrador')

        else

            DMeclipse.TBusuario.Delete;

        end;

    end;

end;

procedure TFrmcadusuario.BitBtn4Click(Sender: TObject);

begin

    botoes(self);

    Edit1.Enabled:=false;

    mudaedit(self, false);

    DMeclipse.TBusuario.cancel;

end;

procedure TFrmcadusuario.BitBtn5Click(Sender: TObject);

begin

```

```

dbedit2.Text:=trim(DBEdit2.text);

if dbedit3.Text <> edit1.Text then

begin

    showmessage('Senhas Incompatíveis');

    dbedit3.Text:="";

    edit1.Text:="";

    dbedit3.SetFocus;

end

else if (trim(dbedit3.text)="")or(trim(DBEdit2.text)="")or(DBRadioGroup1.Value="")
then

begin

    showmessage('Preencha corretamente todos os campos');

    DBEdit2.SetFocus;

end

else

begin

    with usu_pesq do

    begin

        Close;

        SQL.Text := 'select *          ';

        SQL.Add(' from tb_usuario      ');

        SQL.Add(' where nome_usu = :nome ');

        Parameters.ParamByName('nome').Value := DBEdit2.text;

```

```

Open;

if (IsEmpty)then
begin
    DBEdit3.Text:=Crypt('C',DBEdit3.text);

    DMeclipse.TBusuario.Post;

    DMeclipse.TBusuario.Refresh;

    botoes(self);

    Edit1.Enabled:=false;

    mudaedit(self, false);

end

else

begin

    showmessage('Usuário já existente, escolha outro nome');

    DBEdit2.SetFocus;

end;

end;

end;

end;

procedure TFrmcadusuario.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Usuario';

    String_tb.Scampocod := 'Codigo_usu';

```

```

String_tb.Scamponome := 'Nome_usu';

String_tb.Qrytb := DMeclipse.TBusuario;

String_tb.DStb := DMeclipse.DStbusuario;

abrirpesq(self);

end;

procedure TFrmcadusuario.FormClose(Sender: TObject; var Action: TCloseAction);

begin

    DMeclipse.TBusuario.active:=false;

    if menup='1900' then

    begin

        frmprincipal2.Enabled:=true;

    end

    else

    begin

        application.CreateForm(TFrmPrincipal,FrmPrincipal);

        FrmPrincipal.Show;

    end;

    action:= cafree;

    Frmcadusuario:= nil;

end;

procedure TFrmcadusuario.BitBtn6Click(Sender: TObject);

begin

```



```

close;

end;

procedure TFrmcadusuario.FormCreate(Sender: TObject);

begin

    DMeclipse.TBusuario.active:=true;

end;

procedure TFrmcadusuario.DBEdit2Enter(Sender: TObject);

begin

    if dbedit2.text='admin' then

        dbedit2.readonly:=true;

end;

procedure TFrmcadusuario.DBEdit2Exit(Sender: TObject);

begin

    DBEdit2.ReadOnly:=false;

end;

procedure TFrmcadusuario.FormCloseQuery(Sender: TObject;

    var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de usuá-

rios?', mtconfirmation, [mbyes, mbno], 0) = mryes then

        begin

            DMeclipse.TBusuario.cancel;

            canclose:=true;

```

```

end

else

    canclose:=false;

end;

end.

unit Ucadvol;


interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

    Dialogs, StdCtrls, Mask, DBCtrls, ExtCtrls, ComCtrls, Buttons;

type

    Tfrmcadvol = class(TForm)

        PageControl1: TPageControl;

        TabSheet1: TTabSheet;

        TabSheet2: TTabSheet;

        TabSheet3: TTabSheet;

        Label1: TLabel;

        DBEdit1: TDBEdit;

        Label3: TLabel;

        Label6: TLabel;

        DBEdit3: TDBEdit;

```

Label5: TLabel;

Label7: TLabel;

DBEdit7: TDBEdit;

Label8: TLabel;

ex: TDBEdit;

Label9: TLabel;

DBEdit9: TDBEdit;

DBRadioGroup1: TDBRadioGroup;

Label2: TLabel;

Label10: TLabel;

DBEdit10: TDBEdit;

Label11: TLabel;

DBEdit11: TDBEdit;

Label12: TLabel;

DBEdit12: TDBEdit;

Label13: TLabel;

DBEdit13: TDBEdit;

Label14: TLabel;

DBEdit14: TDBEdit;

Label15: TLabel;

DBEdit15: TDBEdit;

Label16: TLabel;

Label17: TLabel;
DBEdit17: TDBEdit;
Label18: TLabel;
Label19: TLabel;
DBEdit19: TDBEdit;
Label20: TLabel;
DBEdit20: TDBEdit;
Label21: TLabel;
DBEdit21: TDBEdit;
Label22: TLabel;
Label23: TLabel;
DBEdit23: TDBEdit;
Label24: TLabel;
DBEdit24: TDBEdit;
Label25: TLabel;
DBEdit25: TDBEdit;
Label26: TLabel;
DBEdit26: TDBEdit;
DBEdit6: TDBEdit;
DBComboBox1: TDBComboBox;
DBRadioGroup2: TDBRadioGroup;
Panel2: TPanel;

```
BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

BitBtn5: TBitBtn;

BitBtn6: TBitBtn;

BitBtn7: TBitBtn;

DBComboBox2: TDBComboBox;

BitBtn10: TBitBtn;

data2: TDateTimePicker;

data1: TDateTimePicker;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCreate(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure bloqletrarg(Sender: TObject; var Key: Char);
```

```

procedure bloqletra(Sender: TObject; var Key: Char);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure BitBtn10Click(Sender: TObject);


private

    { Private declarations }

public

    { Public declarations }

end;

var

    frmcadvol: Tfrmcadvol;

implementation

uses UDMeclipse, API_F11, UpesqGeral, DB, UPrincipal, UCadContato,
    UPrincipal2;

{$R *.dfm}

procedure Tfrmcadvol.FormClose(Sender: TObject; var Action: TCloseAction);

begin

    DMeclipse.TBvoluntario.Active := false;

    DMeclipse.TBfnvoluntario.Active := false;

    DMeclipse.TBcarga.Active := false;

    DMeclipse.TBtpfone.Active := false;

    if menup='1900' then

```

```

begin

    frmprincipal2.Enabled:=true;

end

else

begin

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

action:= cafree;

frmcadvol:= nil;

end;

procedure Tfrmcadvol.FormCreate(Sender: TObject);

begin

    DMeclipse.TBvoluntario.Active := true;

    DMeclipse.TBfnvoluntario.Active := true;

    DMeclipse.TBcarga.Active := true;

    DMeclipse.TBtpfone.Active := true;

    data1.Date:=DMeclipse.TBvoluntariodtadm_vol.AsDateTime;

    data2.Date:=dmeclipse.TBvoluntariodtnasc_vol.AsDateTime;

end;

procedure Tfrmcadvol.FormKeyPress(Sender: TObject; var Key: Char);

begin

```

```

if(Key = #13)then

begin

    Key := #13;

    SelectNext(ActiveControl,true,true);

end;

end;

procedure Tfrmcadvol.BitBtn1Click(Sender: TObject);

begin

    botoes(self);

    mudaedit(self, true);

    DMeclipse.TBvoluntario.Insert;

end;

procedure Tfrmcadvol.BitBtn2Click(Sender: TObject);

begin

    if not(DMeclipse.TBvoluntario.IsEmpty)then

    begin

        botoes(self);

        mudaedit(self, true);

        DMeclipse.TBvoluntario.Edit;

    end

    else

        ShowMessage('Não existe registro para edição.');
```



```

end;

procedure Tfrmcadvol.BitBtn3Click(Sender: TObject);
begin
    if dmeclipse.TBvoluntario.isempty then
    begin
        showmessage('Não existe registro a ser excluído');
        exit;
    end;

    If messagedlg('Tem certeza que deseja excluir esse Voluntário?'+#13+'Essa operação é irreversível!', mtconfirmation, [mbyes, mbno], 0) = mrno then
        exit;

    DMeclipse.TBvoluntario.Delete;

    data1.Date:=dmeclipse.TBvoluntariodtadm_vol.AsDateTime;
    data2.Date:=dmeclipse.TBvoluntariodtnasc_vol.AsDateTime;
end;

procedure Tfrmcadvol.BitBtn4Click(Sender: TObject);
begin
    botoes(self);
    mudaedit(self, false);

    DMeclipse.TBvoluntario.Cancel;

    data1.Date:=DMeclipse.TBvoluntariodtadm_vol.AsDateTime;
    data2.Date:=DMeclipse.TBvoluntariodtnasc_vol.AsDateTime;
end;

```

```

procedure Tfrmcadvol.BitBtn5Click(Sender: TObject);

begin

    trims(self);

    if (DBEdit3.text="")or(DBRadioGroup1.Value="")or(ex.Text="")or(DBEdit9.text="")or((dbedit6.text="")and(dbedit7.text=""))or(DBRadioGroup2.value="")then

        begin

            showmessage('Preencha todos os campos requeridos');

            exit;

        end;

    if (cpf(dbedit7.Text)=false)and(dbedit7.text<>") then

        begin

            showmessage('CPF Inválido');

            PageControl1.ActivePageIndex:=0;

            DBEdit7.SetFocus;

            exit;

        end;

    botoes(self);

    mudaedit(self, false);

    dmeclipse.TBvoluntariodtadm_vol.AsDateTime:=data1.Date;

    dmeclipse.TBvoluntariodtnasc_vol.AsDateTime:=data2.Date;

    DMeclipse.TBvoluntario.Post;

    DMeclipse.TBvoluntario.Refresh;

```

```

end;

procedure Tfrmcadvol.BitBtn6Click(Sender: TObject);

begin

    Close;

end;

procedure Tfrmcadvol.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Voluntario';

    String_tb.Scampocod := 'Codigo_vol';

    String_tb.Scamponome := 'Nome_vol';

    String_tb.Qrytb := DMeclipse.TBvoluntario;

    String_tb.DStb := DMeclipse.DStbvoluntario;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBvoluntariocodigo_vol.AsInteger;

    data1.Date:=DMeclipse.TBvoluntariodtadm_vol.AsDateTime;

    data2.Date:=DMeclipse.TBvoluntariodtnasc_vol.AsDateTime;

end;

procedure Tfrmcadvol.bloqletrarg(Sender: TObject; var Key: Char);

begin

    if ((key<#48)or(key>#57))and(key<>#8)and(key<>#120)and (key<>#88)then

        key:=#0;

end;

```

```

procedure Tfrmcadvol.bloqletra(Sender: TObject; var Key: Char);

begin

    if ((key<#48)or(key>#57))and(key<>#8)then

        key:=#0;

    end;

procedure Tfrmcadvol.FormCloseQuery(Sender: TObject;

    var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de voluntá-
rio?', mtconfirmation, [mbytes, mbno], 0) = mryes then

        begin

            DMeclipse.TBvoluntario.cancel;

            DMeclipse.TBfnvoluntario.cancel;

            dmeclipse.TBtpfone.cancel;

            DMeclipse.TBcarga.Cancel;

            canclose:=true;

        end

    else

        canclose:=false;

    end;

procedure Tfrmcadvol.BitBtn10Click(Sender: TObject);

begin

    if(FrmCadContato = nil)then

```

```

begin

    Application.CreateForm(TFrmCadContato, FrmCadContato);

end;

//PARAMETRO PARA CHAMADA DE CONTATOS

FrmCadContato.Tela := 'VOLUNTARIO';

FrmCadContato.ShowModal;

end;

end.

unit UCadVolCargo;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Buttons, Grids, DBGrids, Mask, DBCtrls, ExtCtrls, DB,
    ADODB;

type

    TfrmCadVol_cargo = class(TForm)

        Panel1: TPanel;

        DBLookupComboBox2: TDBLookupComboBox;

        Label43: TLabel;

        DBEdit14: TDBEdit;

        Label44: TLabel;

        Panel3: TPanel;

```

DBGrid2: TDBGrid;
GroupBox2: TGroupBox;
BitBtn13: TBitBtn;
BitBtn14: TBitBtn;
BitBtn15: TBitBtn;
DSinteresse: TDataSource;
ADOinteresse: TADOQuery;
ADOinteressecodigo_alu: TIntegerField;
ADOinteressecodigo_cur: TIntegerField;
ADOinteressedtfecha_esp: TDateTimeField;
ADOinteresseCurso: TStringField;
Label2: TLabel;
DBEdit5: TDBEdit;
Edit1: TEdit;
SpeedButton1: TSpeedButton;
DSAluno_p: TDataSource;
ADOAluno_p: TADOQuery;
ADOAluno_pcodigo_alu: TAutoIncField;
ADOAluno_pnome_alu: TWideStringField;
ADOinteresseNome: TStringField;
BitBtn30: TBitBtn;
procedure BitBtn13Click(Sender: TObject);

```

procedure SpeedButton1Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn15Click(Sender: TObject);

procedure BitBtn14Click(Sender: TObject);

procedure DBEdit5Exit(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure BitBtn30Click(Sender: TObject);

procedure ADOInteresseBeforePost(DataSet: TDataSet);

    function validar :boolean;

private

    { Private declarations }

public

    { Public declarations }

end;

var

    frmCadVol_cargo: TfrmCadVol_cargo;

implementation

uses API_F11, UDMeclipse, UPrincipal, UpesqGeral, UCadAluno, UCadInteresse;

```

```
{ $R *.dfm }
```

```
procedure TfrmCadVol_cargo.BitBtn13Click(Sender: TObject);
```

```
begin
```

```
  if(validar)then
```

```
  begin
```

```
    if(ADOinteresse.State in [dsinsert])then
```

```
    begin
```

```
      ADOinteresse.Post;
```

```
    end;
```

```
    ADOinteresse.Insert;
```

```
  end;
```

```
end;
```

```
procedure TfrmCadVol_cargo.SpeedButton1Click(Sender: TObject);
```

```
begin
```

```
  String_tb.Stabela := 'Aluno';
```

```
  String_tb.Scampocod := 'codigo_alu';
```

```
  String_tb.Scamponome := 'Nome_alu';
```

```
  String_tb.Qrytb := ADOAluno_p;
```

```
  String_tb.DStb := DSAAluno_p;
```

```
  abrirpesq(self);
```

```
  if not(ADOAluno_p.IsEmpty)then
```

```
  begin
```



```

ADOInteresse.Close;

ADOInteresse.Parameters.ParamByName('codialu').Value      :=      ADOAlu-
no_pcodigo_alu.AsInteger;

ADOInteresse.Open;

ADOInteresse.Insert;

DBEdit5.Text := ADOAluno_pcodigo_alu.AsVariant;

Edit1.Text := ADOAluno_pnome_alu.AsString;

end

else

    ShowMessage('Aluno não cadastrado');

end;

procedure TfrmCadVol_cargo.FormClose(Sender: TObject;

var Action: TCloseAction);

begin

    ADOInteresse.Active:=false;

    ADOAluno_p.Active:=false;

    // Application.CreateForm(Tfrmprincipal,frmprincipal);

    // frmprincipal.show;

    action:= cafree;

    frmCad_Interesse:= nil;

end;

procedure TfrmCadVol_cargo.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

```

```
begin
```

```
  If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de Interesses?', mtconfirmation, [mbytes, mbno], 0) = mryes then
```

```
  begin
```

```
    ADOinteresse.cancel;
```

```
    candclose:=true;
```

```
  end
```

```
  else
```

```
    candclose:=false;
```

```
end;
```

```
procedure TfrmCadVol_cargo.FormKeyPress(Sender: TObject; var Key: Char);
```

```
begin
```

```
  if(Key = #13)then
```

```
  begin
```

```
    Key := #13;
```

```
    SelectNext(ActiveControl,true,true);
```

```
  end;
```

```
end;
```

```
procedure TfrmCadVol_cargo.BitBtn15Click(Sender: TObject);
```

```
begin
```

```
  ADOinteresse.Edit;
```

```
end;
```

```
procedure TfrmCadVol_cargo.BitBtn14Click(Sender: TObject);
```

```

begin

    if not(ADOinteresse.IsEmpty)then

        ADOinteresse.Delete;

    if(DBEdit5.Text = "")then

        Edit1.Clear;

end;

procedure TfrmCadVol_cargo.DBEdit5Exit(Sender: TObject);

begin

    ADOAluno_p.Close;

    ADOAluno_p.SQL.Clear;

    ADOAluno_p.SQL.Add('select codigo_alu, nome_alu from tb_aluno ');

    ADOAluno_p.SQL.Add(' where codigo_alu = :codi ');

    ADOAluno_p.Parameters.ParamByName('codi').Value := DBEdit5.Text;

    ADOAluno_p.Open;

    if not(ADOAluno_p.IsEmpty)then

        begin

            ADOinteresse.Close;

            ADOinteresse.Parameters.ParamByName('codialu').Value := ADOAluno_p.codigo_alu.AsInteger;

            ADOinteresse.Open;

            ADOinteresse.Insert;

            DBEdit5.Text := ADOAluno_p.codigo_alu.AsVariant;

            Edit1.Text := ADOAluno_p.nome_alu.AsString;

```

```

end

else

    ShowMessage('Aluno não cadastrado');

end;

procedure TfrmCadVol_cargo.FormShow(Sender: TObject);

begin

    Panel3.Enabled := false;

    frmCad_Interesse.ADOAluno_p.Active := true;

{ frmCad_Interesse.Edit1.Enabled := true;

    frmCad_Interesse.DBEdit5.Enabled := true;

    frmCad_Interesse.SpeedButton1.Enabled := true;  }

    frmCad_Interesse.ADOinteresse.Active := true;

    frmCad_Interesse.ADOinteresse.Insert;

end;

procedure TfrmCadVol_cargo.BitBtn30Click(Sender: TObject);

begin

    Panel3.Enabled := not Panel3.Enabled;

    ADOinteresse.Cancel;

end;

procedure TfrmCadVol_cargo.ADOinteresseBeforePost(DataSet: TDataSet);

begin

    //validar;

```

```

if(validar = false)then

begin

    ADOinteresse.Cancel;

end;

end;


function TfrmCadVol_cargo.validar: boolean;

var

    valida : boolean;

begin

    valida := true;

    if(trim(DBEdit5.Text) = '')then

        valida := false;

    if(trim(DBEdit14.Text) = '')then

        valida := false;

    if(trim(DBLookupComboBox2.Text) = '')then

        valida := false;

    if(valida = false)then

begin

        ShowMessage('Existem campos necessarios em branco');

        result := false;

end

```

```

else

    Result := true;

end;

end.

unit UCadCargo;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, DBCtrls, StdCtrls, Mask, ExtCtrls, Buttons;

type

    TFrmCargo = class(TForm)

        Panel1: TPanel;

        Label1: TLabel;

        Label2: TLabel;

        DBText1: TDBText;

        DBEdit2: TDBEdit;

        Panel2: TPanel;

        BitBtn1: TBitBtn;

        BitBtn2: TBitBtn;

        BitBtn3: TBitBtn;

        BitBtn4: TBitBtn;

```

```

    BitBtn5: TBitBtn;

    BitBtn6: TBitBtn;

    BitBtn7: TBitBtn;

    procedure FormClose(Sender: TObject; var Action: TCloseAction);

    procedure FormKeyPress(Sender: TObject; var Key: Char);

    procedure BitBtn1Click(Sender: TObject);

    procedure BitBtn2Click(Sender: TObject);

    procedure BitBtn3Click(Sender: TObject);

    procedure BitBtn4Click(Sender: TObject);

    procedure BitBtn5Click(Sender: TObject);

    procedure BitBtn7Click(Sender: TObject);

    procedure BitBtn6Click(Sender: TObject);

private
    { Private declarations }

public
    { Public declarations }

end;

var
    FrmCargo: TFrmCargo;

implementation

uses UDMeclipse, API_F11, UpesqGeral;

```

```
{ $R *.dfm }
```

```
procedure TFrmCargo.FormClose(Sender: TObject; var Action: TCloseAction);
```

```
begin
```

```
    action:= cafree;
```

```
    FrmCargo:= nil;
```

```
end;
```

```
procedure TFrmCargo.FormKeyPress(Sender: TObject; var Key: Char);
```

```
begin
```

```
    if(Key = #13)then
```

```
    begin
```

```
        Key := #13;
```

```
        SelectNext(ActiveControl,true,true);
```

```
    end;
```

```
end;
```

```
procedure TFrmCargo.BitBtn1Click(Sender: TObject);
```

```
begin
```

```
    botoes(self);
```

```
    mudaedit(self, true);
```

```
    DMeclipse.TBcargo.Insert;
```

```
end;
```

```
procedure TFrmCargo.BitBtn2Click(Sender: TObject);
```

```
begin
```



```

if not(DMeclipse.TBcargo.IsEmpty)then

begin

    botoes(self);

    mudaedit(self, true);

    DMeclipse.TBcargo.Edit;

end

else

    ShowMessage('Não existe registro para edição.');
```

```

end;

procedure TFrmCargo.BitBtn3Click(Sender: TObject);

begin

    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBcargo.delete;

end;

procedure TFrmCargo.BitBtn4Click(Sender: TObject);

begin

    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBcargo.Cancel;

end;

procedure TFrmCargo.BitBtn5Click(Sender: TObject);
```

```

begin

    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBcargo.Post;

    DMeclipse.TBcargo.Refresh;

end;

procedure TFrmCargo.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Cargo';

    String_tb.Scampocod := 'Codigo_vol';

    String_tb.Scamponome := 'Nome_vol';

    String_tb.Qrytb := DMeclipse.TBcargo;

    String_tb.DStb := DMeclipse.DStbcargo;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBcargocodigo_vol.AsInteger;

end;

procedure TFrmCargo.BitBtn6Click(Sender: TObject);

begin

    close;

end;

end.

unit UCadAssociado;

```

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, ExtCtrls, DBCtrls, Mask, ImgList, ComCtrls, ToolWin,
Buttons, DB, ADODB, ppModule, raCodMod, ppBands, ppCtrls, ppPrnabl,
ppClass, ppVar, ppCache, ppProd, ppReport, ppDB, ppComm, ppRelatv,
ppDBPipe;

type

TFrmAssociado = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

DBEdit3: TDBEdit;

Label5: TLabel;

Label6: TLabel;

DBEdit6: TDBEdit;

Label7: TLabel;

DBEdit7: TDBEdit;

Label8: TLabel;

DBEdit8: TDBEdit;

Label9: TLabel;

DBEdit9: TDBEdit;

Label10: TLabel;
DBEdit10: TDBEdit;
Label11: TLabel;
DBEdit11: TDBEdit;
Label12: TLabel;
DBEdit12: TDBEdit;
Label13: TLabel;
DBEdit13: TDBEdit;
Label14: TLabel;
DBEdit14: TDBEdit;
Label15: TLabel;
DBEdit15: TDBEdit;
Label17: TLabel;
Label18: TLabel;
DBEdit18: TDBEdit;
Label19: TLabel;
DBEdit19: TDBEdit;
Label20: TLabel;
DBEdit20: TDBEdit;
Label21: TLabel;
DBEdit21: TDBEdit;
Label22: TLabel;

DBEdit22: TDBEdit;
Label23: TLabel;
DBEdit23: TDBEdit;
DBText1: TDBText;
DBRadioGroup1: TDBRadioGroup;
DBEdit17: TDBEdit;
Panel1: TPanel;
lbluf: TLabel;
Panel2: TPanel;
BitBtn1: TBitBtn;
BitBtn2: TBitBtn;
BitBtn3: TBitBtn;
BitBtn4: TBitBtn;
BitBtn5: TBitBtn;
BitBtn6: TBitBtn;
BitBtn7: TBitBtn;
DBComboBox1: TDBComboBox;
data1: TDateTimePicker;
data2: TDateTimePicker;
BitBtn10: TBitBtn;
BitBtn8: TBitBtn;
ppDBPipeline1: TppDBPipeline;

ppDBPipeline1ppField1: TppField;
ppDBPipeline1ppField2: TppField;
ppDBPipeline1ppField3: TppField;
ppDBPipeline1ppField4: TppField;
ppDBPipeline1ppField5: TppField;
ppDBPipeline1ppField6: TppField;
ppDBPipeline1ppField7: TppField;
ppDBPipeline1ppField8: TppField;
ppDBPipeline1ppField9: TppField;
ppDBPipeline1ppField10: TppField;
ppDBPipeline1ppField11: TppField;
ppDBPipeline1ppField12: TppField;
ppDBPipeline1ppField13: TppField;
ppDBPipeline1ppField14: TppField;
ppDBPipeline1ppField15: TppField;
ppDBPipeline1ppField16: TppField;
ppDBPipeline1ppField17: TppField;
ppDBPipeline1ppField18: TppField;
ppDBPipeline1ppField19: TppField;
ppDBPipeline1ppField20: TppField;
ppDBPipeline1ppField21: TppField;
ppDBPipeline1ppField22: TppField;

ppDBPipeline1ppField23: TppField;
ppReport1: TppReport;
ppHeaderBand1: TppHeaderBand;
ppSystemVariable1: TppSystemVariable;
ppLabel34: TppLabel;
ppLine4: TppLine;
ppLine5: TppLine;
ppLabel35: TppLabel;
ppLabel36: TppLabel;
ppLabel37: TppLabel;
ppDetailBand1: TppDetailBand;
ppLabel1: TppLabel;
ppDBText1: TppDBText;
ppLabel2: TppLabel;
ppDBText2: TppDBText;
ppLabel3: TppLabel;
ppDBText3: TppDBText;
ppLabel4: TppLabel;
ppDBText4: TppDBText;
ppLabel5: TppLabel;
ppDBText5: TppDBText;
ppLabel6: TppLabel;

ppDBText6: TppDBText;
ppLabel7: TppLabel;
ppDBText7: TppDBText;
ppLabel8: TppLabel;
ppDBText8: TppDBText;
ppLabel9: TppLabel;
ppDBText9: TppDBText;
ppLabel10: TppLabel;
ppDBText10: TppDBText;
ppLabel11: TppLabel;
ppDBText11: TppDBText;
ppLabel12: TppLabel;
ppDBText12: TppDBText;
ppLabel13: TppLabel;
ppDBText13: TppDBText;
ppLabel14: TppLabel;
ppDBText14: TppDBText;
ppLabel15: TppLabel;
ppDBText15: TppDBText;
ppLabel16: TppLabel;
ppDBText16: TppDBText;
ppLabel17: TppLabel;

ppDBText17: TppDBText;

ppLabel18: TppLabel;

ppDBText18: TppDBText;

ppLabel19: TppLabel;

ppDBText19: TppDBText;

ppLabel20: TppLabel;

ppDBText20: TppDBText;

ppLabel21: TppLabel;

ppDBText21: TppDBText;

ppLabel22: TppLabel;

ppDBText22: TppDBText;

ppLabel23: TppLabel;

ppDBText23: TppDBText;

ppDBText24: TppDBText;

ppLabel24: TppLabel;

ppLabel25: TppLabel;

ppLabel26: TppLabel;

ppLabel27: TppLabel;

ppLabel28: TppLabel;

ppLabel29: TppLabel;

ppLabel30: TppLabel;

ppLabel31: TppLabel;

```
ppLabel32: TppLabel;  
  
ppLabel33: TppLabel;  
  
ppLabel39: TppLabel;  
  
ppLabel40: TppLabel;  
  
ppLabel41: TppLabel;  
  
ppLabel42: TppLabel;  
  
ppLabel43: TppLabel;  
  
ppLabel44: TppLabel;  
  
ppLabel45: TppLabel;  
  
ppLine1: TppLine;  
  
ppLine2: TppLine;  
  
ppFooterBand1: TppFooterBand;  
  
ppLine6: TppLine;  
  
ppLabel38: TppLabel;  
  
raCodeModule1: TraCodeModule;  
  
ppDBPipeline2: TppDBPipeline;  
  
ppDBPipeline2ppField1: TppField;  
  
ppDBPipeline2ppField2: TppField;  
  
ppDBPipeline2ppField3: TppField;  
  
ADOQuery1: TADOQuery;  
  
DataSource1: TDataSource;  
  
procedure FormClose(Sender: TObject; var Action: TCloseAction);
```

```

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure bloqletra(Sender: TObject; var Key: Char);

procedure bloqletrarg(Sender: TObject; var Key: Char);

procedure FormCreate(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure BitBtn10Click(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmAssociado: TFrmAssociado;

implementation

uses UDMeclipse, API_F11, UpesqGeral, UPrincipal, UPrincipal2, UCadContato;

```

```
{ $R *.dfm }
```

```
procedure TFrmAssociado.FormClose(Sender: TObject;  
    var Action: TCloseAction);  
  
begin  
    DMeclipse.TBassociado.Active:=false;  
    dmeclipse.TBfnassociado.Active:=false;  
    dmeclipse.TBtpfone.active:=false;  
  
    if menup='1900' then  
    begin  
        FrmPrincipal2.Enabled := true;  
    end  
    else  
    begin  
        If(FrmPrincipal = nil)then  
        begin  
            application.CreateForm(TFrmPrincipal,FrmPrincipal);  
            FrmPrincipal.Show;  
        end;  
        FrmPrincipal.Enabled := true;  
    end;  
  
    action:= cafree;
```

```

FrmAssociado:= nil;

end;

procedure TFrmAssociado.FormKeyPress(Sender: TObject; var Key: Char);
begin
    if(Key = #13)then
        begin
            Key := #13;

            SelectNext(ActiveControl,true,true);
        end;
    end;

end;

procedure TFrmAssociado.BitBtn2Click(Sender: TObject);
begin
    if not(DMeclipse.TBassociado.IsEmpty)then
        begin
            botoes(self);

            mudaedit(self, true);

            DMeclipse.TBassociado.Edit;
        end
    else
        ShowMessage('Não existe registro para edição.');
```

end;

procedure TFrmAssociado.BitBtn1Click(Sender: TObject);

begin

botoes(self);

mudaedit(self, true);

DMeclipse.TBassociado.Insert;

end;

procedure TFrmAssociado.BitBtn3Click(Sender: TObject);

begin

if dmeclipse.TBassociado.isempty then

begin

showmessage('Não existe registro a ser excluído');

exit;

end;

If messagedlg('Tem certeza que deseja excluir esse aluno?'+#13+'Essa operação é irreversível!', mtconfirmation, [mbytes, mbno], 0) = mrno then

exit;

DMeclipse.TBassociado.Delete;

data1.Date:=dmeclipse.TBassociadodtadm_ass.AsDateTime;

data2.Date:=DMeclipse.TBassociadodtnasc_ass.AsDateTime;

end;

```

procedure TFrmAssociado.BitBtn4Click(Sender: TObject);

begin

    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBassociado.Cancel;

    data1.Date:=dmeclipse.TBassociadodtadm_ass.AsDateTime;

    data2.Date:=DMeclipse.TBassociadodtnasc_ass.AsDateTime;

end;

```

```

procedure TFrmAssociado.BitBtn5Click(Sender: TObject);

begin

    trims(self);

    if
        (trim(DBEdit3.text='')or(DBRadioGroup1.value='')or((DBEdit6.Text='')and(dbedit7.tex
t=''))or(trim(dbedit8.text='')or(trim(dbedit9.text='')or(DBEdit10.text='')or(dbedit11.text
='')or(dbedit13.text='')or(dbedit15.text='')Or(DBComboBox1.text='') then

        begin

            showmessage('Preencha todos os campos requeridos');

            exit;

        end;

    if (cpf(dbedit7.Text)=false)and(dbedit7.text<>') then

        begin

```

```

    showmessage('CPF Inválido');

    DBEdit7.SetFocus;

    exit;

end;

botoes(self);

mudaedit(self, false);

DMeclipse.TBassociadodtadm_ass.AsDateTime:=data1.Date;

DMeclipse.TBassociadodtnasc_ass.AsDateTime:=data2.Date;

DMeclipse.TBassociado.Post;

DMeclipse.TBassociado.Refresh;

end;

procedure TFrmAssociado.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Associado';

    String_tb.Scampocod := 'Codigo_ass';

    String_tb.Scamponome := 'Nome_ass';

    String_tb.Qrytb := DMeclipse.TBassociado;

    String_tb.DStb := DMeclipse.DStbassociado;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBassociadocodigo_ass.AsInteger;

    data1.Date:=dmeclipse.TBassociadodtadm_ass.AsDateTime;

```



```
data2.Date:=DMeclipse.TBassociadodtnasc_ass.AsDateTime;  
  
end;
```

```
procedure TFrmAssociado.BitBtn6Click(Sender: TObject);  
  
begin  
  
    Close;  
  
end;
```

```
procedure TFrmAssociado.bloqletra(Sender: TObject; var Key: Char);  
  
begin  
  
    if ((key<#48)or(key>#57))and(key<>#8)then  
  
        key:=#0;  
  
end;
```

```
procedure TFrmAssociado.bloqletrarg(Sender: TObject; var Key: Char);  
  
begin  
  
    if ((key<#48)or(key>#57))and(key<>#8)and(key<>#120)and (key<>#88)then  
  
        key:=#0;  
  
end;
```

```
procedure TFrmAssociado.FormCreate(Sender: TObject);  
  
begin
```

```

DMeclipse.TBassociado.Active:=true;

dmeclipse.TBfnassociado.Active:=true;

dmeclipse.TBtpfone.active:=true;

data1.Date:=dmeclipse.TBassociadodtadm_ass.AsDateTime;

data2.Date:=DMeclipse.TBassociadodtnasc_ass.AsDateTime;

end;


procedure TFrmAssociado.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de associa-
do?', mtconfirmation, [mbytes, mbno], 0) = mryes then

begin

DMeclipse.TBassociado.cancel;

DMeclipse.TBfnassociado.cancel;

dmeclipse.TBtpfone.cancel;

canclose:=true;

end

else

canclose:=false;

end;


procedure TFrmAssociado.BitBtn10Click(Sender: TObject);

```

```

begin

    if(FrmCadContato = nil)then

        begin

            Application.CreateForm(TFrmCadContato, FrmCadContato);

        end;

        //PARAMETRO PARA CHAMADA DE CONTATOS

        FrmCadContato.Tela := 'ASSOCIADO';

        FrmCadContato.ShowModal;

    end;

end.

unit UCadCurso;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, DBCtrls, StdCtrls, Mask, ExtCtrls, Buttons;

type

    TFrmCurso = class(TForm)

        Label1: TLabel;

```

```
Label2: TLabel;  
  
DBEdit2: TDBEdit;  
  
DBText1: TDBText;  
  
Panel1: TPanel;  
  
Panel2: TPanel;  
  
BitBtn1: TBitBtn;  
  
BitBtn2: TBitBtn;  
  
BitBtn3: TBitBtn;  
  
BitBtn4: TBitBtn;  
  
BitBtn5: TBitBtn;  
  
BitBtn6: TBitBtn;  
  
BitBtn7: TBitBtn;  
  
procedure FormClose(Sender: TObject; var Action: TCloseAction);  
  
procedure FormKeyPress(Sender: TObject; var Key: Char);  
  
procedure BitBtn1Click(Sender: TObject);  
  
procedure BitBtn2Click(Sender: TObject);  
  
procedure BitBtn3Click(Sender: TObject);  
  
procedure BitBtn4Click(Sender: TObject);  
  
procedure BitBtn5Click(Sender: TObject);  
  
procedure BitBtn6Click(Sender: TObject);  
  
procedure BitBtn7Click(Sender: TObject);  
  
procedure FormCreate(Sender: TObject);
```

```

    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private
    { Private declarations }

public
    { Public declarations }

end;

var
    FrmCurso: TFrmCurso;

implementation

uses UDMeclipse, API_F11, UpesqGeral, UPrincipal, UPrincipal2;

{$R *.dfm}

procedure TFrmCurso.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    DMeclipse.TBcurso.Active:=false;

    if menup='1900' then
    begin
        frmprincipal2.Enabled:=true;
    end;
end;

```

```

end

else

begin

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

action:= cafree;

FrmCurso:= nil;

end;


procedure TFrmCurso.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

end;


procedure TFrmCurso.BitBtn1Click(Sender: TObject);

begin

    botoes(self);

```

```

    mudaedit(self, true);

    DMeclipse.TBcurso.Insert;

end;


procedure TFrmCurso.BitBtn2Click(Sender: TObject);

begin

    if not(DMeclipse.TBcurso.IsEmpty)then

        begin

            botoes(self);

            mudaedit(self, true);

            DMeclipse.TBcurso.Edit;

        end

    else

        ShowMessage('Não existe registro para edição.');
```

```

end;


procedure TFrmCurso.BitBtn3Click(Sender: TObject);

begin

    if dmeclipse.TBcurso.isempty then

        begin

            showmessage('Não existe registro a ser excluído');
```

```

        exit;
```

```
end;
```

```
If messagedlg('Tem certeza que deseja excluir esse curso?'+#13+'Essa operação é  
irreversível', mtconfirmation, [mbytes, mbno], 0) = mrno then
```

```
    exit;
```

```
    DMeclipse.TBcurso.Delete;
```

```
end;
```

```
procedure TFrmCurso.BitBtn4Click(Sender: TObject);
```

```
begin
```

```
    botoes(self);
```

```
    mudaedit(self, false);
```

```
    DMeclipse.TBcurso.Cancel;
```

```
end;
```

```
procedure TFrmCurso.BitBtn5Click(Sender: TObject);
```

```
begin
```

```
    trims(self);
```

```
    if DBEdit2.Text="" then
```

```
    begin
```

```
        showmessage('Preencha o nome do curso');
```

```
        DBEdit2.SetFocus;
```

```
    exit;
```

```
end;
```



```

botoes(self);

mudaedit(self, false);

DMeclipse.TBcurso.Post;

DMeclipse.TBcurso.Refresh;

end;


procedure TFrmCurso.BitBtn6Click(Sender: TObject);

begin

    close;

end;


procedure TFrmCurso.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Curso';

    String_tb.Scampocod := 'Codigo_cur';

    String_tb.Scamponome := 'Nome_cur';

    String_tb.Qrytb := DMeclipse.TBcurso;

    String_tb.DStb := DMeclipse.DStbcurso;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBcursocodigo_cur.AsInteger;

end;

```

```
procedure TFrmCurso.FormCreate(Sender: TObject);
```

```
begin
```

```
    DMeclipse.TBcurso.Active:=true;
```

```
end;
```

```
procedure TFrmCurso.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
```

```
begin
```

```
    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de curso?',  
mtconfirmation, [mbyes, mbno], 0) = mryes then
```

```
begin
```

```
    DMeclipse.TBcurso.cancel;
```

```
    canclose:=true;
```

```
end
```

```
else
```

```
    canclose:=false;
```

```
end;
```

```
end.
```

```
unit UpesqGeral;
```

```
interface
```

```
uses
```

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DB, ADODB, Grids, DBGrids, StdCtrls, ExtCtrls;

type

TFrmmpesq = class(TForm)

Panel1: TPanel;

Panel2: TPanel;

Edit1: TEdit;

ComboBox1: TComboBox;

Button1: TButton;

DBGrid1: TDBGrid;

procedure DBGrid1DbClick(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Edit1Exit(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCreate(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

private

{ Private declarations }

public

{ Public declarations }

end;

type

TString_tb = class

Stabela : String;

Scampocod : String;

Scamponome : String;

CField : TAutoIncField;

Qrytb : TADOQuery;

DStb : TDataSource;

resultcodi : integer;

end;

var

Frmpesq: TFrmpesq;

String_tb : TString_tb;

implementation

uses UDMeclipse, UCadLocador, StrUtils, API_F11;

{ \$R *.dfm }

```
procedure TFrmpesq.DBGrid1DbClick(Sender: TObject);
```

```
begin
```

```
    Button1.Click;
```

```
end;
```

```
procedure TFrmpesq.Button1Click(Sender: TObject);
```

```
begin
```

```
    CLOSE;
```

```
end;
```

```
procedure TFrmpesq.Edit1Exit(Sender: TObject);
```

```
begin
```

```
    pesq(Self,    String_tb.Qrytb,    String_tb.Stabela,    String_tb.Scamponome,  
    String_tb.Scampocod);
```

```
end;
```

```
procedure TFrmpesq.FormClose(Sender: TObject; var Action: TCloseAction);
```

```
begin
```

```
    action:= cafree;
```

```
    Frmpesq:= nil;
```

```
end;
```

```

procedure TFrmmpesq.FormCreate(Sender: TObject);

begin

// String_tb := TString_tb.Create;

    DBGrid1.DataSource := String_tb.DStb;

end;


procedure TFrmmpesq.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;


end.

unit UCadAluno;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

```

Dialogs, DBCtrls, StdCtrls, ExtCtrls, Mask, ComCtrls, DB, ToolWin,
Buttons, Grids, DBGrids, ADODB, DBClient;

type

TFrmAluno = class(TForm)

 TabSheet1: TTabSheet;

 TabSheet2: TTabSheet;

 TabSheet3: TTabSheet;

 Label1: TLabel;

 DBText1: TDBText;

 Label2: TLabel;

 DBEdit2: TDBEdit;

 Label4: TLabel;

 Label5: TLabel;

 Label6: TLabel;

 DBEdit6: TDBEdit;

 Label7: TLabel;

 DBEdit7: TDBEdit;

 Label8: TLabel;

 DBEdit8: TDBEdit;

 Label9: TLabel;

 DBEdit9: TDBEdit;

Label10: TLabel;
DBEdit10: TDBEdit;
Label11: TLabel;
DBEdit11: TDBEdit;
DBEdit12: TDBEdit;
Label13: TLabel;
DBEdit13: TDBEdit;
Label14: TLabel;
DBRGsexo: TDBRadioGroup;
DBEdit3: TDBEdit;
DBComboBox1: TDBComboBox;
Label12: TLabel;
Label15: TLabel;
DBEdit15: TDBEdit;
Label16: TLabel;
DBEdit16: TDBEdit;
Label17: TLabel;
DBEdit17: TDBEdit;
Label18: TLabel;
DBEdit18: TDBEdit;
Label19: TLabel;
DBEdit19: TDBEdit;

Label20: TLabel;
DBEdit20: TDBEdit;
Label21: TLabel;
DBEdit21: TDBEdit;
Label22: TLabel;
DBEdit22: TDBEdit;
Label23: TLabel;
DBEdit23: TDBEdit;
Label25: TLabel;
DBEdit25: TDBEdit;
Label26: TLabel;
DBEdit26: TDBEdit;
Label27: TLabel;
DBEdit27: TDBEdit;
Label29: TLabel;
DBEdit29: TDBEdit;
Label30: TLabel;
DBEdit30: TDBEdit;
Label31: TLabel;
DBEdit31: TDBEdit;
Label32: TLabel;
DBEdit32: TDBEdit;

Label33: TLabel;
DBEdit33: TDBEdit;
Label34: TLabel;
DBEdit34: TDBEdit;
Label35: TLabel;
DBEdit35: TDBEdit;
Label36: TLabel;
Label37: TLabel;
DBComboBox2: TDBComboBox;
Label38: TLabel;
DBEdit38: TDBEdit;
DBImage1: TDBImage;
Label39: TLabel;
pagecontrol_alu: TPageControl;
Panel1: TPanel;
BitBtn1: TBitBtn;
BitBtn2: TBitBtn;
BitBtn3: TBitBtn;
BitBtn4: TBitBtn;
BitBtn5: TBitBtn;
BitBtn6: TBitBtn;
BitBtn7: TBitBtn;

```

Label3: TLabel;

DBEdit4: TDBEdit;

ADOQuery2: TADOQuery;

ADOQuery2codigo_alu: TIntegerField;

ADOQuery2tpitel_fnal: TWideStringField;

ADOQuery2fone_fnal: TWideStringField;

SpeedButton1: TSpeedButton;

DBRadioGroup1: TDBRadioGroup;

DBRadioGroup2: TDBRadioGroup;

Button1: TButton;

teste: TDataSource;

BitBtn10: TBitBtn;

data: TDateTimePicker;

Data2: TDateTimePicker;

DBRadioGroup3: TDBRadioGroup;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

```

```

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure BitBtn10Click(Sender: TObject);

procedure SpeedButton1Click(Sender: TObject);

procedure bloqletra(Sender: TObject; var Key: Char);

procedure bloqletrarg(Sender: TObject; var Key: Char);

procedure FormCreate(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmAluno: TFrmAluno;

    codi : integer;

implementation

uses UDMeclipse, UPrincipal, API_F11, UpesqGeral, Ufoto, UCadInteresse,

```

UCadContato, UPrincipal2;

{ \$R *.dfm }

procedure TFrmAluno.FormClose(Sender: TObject; var Action: TCloseAction);

begin

 DMeclipse.TBaluno.Active:=false;

 DMeclipse.TBfnaluno.Active:=false;

 dmeclipse.TBtpfone.Active:=false;

 dmeclipse.TBlistaespera.Active:=false;

 if menup='1900' then

 begin

 frmprincipal2.Enabled:=true;

 end

 else

 begin

 application.CreateForm(TFrmPrincipal,FrmPrincipal);

 FrmPrincipal.Show;

 end;

 Action:= caFree;

 FrmAluno:= nil;

end;

```

procedure TFrmAluno.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de aluno?',
mtconfirmation, [mbytes, mbno], 0) = mryes then

        begin

            //  ADOQuery1.Cancel;

            DMeclipse.TBaluno.cancel;

            DMeclipse.TBfnaluno.cancel;

            dmeclipse.TBtpfone.cancel;

            dmeclipse.TBlistaespera.cancel;

            canclose:=true;

        end

    else

        canclose:=false;

    end;

procedure TFrmAluno.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

end;

```

```

procedure TFrmAluno.BitBtn1Click(Sender: TObject);

begin

    botoes(self);

    mudaedit(self, true);

    DMeclipse.TBaluno.Insert;

    DMeclipse.TBalunodtcad_alu.AsString:=datetostr(date);

    DMeclipse.TBalunoStatus.AsString:='L';

    pagecontrol_alu.ActivePageIndex:=0;

    dbedit2.setfocus;

end;

procedure TFrmAluno.BitBtn2Click(Sender: TObject);

begin

    if not(DMeclipse.TBaluno.IsEmpty)then

        begin

            botoes(self);

            mudaedit(self, true);

            DMeclipse.TBaluno.Edit;

        end

    else

        begin

            ShowMessage('Não existe registro para edição.');
```

```

        exit;

    end;

end;

```

```

end;

end;

procedure TFrmAluno.BitBtn3Click(Sender: TObject);

begin
    if dmeclipse.tbaluno.isempty then
        begin
            showmessage('Não existe registro a ser excluído');

            exit;
        end;

    If messagedlg('Tem certeza que deseja excluir esse aluno?'+#13+'Essa operação é
    irreversível', mtconfirmation, [mbytes, mbno], 0) = mrno then

        exit;

    DMeclipse.TBaluno.Delete;

    data.Date:=DMeclipse.TBalunodtnasc_alu.AsDateTime;

    data2.Date:=dmeclipse.TBalunodtcad_alu.AsDateTime;

end;

procedure TFrmAluno.BitBtn4Click(Sender: TObject);

begin
    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBaluno.Cancel;

    data.Date:=DMeclipse.TBalunodtnasc_alu.AsDateTime;

    data2.Date:=dmeclipse.TBalunodtcad_alu.AsDateTime;

```


end;

procedure TFrmAluno.BitBtn5Click(Sender: TObject);

begin

 trims(self);

 if

 (trim(dbedit2.text='')or(DBRGsexo.Value='')or(DBEdit3.text='')or(trim(dbedit6.text='')
 or(trim(dbedit7.text='')or(trim(DBEdit8.Text='')or(trim(DBEdit9.Text='') or
 (trim(dbedit11.text='') or(Trim(dbedit13.text='') or(DBComboBox1.text='')
 or(trim(dbedit15.text='')or(trim(DBEdit19.Text='')or((dbedit17.text='')and(dbedit18.te
 xt='') and (dbedit21.text='')and(dbedit22.text='')) then

 begin

 showmessage('Preencha todos os campos requeridos');

 exit;

end;

if (cpf(dbedit18.Text)=false)and(dbedit18.text<>') then

begin

 showmessage('CPF Inválido');

 pagecontrol_alu.ActivePageIndex:=1;

 DBEdit18.SetFocus;

 exit;

end;

if (cpf(dbedit22.Text)=false)and(dbedit22.text<>') then

begin

```

showmessage('CPF Inválido');

pagecontrol_alu.ActivePageIndex:=1;

DBEdit22.SetFocus;

exit;

end;

botoes(self);

mudaedit(self, false);

DMeclipse.TBalunodtnasc_alu.AsDateTime:=data.Date;

dmeclipse.TBalunodtcad_alu.AsDateTime:=data2.Date;

DMeclipse.TBaluno.Post;


//tel

if(DMeclipse.TBaluno.State in [dsinsert])then

begin

    if(FrmCadContato = nil)then

begin

    Application.CreateForm(TFrmCadContato, FrmCadContato);

    FrmCadContato.Tela := 'ALUNO';

    FrmCadContato.ShowModal;

end;

end

else

```

```

begin

    If messagedlg('Deseja ir as formulário de cadastro de Contatos?', mtconfirmation,
[mbytes, mbno], 0) = mryes then

        begin

            if(frmCad_Interesse = nil)then

                Application.CreateForm(TFrmCadContato, FrmCadContato);

                FrmCadContato.Tela := 'ALUNO';

                FrmCadContato.ShowModal;

            end;

        end;

//interesse

if(DMeclipse.TBaluno.State in [dsinsert])then

begin

    if(frmCad_Interesse = nil)then

        begin

            Application.CreateForm(TfrmCad_Interesse, frmCad_Interesse);

            frmCad_Interesse.ShowModal;

        end;

    end

else

begin

    If messagedlg('Deseja ir as formulário de cadastro de Interesses?', mtconfirmation,
[mbytes, mbno], 0) = mryes then

```

```

begin

    if(frmCad_Interesse = nil)then

        Application.CreateForm(TfrmCad_Interesse, frmCad_Interesse);

        frmCad_Interesse.ShowModal;

    end;

end;

//geral

DMeclipse.ADOCBD.BeginTrans;

try

    DMeclipse.ADOCBD.CommitTrans;

except

    DMeclipse.ADOCBD.RollbackTrans;

end;

end;

procedure TFrmAluno.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Aluno';

    String_tb.Scampocod := 'Codigo_alu';

    String_tb.Scamponome := 'Nome_alu';

    String_tb.Qrytb := DMeclipse.TBaluno;

    String_tb.DStb := DMeclipse.DStbaluno;

    abrirpesq(self);

```

```

data.Date:=DMeclipse.TBalunodtnasc_alu.AsDateTime;

data2.Date:=dmeclipse.TBalunodtcad_alu.AsDateTime;

end;

procedure TFrmAluno.BitBtn10Click(Sender: TObject);

begin

    if(FrmCadContato = nil)then

        begin

            Application.CreateForm(TFrmCadContato, FrmCadContato);

        end;

        //PARAMETRO PARA CHAMADA DE CONTATOS

        FrmCadContato.Tela := 'ALUNO';

        FrmCadContato.ShowModal;

    end;

procedure TFrmAluno.SpeedButton1Click(Sender: TObject);

begin

    If(foto = nil)then

        begin

            Application.CreateForm(Tfoto, foto);

            foto.Showmodal;

        end;

    end;

end;

procedure TFrmAluno.bloqletra(Sender: TObject; var Key: Char);

```

```

begin

    if ((key<#48)or(key>#57))and(key<>#8)then

        key:=#0;

    end;

procedure TFrmAluno.bloqletrarg(Sender: TObject; var Key: Char);

begin

    if ((key<#48)or(key>#57))and(key<>#8)and(key<>#120)and (key<>#88)then

        key:=#0;

    end;

procedure TFrmAluno.FormCreate(Sender: TObject);

begin

    DMeclipse.TBaluno.Active:=true;

    DMeclipse.TBfnaluno.Active:=true;

    dmeclipse.TBtpfone.Active:=true;

    dmeclipse.TBlistaespera.Active:=true;

    data.Date:=DMeclipse.TBalunodtnasc_alu.AsDateTime;

    data2.date:=dmeclipse.TBalunodtcad_alu.AsDateTime;

end;

procedure TFrmAluno.BitBtn6Click(Sender: TObject);

begin

    close;

end;

```

```

procedure TFrmAluno.Button1Click(Sender: TObject);

begin
    if(frmCad_Interesse = nil)then
        begin
            Application.CreateForm(TfrmCad_Interesse, frmCad_Interesse);

            frmCad_Interesse.ShowModal;

        end;
    end;

end.

unit Ufoto;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, Buttons, ExtCtrls, JLCVideo;

type

    TFoto = class(TForm)

        JLCVideo1: TJLCVideo;

        SpeedButton1: TSpeedButton;

        SpeedButton2: TSpeedButton;

        SpeedButton3: TSpeedButton;

        Image1: TImage;

```

```

procedure SpeedButton1Click(Sender: TObject);

procedure SpeedButton3Click(Sender: TObject);

procedure SpeedButton2Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCreate(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    Foto: TFoto;

implementation

uses UCadAluno, UDMeclipse;

{$R *.dfm}

procedure TFoto.SpeedButton1Click(Sender: TObject);

begin

    JLCVideo1.GrabarImagenDisco;

    image1.Picture.LoadFromFile('foto.bmp');

    image1.visible:=true;

    JLCVideo1.Visible:=false;

    SpeedButton2.Enabled:=true;

```



```

SpeedButton1.Enabled:=false;

SpeedButton3.Enabled:=true;

end;

procedure TFoto.SpeedButton3Click(Sender: TObject);
begin
    SpeedButton2.Enabled:=false;

    SpeedButton1.Enabled:=true;

    speedbutton3.Enabled:=false;

    image1.Visible:=false;

    JLCVideo1.Visible:=true;

end;

procedure TFoto.SpeedButton2Click(Sender: TObject);
begin
    // FrmAluno.DBImage1.Picture.LoadFromFile('foto.bmp');

    DMeclipse.TBalunofoto.LoadFromFile('foto.bmp');

    close;

end;

procedure TFoto.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    action:=cafree;

    foto:=nil;

end;

```

```

procedure TFoto.FormCreate(Sender: TObject);

begin
    JLCVideo1.Activo:=true;

end;

end.

unit UCadInteresse;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Buttons, Grids, DBGrids, Mask, DBCtrls, ExtCtrls, DB,
    ADODB;

type

    TfrmCad_Interesse = class(TForm)

        Panel1: TPanel;

        DBLookupComboBox2: TDBLookupComboBox;

        Label43: TLabel;

        DBEdit14: TDBEdit;

        Label44: TLabel;

        Panel3: TPanel;

        DBGrid2: TDBGrid;

        GroupBox2: TGroupBox;

        BitBtn13: TBitBtn;

```

```

BitBtn14: TBitBtn;

BitBtn15: TBitBtn;

DSinteresse: TDataSource;

ADOinteresse: TADOQuery;

ADOinteressecodigo_alu: TIntegerField;

ADOinteressecodigo_cur: TIntegerField;

ADOinteressedtfecha_esp: TDateTimeField;

ADOinteresseCurso: TStringField;

Label2: TLabel;

DBEdit5: TDBEdit;

Edit1: TEdit;

SpeedButton1: TSpeedButton;

DSAluno_p: TDataSource;

ADOAluno_p: TADOQuery;

ADOAluno_pcodigo_alu: TAutoIncField;

ADOAluno_pnome_alu: TWideStringField;

ADOinteresseNome: TStringField;

BitBtn30: TBitBtn;

procedure BitBtn13Click(Sender: TObject);

procedure SpeedButton1Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

```

```

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn15Click(Sender: TObject);

procedure BitBtn14Click(Sender: TObject);

procedure DBEdit5Exit(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure BitBtn30Click(Sender: TObject);

procedure ADOInteresseBeforePost(DataSet: TDataSet);

    function validar :boolean;

private

    { Private declarations }

public

    { Public declarations }

end;

var

    frmCad_Interesse: TfrmCad_Interesse;

implementation

uses API_F11, UDMeclipse, UPrincipal, UpesqGeral, UCadAluno;

{$R *.dfm}

procedure TfrmCad_Interesse.BitBtn13Click(Sender: TObject);

begin

    if(validar)then

        begin

```

```

if(ADOInteresse.State in [dsinsert])then

begin

    ADOInteresse.Post;

end;

ADOInteresse.Insert;

end;

end;

procedure TfrmCad_Interesse.SpeedButton1Click(Sender: TObject);

begin

    String_tb.Stabela := 'Aluno';

    String_tb.Scampocod := 'codigo_alu';

    String_tb.Scamponome := 'Nome_alu';

    String_tb.Qrytb := ADOAluno_p;

    String_tb.DStb := DSAAluno_p;

    abrirpesq(self);

    if not(ADOAluno_p.IsEmpty)then

begin

    ADOInteresse.Close;

    ADOInteresse.Parameters.ParamByName('codialu').Value      :=      ADOAlu-
no_pcodigo_alu.AsInteger;

    ADOInteresse.Open;

    ADOInteresse.Insert;

    DBEdit5.Text := ADOAluno_pcodigo_alu.AsVariant;

```

```

    Edit1.Text := ADOAluno_pnome_alu.AsString;

end

else

    ShowMessage('Aluno não cadastrado');

end;

procedure TfrmCad_Interesse.FormClose(Sender: TObject;
    var Action: TCloseAction);

begin

    ADOinteresse.Active:=false;

    ADOAluno_p.Active:=false;

    // Application.CreateForm(Tfrmprincipal,frmprincipal);

    // frmprincipal.show;

    action:= cafree;

    frmCad_Interesse:= nil;

end;

procedure TfrmCad_Interesse.FormCloseQuery(Sender: TObject;
    var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de Interes-
ses?', mtconfirmation, [mbyes, mbno], 0) = mryes then

    begin

        ADOinteresse.cancel;

```

```

        canclose:=true;

    end

    else

        canclose:=false;

    end;

procedure TfrmCad_Interesse.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

procedure TfrmCad_Interesse.BitBtn15Click(Sender: TObject);

begin

    ADOInteresse.Edit;

end;

procedure TfrmCad_Interesse.BitBtn14Click(Sender: TObject);

begin

    if not(ADOInteresse.IsEmpty)then

        ADOInteresse.Delete;

        if(DBEdit5.Text = "")then

```

```

    Edit1.Clear;

end;

procedure TfrmCad_Interesse.DBEdit5Exit(Sender: TObject);

begin
    ADOAluno_p.Close;

    ADOAluno_p.SQL.Clear;

    ADOAluno_p.SQL.Add('select codigo_alu, nome_alu from tb_aluno ');

    ADOAluno_p.SQL.Add(' where codigo_alu = :codi ');

    ADOAluno_p.Parameters.ParamByName('codi').Value := DBEdit5.Text;

    ADOAluno_p.Open;

    if not(ADOAluno_p.IsEmpty)then

    begin

        ADOInteresse.Close;

        ADOInteresse.Parameters.ParamByName('codialu').Value := ADOAluno_p.codigo_alu.AsInteger;

        ADOInteresse.Open;

        ADOInteresse.Insert;

        DBEdit5.Text := ADOAluno_p.codigo_alu.AsVariant;

        Edit1.Text := ADOAluno_p.nome_alu.AsString;

    end

    else

        ShowMessage('Aluno não cadastrado');

    end;

```



```
procedure TfrmCad_Interesse.FormShow(Sender: TObject);
```

```
begin
```

```
    Panel3.Enabled := false;
```

```
    frmCad_Interesse.ADOAluno_p.Active := true;
```

```
{ frmCad_Interesse.Edit1.Enabled := true;
```

```
    frmCad_Interesse.DBEdit5.Enabled := true;
```

```
    frmCad_Interesse.SpeedButton1.Enabled := true; }
```

```
    frmCad_Interesse.ADOinteresse.Active := true;
```

```
    frmCad_Interesse.ADOinteresse.Insert;
```

```
end;
```

```
procedure TfrmCad_Interesse.BitBtn30Click(Sender: TObject);
```

```
begin
```

```
    Panel3.Enabled := not Panel3.Enabled;
```

```
    ADOinteresse.Cancel;
```

```
end;
```

```
procedure TfrmCad_Interesse.ADOinteresseBeforePost(DataSet: TDataSet);
```

```
begin
```

```
//validar;
```

```
    if(validar = false)then
```

```
    begin
```

```
        ADOinteresse.Cancel;
```

```

    end;

end;

function TfrmCad_Interesse.validar: boolean;

var

    valida : boolean;

begin

    valida := true;

    if(trim(DBEdit5.Text) = '')then

        valida := false;

    if(trim(DBEdit14.Text) = '')then

        valida := false;

    if(trim(DBLookupComboBox2.Text) = '')then

        valida := false;


    if(valida = false)then

        begin

            ShowMessage('Existem campos necessarios em branco');

            result := false;

        end

    else

        Result := true;

    end;

end;

```

end.

unit UCadTurma;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, DBCtrls, Mask, ExtCtrls, Buttons;

type

TFrmTurma = class(TForm)

Label1: TLabel;

DBEdit1: TDBEdit;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

DBEdit4: TDBEdit;

Label5: TLabel;

DBEdit5: TDBEdit;

Label6: TLabel;

DBEdit6: TDBEdit;

Label7: TLabel;

DBEdit7: TDBEdit;

Label8: TLabel;

DBEdit8: TDBEdit;

```
Label9: TLabel;  
  
DBEdit9: TDBEdit;  
  
Label10: TLabel;  
  
DBMemo1: TDBMemo;  
  
Panel1: TPanel;  
  
Panel2: TPanel;  
  
BitBtn1: TBitBtn;  
  
BitBtn2: TBitBtn;  
  
BitBtn3: TBitBtn;  
  
BitBtn4: TBitBtn;  
  
BitBtn5: TBitBtn;  
  
BitBtn6: TBitBtn;  
  
BitBtn7: TBitBtn;  
  
SpeedButton1: TSpeedButton;  
  
SpeedButton2: TSpeedButton;  
  
DBLookupComboBox1: TDBLookupComboBox;  
  
DBLookupComboBox2: TDBLookupComboBox;  
  
DBRadioGroup1: TDBRadioGroup;  
  
procedure FormClose(Sender: TObject; var Action: TCloseAction);  
  
procedure FormKeyPress(Sender: TObject; var Key: Char);  
  
procedure BitBtn1Click(Sender: TObject);  
  
procedure BitBtn2Click(Sender: TObject);
```

```

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure SpeedButton1Click(Sender: TObject);

procedure SpeedButton2Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmTurma: TFrmTurma;

implementation

uses UDMeclipse, API_F11, UpesqGeral, UPrincipal, UPrincipal2;

{$R *.dfm}

procedure TFrmTurma.FormClose(Sender: TObject; var Action: TCloseAction);

begin

```

```

if menup='1900' then

begin

    frmprincipal2.Enabled:=true;

end

else

begin

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

DMeclipse.TBturma.Active:=false;

DMeclipse.TBcurso.Active:=false;

dmeclipse.TBvoluntario.active:=false;

action:= cafree;

FrmTurma:= nil;

end;

procedure TFrmTurma.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

    begin

        Key := #13;

        SelectNext(ActiveControl,true,true);

    end;

end;

```

```

end;

procedure TFrmTurma.BitBtn1Click(Sender: TObject);
begin
    botoes(self);

    DBLookupComboBox1.Enabled:=true;

    DBLookupComboBox2.Enabled:=true;

    DBMemo1.Enabled:=true;

    mudaedit(self, true);

    DMeclipse.TBturma.Insert;

    DMeclipse.TBturmadt_ini.AsString:=datetostr(date);
end;

procedure TFrmTurma.BitBtn2Click(Sender: TObject);
begin
    if not(DMeclipse.TBturma.IsEmpty)then
    begin
        botoes(self);

        DBLookupComboBox1.Enabled:=true;

        DBLookupComboBox2.Enabled:=true;

        DBMemo1.Enabled:=true;

        mudaedit(self, true);

        DMeclipse.TBturma.Edit;
    end
end

```

```

else

    ShowMessage('Não existe registro para edição.');
```

end;

```

procedure TFrmTurma.BitBtn3Click(Sender: TObject);

begin

    if dmeclipse.TBturma.isempty then

        begin

            showmessage('Não existe registro a ser excluído');
```

exit;

```

        end;

    If messagedlg('Tem certeza que deseja excluir essa locação?'+#13+'Essa operação
é irreversível', mtconfirmation, [mbyes, mbno], 0) = mrno then

        exit;

    DMeclipse.TBturma.Delete;

end;

procedure TFrmTurma.BitBtn4Click(Sender: TObject);

begin

    botoes(self);

    DBLookupComboBox1.Enabled:=false;

    DBLookupComboBox2.Enabled:=false;

    DBMemo1.Enabled:=false;

    mudaedit(self, false);

    DMeclipse.TBturma.Cancel;
```



```

end;

procedure TFrmTurma.BitBtn5Click(Sender: TObject);

begin
    trims(self);

    If DBEdit4.text="" then

    begin

        showmessage('Preencha o nome da turma');

        dbedit4.SetFocus;

        exit;

    end;

    If DBLookupComboBox1.text="" then

    begin

        showmessage('Preencha o nome do curso');

        DBLookupComboBox1.SetFocus;

        exit;

    end;

    If DBLookupComboBox2.text="" then

    begin

        showmessage('Preencha o nome do Professor');

        DBLookupComboBox2.SetFocus;

        exit;

    end;

```

```

If DBedit8.text="" then

begin

    showmessage('Preencha o número de vagas');

    DBedit8.SetFocus;

    exit;

end;

DMeclipse.TBturma.Post;

DBLookupComboBox1.Enabled:=false;

DBLookupComboBox2.Enabled:=false;

DBMemo1.Enabled:=false;

DMeclipse.TBturma.Refresh;

end;

procedure TFrmTurma.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Turma';

    String_tb.Scampocod := 'Codigo_tur';

    String_tb.Scamponome := 'Nome_tur';

    String_tb.Qrytb := DMeclipse.TBturma;

    String_tb.DStb := DMeclipse.DStbturma;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBturmacodigo_tur.AsInteger;

end;

```

```
procedure TFrmTurma.FormCreate(Sender: TObject);
```

```
begin
```

```
    DMeclipse.TBturma.Active:=true;
```

```
    DMeclipse.TBcurso.Active:=true;
```

```
    dmeclipse.TBvoluntario.active:=true;
```

```
end;
```

```
procedure TFrmTurma.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
```

```
begin
```

```
    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de turma?',  
mtconfirmation, [mbytes, mbno], 0) = mryes then
```

```
    begin
```

```
        DMeclipse.TBturma.cancel;
```

```
        canclose:=true;
```

```
    end
```

```
    else
```

```
end;
```

```
procedure TFrmTurma.SpeedButton1Click(Sender: TObject);
```

```
begin
```

```
    String_tb.Stabela := 'curso';
```

```
    String_tb.Scampocod := 'codigo_cur';
```

```
    String_tb.Scamponome := 'Nome_cur';
```

```
    String_tb.Qrytb := DMeclipse.TBcurso;
```

```

String_tb.DStb := DMeclipse.DStbcurso;

abrirpesq(self);

DMeclipse.TBturmacodigo_cur.AsString:=DMeclipse.TBcursocodigo_cur.AsString;

end;

procedure TFrmTurma.SpeedButton2Click(Sender: TObject);

begin

String_tb.Stabela := 'voluntario';

String_tb.Scampocod := 'codigo_vol';

String_tb.Scamponome := 'Nome_vol';

String_tb.Qrytb := DMeclipse.TBvoluntario;

String_tb.DStb := DMeclipse.DStbvoluntario;

abrirpesq(self);

DMec-
lipse.TBturmacodigo_vol.AsString:=DMeclipse.TBvoluntariocodigo_vol.AsString;

// DBEdit3.Text:=DMeclipse.TBcursocodigo_cur.AsString;

end;

procedure TFrmTurma.BitBtn6Click(Sender: TObject);

begin

close;

end;

end.

unit UcadHistorico;

interface

```

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DB, ADODB, Buttons, StdCtrls, Mask, DBCtrls, ExtCtrls;

type

TfrmCadHistorico = class(TForm)

Panel1: TPanel;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

BitBtn5: TBitBtn;

BitBtn6: TBitBtn;

BitBtn7: TBitBtn;

Panel2: TPanel;

DBEdit5: TDBEdit;

SpeedButton1: TSpeedButton;

ADOAluno_p: TADOQuery;

DSAluno_p: TDataSource;

Edit1: TEdit;

ADOAluno_pcodigo_alu: TAutoIncField;

ADOAluno_pnome_alu: TWideStringField;

DBEdit1: TDBEdit;

```
SpeedButton2: TSpeedButton;

Edit2: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

DBEdit2: TDBEdit;

Label6: TLabel;

DBMemo1: TDBMemo;

Label7: TLabel;

DBEdit4: TDBEdit;

Label8: TLabel;

DBText1: TDBText;

ComboBox1: TComboBox;

procedure SpeedButton1Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);
```

```

procedure BitBtn7Click(Sender: TObject);

procedure BitBtn8Click(Sender: TObject);

procedure SpeedButton2Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure FormActivate(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    frmCadHistorico: TfrmCadHistorico;

implementation

uses UDMeclipse, API_F11, UpesqGeral, Unit1, pesq_t, Math, UPrincipal,
    UPrincipal2;

{$R *.dfm}

procedure TfrmCadHistorico.SpeedButton1Click(Sender: TObject);

begin

    String_tb.Stabela := 'Aluno';

```

```

String_tb.Scampocod := 'codigo_alu';

String_tb.Scamponome := 'Nome_alu';

String_tb.Qrytb := ADOAluno_p;

String_tb.DStb := DSAAluno_p;

abrirpesq(self);

if not(ADOAluno_p.IsEmpty)then

begin

    DBEdit5.Text := ADOAluno_p.codigo_alu.AsVariant;

    Edit1.Text := ADOAluno_p.nome_alu.AsString;

    DMeclipse.TBhistoricocodigo_alu.AsInteger := ADOAluno_p.codigo_alu.AsInteger;

end;

end;

procedure TfrmCadHistorico.FormClose(Sender: TObject;

var Action: TCloseAction);

begin

    DMeclipse.TBhistorico.Active:=false;

    if menup='1900' then

    begin

        frmprincipal2.Enabled:=true;

    end

    else

    begin

```



```

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

action:= cafree;

frmCadHistorico:= nil;

end;

procedure TfrmCadHistorico.FormKeyPress(Sender: TObject; var Key: Char);
begin
    if(Key = #13)then
    begin
        Key := #13;

        SelectNext(ActiveControl,true,true);

    end;

end;

procedure TfrmCadHistorico.BitBtn1Click(Sender: TObject);
begin
    botoes(self);

    mudaedit(self, true);

    DMeclipse.TBhistorico.Insert;

    ComboBox1.Enabled := true;

end;

procedure TfrmCadHistorico.BitBtn2Click(Sender: TObject);

```

```

begin

    if not(DMeclipse.TBhistorico.IsEmpty)then

        begin

            botoes(self);

            mudaedit(self, true);

            DMeclipse.TBhistorico.Edit;

            ComboBox1.Enabled := true;

        end

    else

        begin

            ShowMessage('Não existe registro para edição.');
```

exit;

```

        end;

    end;

procedure TfrmCadHistorico.BitBtn3Click(Sender: TObject);

begin

    if (DMeclipse.TBhistorico.IsEmpty)then

        begin

            ShowMessage('Não existe registro a ser excluido');
```

exit;

```

        end;

        If messagedlg('Tem certeza que deseja excluir esse aluno?'+#13+'Essa operação é
irreversível!', mtconfirmation, [mbytes, mbno], 0) = mrno then

```

```

    exit;

    DMeclipse.TBhistorico.Delete;

end;

procedure TfrmCadHistorico.BitBtn4Click(Sender: TObject);

begin

    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBhistorico.Cancel;

    ComboBox1.Enabled := false;

end;

procedure TfrmCadHistorico.BitBtn5Click(Sender: TObject);

begin

    If DBEdit5.text="" then

    begin

        showmessage('Necessário escolher um aluno');

        dbedit5.SetFocus;

        exit;

    end;

    If DBEdit1.text="" then

    begin

        showmessage('Necessário escolher uma turma');

        dbedit1.SetFocus;

```

```

    exit;

end;

botoes(self);

mudaedit(self, false);

ComboBox1.Enabled := false;

if(ComboBox1.ItemIndex = 0)then
begin
    DMeclipse.TBhistoricostatus_alu.AsString := 'C';
end

else if(ComboBox1.ItemIndex = 1)then
begin
    DMeclipse.TBhistoricostatus_alu.AsString := 'F';
end

else if(combobox1.ItemIndex=2)then
begin
    DMeclipse.TBhistoricostatus_alu.AsString := 'L';
end;

DMeclipse.TBhistorico.Post;

// DMeclipse.TBhistorico.Refresh;

end;

procedure TfrmCadHistorico.BitBtn7Click(Sender: TObject);

begin

```

```

If(Frmpesq_hist = nil)then
begin
    Application.CreateForm(TFrmpesq_hist, Frmpesq_hist);
    Frmpesq_hist.Show;
end;

ComboBox1.Enabled := false;

end;

procedure TfrmCadHistorico.BitBtn8Click(Sender: TObject);
begin
    Close;
end;

procedure TfrmCadHistorico.SpeedButton2Click(Sender: TObject);
begin
    If(Frmpesq_tur = nil)then
    begin
        Application.CreateForm(TFrmpesq_tur, Frmpesq_tur);
        Frmpesq_tur.ShowModal;
    end;

    if not(DMeclipse.TBturma.IsEmpty)then
    begin
        DBEdit1.Text := DMeclipse.TBturmacodigo_tur.AsVariant;
        Edit2.Text := DMeclipse.TBturmanome_tur.AsString;
    end;
end;

```

```

        DMeclipse.TBhistoricocodigo_tur.AsInteger          :=          DMeclipse.TBturmacodigo_tur.AsVariant;

    end;

end;

procedure TfrmCadHistorico.BitBtn6Click(Sender: TObject);

begin

    Close;

end;

procedure TfrmCadHistorico.FormActivate(Sender: TObject);

begin

    if(DMeclipse.TBhistoricostatus_alu.AsString = 'C')then

        ComboBox1.ItemIndex := 0

    else if(DMeclipse.TBhistoricostatus_alu.AsString = 'F') then

        ComboBox1.ItemIndex := 1;

end;

procedure TfrmCadHistorico.FormCreate(Sender: TObject);

begin

    DMeclipse.TBaluno.Active:=true;

end;

procedure TfrmCadHistorico.FormCloseQuery(Sender: TObject;

    var CanClose: Boolean);

begin

```

```
If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de histórico?', mtconfirmation, [mbytes, mbno], 0) = mryes then
```

```
begin
```

```
    DMeclipse.TBhistorico.cancel;
```

```
    canclosen:=true;
```

```
end
```

```
else
```

```
    canclosen:=false;
```

```
end;
```

```
end.
```

```
unit UCadContato;
```

```
interface
```

```
uses
```

```
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
```

```
    Dialogs, DB, ADODB, StdCtrls, Buttons, Grids, DBGrids, ExtCtrls, Mask,
```

```
    DBCtrls;
```

```
type
```

```
    TFrmCadContato = class(TForm)
```

```
        DBLookupComboBox1: TDBLookupComboBox;
```

```
        Label41: TLabel;
```

```
        DBEdit5: TDBEdit;
```

```
        Label42: TLabel;
```

```
        Panel2: TPanel;
```

DBGrid1: TDBGrid;
GroupBox1: TGroupBox;
BitBtn8: TBitBtn;
BitBtn9: TBitBtn;
BitBtn12: TBitBtn;
ADOQuery1: TADOQuery;
ADOQuery1codigo_alu: TIntegerField;
ADOQuery1ptel_fnal: TWideStringField;
ADOQuery1fone_fnal: TWideStringField;
DataSource1: TDataSource;
DBEdit1: TDBEdit;
Label2: TLabel;
SpeedButton1: TSpeedButton;
Edit1: TEdit;
ADOAluno_p: TADOQuery;
ADOAluno_pcodigo_alu: TAutoIncField;
ADOAluno_pnome_alu: TWideStringField;
DSAluno_p: TDataSource;
ADOAssociado: TADOQuery;
ADOVoluntario: TADOQuery;
dsAssociado: TDataSource;
dsVoluntario: TDataSource;


```

ADOVoluntariocodigo_vol: TIntegerField;

ADOVoluntariotpfon_vol: TWideStringField;

ADOVoluntariofone_vol: TWideStringField;

ADOAssociadocodigo_ass: TIntegerField;

ADOAssociadotpfone_fnass: TWideStringField;

ADOAssociadofone_fnass: TWideStringField;

ADOAssociado_p: TADOQuery;

dsAssociado_p: TDataSource;

ADOVoluntario_p: TADOQuery;

dsVoluntario_p: TDataSource;

ADOAssociado_pcodigo_ass: TAutoIncField;

ADOAssociado_pnome_ass: TWideStringField;

ADOVoluntario_pcodigo_vol: TAutoIncField;

ADOVoluntario_pnome_vol: TWideStringField;

BitBtn30: TBitBtn;

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure BitBtn8Click(Sender: TObject);

procedure BitBtn12Click(Sender: TObject);

procedure BitBtn9Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure SpeedButton1Click(Sender: TObject);

```

```

procedure DBEdit1Exit(Sender: TObject);

procedure FormShow(Sender: TObject);

function validar :boolean;

procedure ADOQuery1BeforePost(DataSet: TDataSet);

procedure BitBtn30Click(Sender: TObject);

procedure ADOAssociadoBeforePost(DataSet: TDataSet);

procedure ADOVoluntarioBeforePost(DataSet: TDataSet);

private

    { Private declarations }

public

    { Public declarations }

    Tela : String;

end;

var

    FrmCadContato: TFrmCadContato;

implementation

uses UCadAluno, API_F11, UpesqGeral, UCadAssociado, UDMeclipse;

{$R *.dfm}

procedure TFrmCadContato.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

```

```

    Key := #13;

    SelectNext(ActiveControl,true,true);

end;

end;

procedure TFrmCadContato.FormClose(Sender: TObject;

    var Action: TCloseAction);

begin

    //ALUNO

    if(Tela = 'ALUNO')then

    begin

        ADOQuery1.Active := false;

        ADOAluno_p.Active := false;

    end;

    //ASSOCIADO

    if(tela = 'ASSOCIADO')then

    begin

        ADOAssociado.Active := false;

        ADOAssociado_p.Active := false;

    end;

    //VOLUNTARIO

    if(tela = 'VOLUNTARIO')then

    begin

```

```

ADOVoluntario.Active := false;

ADOVoluntario_p.Active := false;

end;

// Application.CreateForm(Tfrmprincipal,frmprincipal);

// FrmAssociado.show;

action:= cafree;

FrmCadContato := nil;

end;

procedure TFrmCadContato.BitBtn8Click(Sender: TObject);

begin

//validar;

if(validar)then

begin

if(Tela = 'ALUNO')then

begin

if(ADOQuery1.State in [dsinsert])then

begin

ADOQuery1.Post;

end;

ADOQuery1.Insert;

end;

end;

```

```

if(tela = 'ASSOCIADO')then
begin
    if(ADOAssociado.State in [dsinsert])then
        begin
            ADOAssociado.Post;
        end;
        ADOAssociado.Insert;
    end;
if(tela = 'VOLUNTARIO')then
begin
    if(ADOVoluntario.State in [dsinsert])then
        begin
            ADOVoluntario.Post;
        end;
        ADOVoluntario.Insert;
    end;
end;
end;
procedure TFrmCadContato.BitBtn12Click(Sender: TObject);
begin
    if(Tela = 'ALUNO')then
        begin

```

```

        ADOQuery1.Edit;

end;

if(tela = 'ASSOCIADO')then

begin

    ADOAssociado.Edit;

end;

if(tela = 'VOLUNTARIO')then

begin

    ADOVoluntario.Edit;

end;

// Panel2.Enabled := true;

end;

procedure TFrmCadContato.BitBtn9Click(Sender: TObject);

begin

    if(Tela = 'ALUNO')then

    begin

        if not(ADOQuery1.IsEmpty)then

            ADOQuery1.Delete;

        if(DBEdit5.Text = "")then

            Edit1.Clear;

        end;

end;

```

```

if(tela = 'ASSOCIADO')then

begin

    if not(ADOAssociado.IsEmpty)then

        ADOAssociado.Delete;

    end;

if(tela = 'VOLUNTARIO')then

begin

    if not(ADOVoluntario.IsEmpty)then

        ADOVoluntario.Delete;

    end;

    Panel2.Enabled := false;

end;

procedure TFrmCadContato.FormCloseQuery(Sender: TObject;

var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de conta-

tos?', mtconfirmation, [mbyes, mbno], 0) = mryes then

begin

    if(Tela = 'ALUNO')then

begin

        ADOQuery1.cancel;

        ADOAluno_p.Cancel;

    end;

```

```

if(tela = 'ASSOCIADO')then
begin
    ADOAssociado.Cancel;
    ADOAssociado_p.Cancel;
end;

if(tela = 'VOLUNTARIO')then
begin
    ADOVoluntario.Cancel;
    ADOVoluntario_p.Cancel;
end;

canclose:=true;

end

else

    canclose:=false;

end;

procedure TFrmCadContato.SpeedButton1Click(Sender: TObject);

begin
    //ALUNO

    if(Tela = 'ALUNO')then
begin
    String_tb.Stabela := 'Aluno';

    String_tb.Scampocod := 'codigo_alu';

```



```

String_tb.Scamponome := 'Nome_alu';

String_tb.Qrytb := ADOAluno_p;

String_tb.DStb := DSAluno_p;

abrirpesq(self);

if not(ADOAluno_p.IsEmpty)then

begin

    ADOQuery1.Close;

    ADOQuery1.Parameters.ParamByName('codialu').Value      :=      ADOAlu-
no_pcodigo_alu.AsInteger;

    ADOQuery1.Open;

    ADOQuery1.Insert;

    DBEdit1.Text := ADOAluno_pcodigo_alu.AsVariant;

    Edit1.Text := ADOAluno_pnome_alu.AsString;

end

else

    if(Tela = 'ALUNO')then

        ShowMessage('Aluno não cadastrado');

end;

//ASSOCIADO

if(Tela = 'ASSOCIADO')then

begin

    String_tb.Stabela := 'Associado';

    String_tb.Scampocod := 'codigo_ass';

```

```

String_tb.Scamponome := 'Nome_ass';

String_tb.Qrytb := ADOAssociado_p;

String_tb.DStb := dsAssociado_p;

abrirpesq(self);

if not(ADOAssociado_p.IsEmpty)then

begin

    with ADOAssociado do

    begin

        Close;

        Parameters.ParamByName('CODIASS').Value      :=      ADOAssocia-
do_pcodigo_ass.AsInteger;

        Open;

        Insert;

        DBEdit1.Text := ADOAssociado_pcodigo_ass.AsVariant;

        Edit1.Text := ADOAssociado_pnome_ass.AsString;

    end;

end

else

    ShowMessage('Associado não cadastrado');

end;

if(Tela = 'VOLUNTARIO')then

begin

```

```

String_tb.Stabela := 'VOLUNTARIO';

String_tb.Scampocod := 'CODIGO_VOL';

String_tb.Scamponome := 'NOME_VOL';

String_tb.Qrytb := ADOVoluntario_p;

String_tb.DStb := dsVoluntario_p;

abrirpesq(self);

with ADOVoluntario do

begin

    if not(ADOVoluntario_p.IsEmpty)then

        begin

            Close;

            Parameters.ParamByName('CODIVOL').Value      :=      ADOVolunta-
rio_pcodigo_vol.AsInteger;

            Open;

            Insert;

            DBEdit1.Text := ADOVoluntario_pcodigo_vol.AsVariant;

            Edit1.Text := ADOVoluntario_pnome_vol.AsString;

        end

    else

        ShowMessage('Associado não cadastrado');

    end;

end;

end;

end;

```

```

procedure TFrmCadContato.DBEdit1Exit(Sender: TObject);

begin
    if(trim(DBEdit1.Text) <> '')then
    begin
        if(tela = 'ALUNO')then
        begin
            ADOAluno_p.Close;

            ADOAluno_p.SQL.Clear;

            ADOAluno_p.SQL.Add('select codigo_alu, nome_alu from tb_aluno ');

            ADOAluno_p.SQL.Add(' where codigo_alu = :codi ');

            ADOAluno_p.Parameters.ParamByName('CODI').Value :=
StrToInt(DBEdit1.Text);

            ADOAluno_p.Open;

            if not(ADOAluno_p.IsEmpty)then
            begin
                ADOQuery1.Close;

                ADOQuery1.Parameters.ParamByName('codialu').Value := ADOAlu-
no_pcodigo_alu.AsInteger;

                ADOQuery1.Open;

                ADOQuery1.Insert;

                DBEdit1.Text := ADOAluno_pcodigo_alu.AsVariant;

                Edit1.Text := ADOAluno_pnome_alu.AsString;
            end
        end
    end

```

```

else

begin

    ShowMessage('Aluno não cadastrado');

    DBEdit1.SetFocus;

end;

end;

if(tela = 'ASSOCIADO')then

begin

    ADOAssociado_p.Close;

    ADOAssociado_p.SQL.Clear;

    ADOAssociado_p.SQL.Add('select codigo_ass, nome_ass from tb_associado
');

    ADOAssociado_p.SQL.Add(' where codigo_ass = :codi
');

    ADOAssociado_p.Parameters.ParamByName('CODI').Value
:=
StrToInt(DBEdit1.Text);

    ADOAssociado_p.Open;

    if not(ADOAssociado_p.IsEmpty)then

begin

    with ADOAssociado do

begin

        Close;

        Parameters.ParamByName('CODIASS').Value
:=
ADOAssocia-
do_p.codigo_ass.AsInteger;

```

```

    Open;

    Insert;

    DBEdit1.Text := ADOAssociado_pcodigo_ass.AsVariant;

    Edit1.Text := ADOAssociado_pnome_ass.AsString;

end;

end

else

begin

    ShowMessage('Associado não cadastrado');

    DBEdit1.SetFocus;

end;

end;

if(tela = 'VOLUNTARIO')then

begin

    ADOVoluntario_p.Close;

    ADOVoluntario_p.SQL.Clear;

    ADOVoluntario_p.SQL.Add('select codigo_vol, nome_vol from tb_voluntario ');

    ADOVoluntario_p.SQL.Add(' where codigo_vol = :codi ');

    ADOVoluntario_p.Parameters.ParamByName('CODI').Value :=
StrToInt(DBEdit1.Text);

    ADOVoluntario_p.Open;

    with ADOVoluntario do

begin

```

```

if not(ADOVoluntario_p.IsEmpty)then
begin
    Close;

    Parameters.ParamByName('CODIVOL').Value      :=      ADOVolunta-
rio_pcodigo_vol.AsInteger;

    Open;

    Insert;

    DBEdit1.Text := ADOVoluntario_pcodigo_vol.AsVariant;

    Edit1.Text := ADOVoluntario_pnome_vol.AsString;
end

else

begin

    ShowMessage('Voluntario não cadastrado');

    DBEdit1.SetFocus;

end;

end;

end;

end;

end;

```

```

procedure TFrmCadContato.FormShow(Sender: TObject);

begin

    DBEdit5.DataField := "";

```

```

DBEdit1.DataField := '';

DBLookupComboBox1.DataField := '';

Panel2.Enabled := false;

//ALUNO

if(tela = 'ALUNO')then

begin

    Label2.Caption := 'Aluno';

    ADOQuery1.Active := true;

    ADOAluno_p.Active := true;

    DBEdit1.DataSource := DataSource1;

    DBEdit1.DataField := 'codigo_alu';

    DBLookupComboBox1.DataSource := DataSource1;

    DBLookupComboBox1.DataField := 'tptel_fnalu';

    DBEdit5.DataSource := DataSource1;

    DBEdit5.DataField := 'fone_fnalu';

    DBGrid1.DataSource := DataSource1;

    ADOQuery1.Insert;

end;

//ASSOCIADO

if(tela = 'ASSOCIADO')then

begin

    Label2.Caption := 'Associado';

```



```

ADOAssociado.Active := true;

ADOAssociado_p.Active := true;

DBEdit1.DataSource := dsAssociado;

DBEdit1.DataField := 'codigo_ass';

DBLookupComboBox1.DataSource := dsAssociado;

DBLookupComboBox1.DataField := 'tpfone_fnass';

DBEdit5.DataSource := dsAssociado;

DBEdit5.DataField := 'fone_fnass';

DBGrid1.DataSource := dsAssociado;

ADOAssociado.Insert;

end;

//VOLUNTARIO

if(tela = 'VOLUNTARIO')then
begin

    Label2.Caption := 'Voluntario';

    ADOVoluntario.Active := true;

    ADOVoluntario_p.Active := true;

    DBEdit1.DataSource := dsVoluntario;

    DBEdit1.DataField := 'codigo_vol';

    DBLookupComboBox1.DataSource := dsVoluntario;

    DBLookupComboBox1.DataField := 'tpfon_vol';

    DBEdit5.DataSource := dsVoluntario;

```

```

DBEdit5.DataField := 'fone_vol';

DBGrid1.DataSource := dsVoluntario;

ADOVoluntario.Insert;

end;

DBGrid1.Columns[0].Title.Caption := 'Cod.';

DBGrid1.Columns[1].Title.Caption := 'Tipo';

DBGrid1.Columns[2].Title.Caption := 'Número';

end;

function TFrmCadContato.validar: boolean;

var

    valida: boolean;

begin

    valida := true;

    if(trim(DBEdit5.Text) = '')then

        valida := false;

    if(trim(DBEdit1.Text) = '')then

        valida := false;

    if(trim(DBLookupComboBox1.Text) = '')then

        valida := false;

    if(valida = false)then

        begin

            ShowMessage('Existem campos necessarios em branco');

```

```

        result := false;

    end

    else

        Result := true;

    end;

procedure TFrmCadContato.ADOQuery1BeforePost(DataSet: TDataSet);

begin

    //validar;

    if(validar = false)then

        begin

            ADOQuery1.Cancel;

            ADOAsociado.Cancel;

            ADOVoluntario.Cancel;

        end;

    end;

procedure TFrmCadContato.BitBtn30Click(Sender: TObject);

begin

    Panel2.Enabled := not Panel2.Enabled;

    ADOQuery1.Cancel;

    ADOAsociado.Cancel;

    ADOVoluntario.Cancel;

end;

```

```

procedure TFrmCadContato.ADOAssociadoBeforePost(DataSet: TDataSet);

begin

//validar;

if(validar = false)then

begin

ADOQuery1.Cancel;

ADOAssociado.Cancel;

ADOVoluntario.Cancel;

end;

end;

procedure TFrmCadContato.ADOVoluntarioBeforePost(DataSet: TDataSet);

begin

//validar;

if(validar = false)then

begin

ADOQuery1.Cancel;

ADOAssociado.Cancel;

ADOVoluntario.Cancel;

end;

end;

end.

unit UCadLivro;

```

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, Mask, DBCtrls, ExtCtrls, Buttons, DB, ADODB;

type

TFrmLivro = class(TForm)

Label1: TLabel;

DBEdit1: TDBEdit;

Label2: TLabel;

DBEdit2: TDBEdit;

Label3: TLabel;

DBEdit3: TDBEdit;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

DBEdit6: TDBEdit;

DBMemo1: TDBMemo;

Panel1: TPanel;

Panel2: TPanel;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

```
BitBtn4: TBitBtn;

BitBtn5: TBitBtn;

BitBtn6: TBitBtn;

BitBtn7: TBitBtn;

GENERO: TADOQuery;

GENEROcodigo_gen: TAutoIncField;

GENEROnome_gen: TWideStringField;

GENEROcor_gen: TWideStringField;

SpeedButton1: TSpeedButton;

DBRadioGroup1: TDBRadioGroup;

DBLookupComboBox1: TDBLookupComboBox;

DBText1: TDBText;

Label7: TLabel;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);
```

```

    procedure SpeedButton1Click(Sender: TObject);

    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

    procedure FormCreate(Sender: TObject);

private
    { Private declarations }

public
    { Public declarations }

end;

var
    FrmLivro: TFrmLivro;

implementation

uses UDMeclipse, API_F11, UpesqGeral, ComObj, ConvUtils, Math, UPrincipal,
    UPrincipal2;

{$R *.dfm}

procedure TFrmLivro.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    DMeclipse.TBlivro.active:=false;

    DMeclipse.TBgenero.active:=false;

    if menup='1900' then
    begin
        frmprincipal2.Enabled:=true;
    end;
end;

```

```

end

else

begin

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

action:= cafree;

FrmLivro:= nil;

end;

procedure TFrmLivro.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

        begin

            Key := #13;

            SelectNext(ActiveControl,true,true);

        end;

    end;

procedure TFrmLivro.BitBtn1Click(Sender: TObject);

begin

    botoes(self);

    DBLookupComboBox1.enabled:=true;

    mudaedit(self, true);

```



```

DMeclipse.TBlivro.Insert;

DMeclipse.TBlivrodisp_liv.AsString:='d';

end;

procedure TFrmLivro.BitBtn2Click(Sender: TObject);

begin

    if not(DMeclipse.TBlivro.IsEmpty)then

        begin

            botoes(self);

            DBLookupComboBox1.enabled:=true;

            mudaedit(self, true);

            DMeclipse.TBlivro.Edit;

        end

    else

        ShowMessage('Não existe registro para edição.');
```

end;

```

procedure TFrmLivro.BitBtn3Click(Sender: TObject);

begin

    if dmeclipse.TBlivro.isempty then

        begin

            showmessage('Não existe registro a ser excluído');
```

exit;

```

end;

If messagedlg('Tem certeza que deseja excluir esse curso?'+#13+'Essa operação é
irreversível', mtconfirmation, [mbytes, mbno], 0) = mrno then

    exit;

DMeclipse.TBlivro.Delete;

end;

procedure TFrmLivro.BitBtn4Click(Sender: TObject);

begin

    botoes(self);

    DBLookupComboBox1.enabled:=false;

    mudaedit(self, false);

    DMeclipse.TBlivro.Cancel;

end;

procedure TFrmLivro.BitBtn5Click(Sender: TObject);

begin

    trims(self);

    if DBEdit1.text="" then

        begin

            showmessage('Preencha o Código de Tombo do Livro');

            DBEdit1.SetFocus;

            exit;

        end;

    if DBEdit2.text="" then

```

```

begin

    showmessage('Preencha o Nome do Livro');

    DBEdit2.SetFocus;

    exit;

end;

if DBLookupComboBox1.text="" then

begin

    showmessage('Preencha o Genêro do Livro');

    DBLookupComboBox1.SetFocus;

    exit;

end;

if DBedit6.text="" then

begin

    showmessage('Preencha o Código do Livro');

    DBEdit6.SetFocus;

    exit;

end;

botoes(self);

DBLookupComboBox1.enabled:=false;

mudaedit(self, false);

DMeclipse.TBlivro.Post;

DMeclipse.TBlivro.Refresh;

```

```

end;

procedure TFrmLivro.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Livro';

    String_tb.Scampocod := 'codigo_tombo';

    String_tb.Scamponome := 'Nome_liv';

    String_tb.Qrytb := DMeclipse.TBlivro;

    String_tb.DStb := DMeclipse.DStblivro;

    abrirpesq(self);

    // String_tb.resultcodi := DMeclipse.TBlivrotombo_liv.AsInteger;

end;

procedure TFrmLivro.BitBtn6Click(Sender: TObject);

begin

    Close;

end;

procedure TFrmLivro.SpeedButton1Click(Sender: TObject);

begin

    String_tb.Stabela := 'Genero';

    String_tb.Scampocod := 'codigo_gen';

    String_tb.Scamponome := 'Nome_gen';

    String_tb.Qrytb := DMeclipse.TBgenero;

    String_tb.DStb := DMeclipse.DStbgenero;

```

```

    abrirpesq(self);

    dmeclipse.TBlivrocodigo_gen.AsString          :=          DMec-
lipse.TBgenerocodigo_gen.AsString;

end;

procedure TFrmLivro.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de livros?',
mtconfirmation, [mbyes, mbno], 0) = mryes then

        begin

            DMeclipse.TBlivro.cancel;

            candclose:=true;

        end

    else

        candclose:=false;

    end;

procedure TFrmLivro.FormCreate(Sender: TObject);

begin

    DMeclipse.TBlivro.active:=true;

    DMeclipse.TBgenero.active:=true;

end;

end.

unit UCadGenero;

interface

```

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DBCtrls, StdCtrls, Mask, ExtCtrls, Buttons;

type

TFrmGenero = class(TForm)

Label1: TLabel;

Label2: TLabel;

DBEdit2: TDBEdit;

Label3: TLabel;

DBEdit3: TDBEdit;

DBText1: TDBText;

Panel1: TPanel;

Panel2: TPanel;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

BitBtn5: TBitBtn;

BitBtn6: TBitBtn;

BitBtn7: TBitBtn;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormKeyPress(Sender: TObject; var Key: Char);

```

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure FormCreate(Sender: TObject);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmGenero: TFrmGenero;

implementation

uses UDMeclipse, API_F11, UpesqGeral, UPrincipal, UPrincipal2;

{$R *.dfm}

procedure TFrmGenero.FormClose(Sender: TObject; var Action: TCloseAction);

begin

    DMeclipse.TBgenero.Active:=false;

```

```

if menup='1900' then

begin

    frmprincipal2.Enabled:=true;

end

else

begin

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

action:= cafree;

FrmGenero:= nil;

end;

procedure TFrmGenero.FormKeyPress(Sender: TObject; var Key: Char);

begin

    if(Key = #13)then

    begin

        Key := #13;

        SelectNext(ActiveControl,true,true);

    end;

end;

procedure TFrmGenero.BitBtn1Click(Sender: TObject);

begin

```



```

botoes(self);

mudaedit(self, true);

DMeclipse.TBgenero.Insert;

end;

procedure TFrmGenero.BitBtn2Click(Sender: TObject);

begin

    if not(DMeclipse.TBgenero.IsEmpty)then

        begin

            botoes(self);

            mudaedit(self, true);

            DMeclipse.TBgenero.Edit;

        end

    else

        ShowMessage('Não existe registro para edição.');
```

```

end;

procedure TFrmGenero.BitBtn3Click(Sender: TObject);

begin

    if dmeclipse.TBcurso.isempty then

        begin

            showmessage('Não existe registro a ser excluído');
```

```

            exit;

        end;

end;

```

```
If messagedlg('Tem certeza que deseja excluir esse curso?'+#13+'Essa operação é irreversível', mtconfirmation, [mbytes, mbno], 0) = mrno then
```

```
    exit;
```

```
    DMeclipse.TBgenero.Delete;
```

```
end;
```

```
procedure TFrmGenero.BitBtn4Click(Sender: TObject);
```

```
begin
```

```
    botoes(self);
```

```
    mudaedit(self, false);
```

```
    DMeclipse.TBgenero.Cancel;
```

```
end;
```

```
procedure TFrmGenero.BitBtn5Click(Sender: TObject);
```

```
begin
```

```
    trims(self);
```

```
    if DBEdit2.Text="" then
```

```
    begin
```

```
        showmessage('Preencha o nome do gênero');
```

```
        DBEdit2.SetFocus;
```

```
    exit;
```

```
end;
```

```
    if DBEdit3.Text="" then
```

```
    begin
```

```

    showmessage('Preencha a cor do gênero');

    DBEdit3.SetFocus;

    exit;

end;

botoes(self);

mudaedit(self, false);

DMeclipse.TBgenero.Post;

DMeclipse.TBgenero.Refresh;

end;

procedure TFrmGenero.BitBtn6Click(Sender: TObject);

begin

    Close;

end;

procedure TFrmGenero.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Genero';

    String_tb.Scampocod := 'Codigo_gen';

    String_tb.Scamponome := 'Nome_gen';

    String_tb.Qrytb := DMeclipse.TBgenero;

    String_tb.DStb := DMeclipse.DStbgenero;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBgenerocodigo_gen.AsInteger;

```

```

end;

procedure TFrmGenero.FormCloseQuery(Sender: TObject;
    var CanClose: Boolean);
begin
    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de gê-
ros?', mtconfirmation, [mbyes, mbno], 0) = mryes then
    begin
        DMeclipse.TBgenero.cancel;

        canclose:=true;
    end
    else
        canclose:=false;
end;

procedure TFrmGenero.FormCreate(Sender: TObject);
begin
    DMeclipse.TBgenero.Active:=true;
end;

end.

unit UCadLocador;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, ExtCtrls, DBCtrls, Mask, Buttons, ppCtrls, ppVar,

```

ppPrnabl, ppClass, ppDB, ppBands, ppCache, ppProd, ppReport, ppComm,
ppRelatv, ppDBPipe, DB, ADODB;

type

TFrmLocador = class(TForm)

Label1: TLabel;

DBEdit1: TDBEdit;

Label2: TLabel;

DBEdit2: TDBEdit;

Label4: TLabel;

DBEdit4: TDBEdit;

Label5: TLabel;

DBEdit5: TDBEdit;

Label6: TLabel;

DBEdit6: TDBEdit;

Label7: TLabel;

DBEdit7: TDBEdit;

Label8: TLabel;

Label9: TLabel;

DBEdit9: TDBEdit;

DBRadioGroup1: TDBRadioGroup;

Panel1: TPanel;

Panel2: TPanel;

BitBtn1: TBitBtn;
BitBtn2: TBitBtn;
BitBtn3: TBitBtn;
BitBtn4: TBitBtn;
BitBtn5: TBitBtn;
BitBtn6: TBitBtn;
BitBtn7: TBitBtn;
Button1: TButton;
ppDBPipeline1: TppDBPipeline;
ppReport1: TppReport;
ppHeaderBand1: TppHeaderBand;
ppDetailBand1: TppDetailBand;
ppFooterBand1: TppFooterBand;
ppLabel1: TppLabel;
ppDBText1: TppDBText;
ppDBText2: TppDBText;
ppDBText3: TppDBText;
ppDBText4: TppDBText;
ppDBText5: TppDBText;
ppDBText6: TppDBText;
ppLabel2: TppLabel;
ppLabel3: TppLabel;

ppLabel4: TppLabel;
ppLabel5: TppLabel;
ppLabel6: TppLabel;
ppLabel7: TppLabel;
ppShape1: TppShape;
ppSystemVariable1: TppSystemVariable;
ppLine1: TppLine;
ppLabel8: TppLabel;
ppSystemVariable2: TppSystemVariable;
ppLabel9: TppLabel;
ppLabel10: TppLabel;
ppDBText7: TppDBText;
ppLabel11: TppLabel;
ppDBText8: TppDBText;
ppLine2: TppLine;
ADOqryteste: TADOQuery;
DSteste: TDataSource;
ppDBPipeline2: TppDBPipeline;
Button2: TButton;
ADOqrytestesigla_tf: TWideStringField;
ADOqrytestetphone_tf: TWideStringField;
ADOqrytestestatus_tf: TWideStringField;

ADOqrytestetb_alunocodigo_alu: TAutoIncField;

ADOqrytestenome_alu: TWideStringField;

ADOqrytestesx_alu: TWideStringField;

ADOqrytestedtnasc_alu: TDateTimeField;

ADOqrytesterg_alu: TWideStringField;

ADOqrytestenacional_alu: TWideStringField;

ADOqrytestenatural_alu: TWideStringField;

ADOqrytesteend_alu: TWideStringField;

ADOqrytestenum_alu: TIntegerField;

ADOqrytestecompl_alu: TWideStringField;

ADOqrytestebairro_alu: TWideStringField;

ADOqrytestecep_alu: TWideStringField;

ADOqrytestecid_alu: TWideStringField;

ADOqrytesteuf_alu: TWideStringField;

ADOqrytestemae_alu: TWideStringField;

ADOqrytesteprofmae_alu: TWideStringField;

ADOqrytestergmae_alu: TWideStringField;

ADOqrytestecpfmae_alu: TWideStringField;

ADOqrytestepai_alu: TWideStringField;

ADOqrytesteprofpai_alu: TWideStringField;

ADOqrytestergpai_alu: TWideStringField;

ADOqrytestecpfpai_alu: TWideStringField;

ADOqrytesterenda_alu: TIntegerField;
ADOqrytestecsap_alu: TWideStringField;
ADOqrytestevlalug_alu: TIntegerField;
ADOqrytesteqt_resid_alu: TIntegerField;
ADOqrytesteconhece_alu: TWideStringField;
ADOqrytesteestuda_alu: TWideStringField;
ADOqrytestetpesc_alu: TWideStringField;
ADOqrytestenomeesc_alu: TWideStringField;
ADOqrytesteserieesc_alu: TWideStringField;
ADOqrytesteendesc_alu: TWideStringField;
ADOqrytestebairroesc_alu: TWideStringField;
ADOqrytestecepesc_alu: TWideStringField;
ADOqrytestecidesc_alu: TWideStringField;
ADOqrytesteufesc_alu: TWideStringField;
ADOqrytestedtcad_alu: TDateTimeField;
ADOqrytestecodigo_vol: TIntegerField;
ADOqrytesteStatus: TWideStringField;
ADOqrytestetb_fnalunocodigo_alu: TIntegerField;
ADOqrytestetptel_fnalu: TWideStringField;
ADOqrytestefone_fnalu: TWideStringField;
ppReport2: TppReport;
ppHeaderBand2: TppHeaderBand;

ppShape2: TppShape;
ppLabel12: TppLabel;
ppLabel13: TppLabel;
ppLabel14: TppLabel;
ppLabel15: TppLabel;
ppLabel16: TppLabel;
ppLabel17: TppLabel;
ppLabel18: TppLabel;
ppSystemVariable3: TppSystemVariable;
ppLabel19: TppLabel;
ppLabel20: TppLabel;
ppLine3: TppLine;
ppDetailBand2: TppDetailBand;
ppDBText9: TppDBText;
ppDBText10: TppDBText;
ppDBText11: TppDBText;
ppDBText12: TppDBText;
ppDBText13: TppDBText;
ppDBText14: TppDBText;
ppDBText15: TppDBText;
ppDBText16: TppDBText;
ppFooterBand2: TppFooterBand;

```

ppLine4: TppLine;

ppLabel21: TppLabel;

ppSystemVariable4: TppSystemVariable;

ppLabel22: TppLabel;

DBComboBox1: TDBComboBox;

DBRadioGroup2: TDBRadioGroup;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure bloqletrarg(Sender: TObject; var Key: Char);

procedure bloqletra(Sender: TObject; var Key: Char);

procedure FormCreate(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

```

```

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmLocador: TFrmLocador;

implementation

uses UDMeclipse, API_F11, UpesqGeral, Uatualizar, Math, UPrincipal,

    UPrincipal2;

{$R *.dfm}

procedure TFrmLocador.FormClose(Sender: TObject; var Action: TCloseAction);
begin

    dmeclipse.TBlocador.active:=false;

    DMeclipse.TBfnlocador.active:=false;

    DMeclipse.TBtpfone.active:=false;

    if menup='1900' then

        begin

            frmprincipal2.Enabled:=true;

        end

    else

        begin

            application.CreateForm(TFrmPrincipal,FrmPrincipal);

```

```

    FrmPrincipal.Show;

end;

action:= cafree;

FrmLocador:= nil;

end;

procedure TFrmLocador.FormKeyPress(Sender: TObject; var Key: Char);
begin
    if(Key = #13)then
    begin
        Key := #13;

        SelectNext(ActiveControl,true,true);

    end;

end;

procedure TFrmLocador.BitBtn1Click(Sender: TObject);
begin
    botoes(self);

    mudaedit(self, true);

    DMeclipse.TBlocador.Insert;

    DMeclipse.TBlocadorBlock_loc.AsString:='n';

end;

procedure TFrmLocador.BitBtn2Click(Sender: TObject);
begin

```

```

if not(DMeclipse.TBlocador.IsEmpty)then

begin

    botoes(self);

    mudaedit(self, true);

    DMeclipse.TBlocador.Edit;

end

else

    ShowMessage('Não existe registro para edição.');
```

```

end;

procedure TFrmLocador.BitBtn3Click(Sender: TObject);

begin

    if dmeclipse.TBlocador.isempty then

begin

    showmessage('Não existe registro a ser excluído');

    exit;

end;

    If messagedlg('Tem certeza que deseja excluir esse curso?'+#13+'Essa operação é
irreversível!', mtconfirmation, [mbytes, mbno], 0) = mrno then

        exit;

    DMeclipse.TBlocador.Delete;

end;

procedure TFrmLocador.BitBtn4Click(Sender: TObject);

begin
```

```

botoes(self);

mudaedit(self, false);

DMeclipse.TBlocador.Cancel;

end;

procedure TFrmLocador.BitBtn5Click(Sender: TObject);

begin

    trims(self);

    if (dbedit2.text="")or(DBRadioGroup1.Value="")or((DBEdit4.text="")and(DBEdit5.text="))or(DB
Edit6.text="")or(dbedit7.text="")or(DBComboBox1.text=") then

    begin

        showmessage('Preencha todos os campos requeridos');

        exit;

    end;

    if (cpf(dbedit5.Text)=false)and(dbedit5.text<>") then

    begin

        showmessage('CPF Inválido');

        DBEdit5.SetFocus;

        exit;

    end;

    botoes(self);

    mudaedit(self, false);

    DMeclipse.TBlocador.Post;

```

```

    DMeclipse.TBlocador.Refresh;

end;

procedure TFrmLocador.BitBtn6Click(Sender: TObject);

begin

    Close;

end;

procedure TFrmLocador.BitBtn7Click(Sender: TObject);

begin

    String_tb.Stabela := 'Locador';

    String_tb.Scampocod := 'Codigo_loc';

    String_tb.Scamponome := 'Nome_Loc';

    String_tb.Qrytb := DMeclipse.TBlocador;

    String_tb.DStb := DMeclipse.DStblocador;

    abrirpesq(self);

    String_tb.resultcodi := DMeclipse.TBlocadorcodigo_loc.AsInteger;

    { DMeclipse.TBlocador.Close;

    DMeclipse.TBlocador.SQL.Clear;

    DMeclipse.TBlocador.SQL.Add('SELECT * FROM TB_LOCADOR');

    DMeclipse.TBlocador.Open;    }

end;

procedure TFrmLocador.Button1Click(Sender: TObject);

begin

```



```

If not(DMeclipse.TBlocador.IsEmpty)then

begin

    ppReport1.Print;

end;

end;

procedure TFrmLocador.Button2Click(Sender: TObject);

begin

    If not(ADOqryteste.IsEmpty)then

begin

    ppReport2.Print;

end;

// else

// begin

//  ShowMessage('teste');

// end;

end;

procedure TFrmLocador.bloqletrarg(Sender: TObject; var Key: Char);

begin

    if ((key<#48)or(key>#57))and(key<>#8)and(key<>#120)and (key<>#88)then

        key:=#0;

end;

procedure TFrmLocador.bloqletra(Sender: TObject; var Key: Char);

```

```

begin

    if ((key<#48)or(key>#57))and(key<>#8) then

        key:=#0;

    end;

procedure TFrmLocador.FormCreate(Sender: TObject);

begin

    DMeclipse.TBlocador.active:=true;

    DMeclipse.TBfnlocador.active:=true;

    DMeclipse.TBtpfone.active:=true;

end;

procedure TFrmLocador.FormCloseQuery(Sender: TObject;

    var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de aluno?',

mtconfirmation, [mbyes, mbno], 0) = mryes then

        begin

            DMeclipse.TBlocador.cancel;

            DMeclipse.TBfnlocador.cancel;

            dmeclipse.TBtpfone.cancel;

            canclose:=true;

        end

    else

        canclose:=false;

```

```

end;

end.

unit ULocador;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, ExtCtrls, DBCtrls, Mask;

type

    TFrmLocador = class(TForm)

        Label1: TLabel;

        DBEdit1: TDBEdit;

        Label2: TLabel;

        DBEdit2: TDBEdit;

        Label4: TLabel;

        DBEdit4: TDBEdit;

        Label5: TLabel;

        DBEdit5: TDBEdit;

        Label6: TLabel;

        DBEdit6: TDBEdit;

        Label7: TLabel;

        DBEdit7: TDBEdit;

        Label8: TLabel;

```

```

Label9: TLabel;

DBEdit9: TDBEdit;

Label10: TLabel;

DBEdit10: TDBEdit;

DBRadioGroup1: TDBRadioGroup;

DBEdit8: TDBEdit;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormKeyPress(Sender: TObject; var Key: Char);

private

    { Private declarations }

public

    { Public declarations }

end;

var

    FrmLocador: TFrmLocador;

implementation

uses UDMeclipse;

{$R *.dfm}

procedure TFrmLocador.FormClose(Sender: TObject; var Action: TCloseAction);

begin

    action:= cafree;

    FrmLocador:= nil;

```

```

end;

procedure TFrmLocador.FormKeyPress(Sender: TObject; var Key: Char);
begin
    if(Key = #13)then
    begin
        Key := #13;

        SelectNext(ActiveControl,true,true);
    end;
end;

end.

unit ULocLivro;

interface

uses

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Buttons, ExtCtrls, DBCtrls, Mask,db;

type

    TFrmLocLivro = class(TForm)

        Panel2: TPanel;

        BitBtn1: TBitBtn;

        BitBtn2: TBitBtn;

        BitBtn3: TBitBtn;

        BitBtn4: TBitBtn;

```

BitBtn5: TBitBtn;

BitBtn6: TBitBtn;

BitBtn7: TBitBtn;

Label1: TLabel;

Label3: TLabel;

DBEdit3: TDBEdit;

Label4: TLabel;

DBEdit4: TDBEdit;

Label5: TLabel;

DBEdit5: TDBEdit;

Label6: TLabel;

Label7: TLabel;

DBMemo1: TDBMemo;

Label8: TLabel;

DBEdit7: TDBEdit;

Label2: TLabel;

DBComboBox1: TDBComboBox;

DBText1: TDBText;

DBEdit1: TDBEdit;

BitBtn8: TBitBtn;

BitBtn9: TBitBtn;

Label9: TLabel;

```

Label10: TLabel;

DBText2: TDBText;

DBText3: TDBText;

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure BitBtn7Click(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure DBLookupComboBox1Exit(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure habilitarpesq;

procedure desabilitarpesq;

procedure FormCreate(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

private

    { Private declarations }

public

    { Public declarations }

```

```

end;

var

  FrmLocLivro: TFrmLocLivro;

implementation

uses UDMeclipse, API_F11, UCadAluno, UpesqGeral, UPrincipal, UPrincipal2;

{$R *.dfm}

procedure TFrmLocLivro.BitBtn1Click(Sender: TObject);

begin

  botoes(self);

  mudaedit(self, true);

  habilitarpesq;

  DMeclipse.TBlocacao.Insert;

  DMeclipse.TBlocacaopendencia_locac.AsString:='Sim';

end;

procedure TFrmLocLivro.BitBtn2Click(Sender: TObject);

begin

if not(DMeclipse.TBcargo.IsEmpty)then

begin

  botoes(self);

  mudaedit(self, true);

  habilitarpesq;

```



```

    DMeclipse.TBlocacao.Edit;

end

else

    ShowMessage('Não existe registro para edição.');
```

end;

```

procedure TFrmLocLivro.BitBtn3Click(Sender: TObject);

begin

    if dmeclipse.TBlocacao.isempty then

        begin

            showmessage('Não existe registro a ser excluído');
```

exit;

```

        end;

        If messagedlg('Tem certeza que deseja excluir essa locação?'+#13+'Essa operação
é irreversível!', mtconfirmation, [mbyes, mbno], 0) = mrno then

            exit;

            DMeclipse.TBlocacao.delete;

        end;

    procedure TFrmLocLivro.BitBtn4Click(Sender: TObject);

    begin

        botoes(self);

        mudaedit(self, false);

        desabilitarpesq;

        DMeclipse.TBlocacao.Cancel;
```

```

end;

procedure TFrmLocLivro.BitBtn5Click(Sender: TObject);
begin
    trims(self);

    if (trim(dbedit7.text='')or(dbedit1.text='')or(data(DBEdit3.text)=false) then
    begin
        showmessage('Preencha todos os campos requeridos');

        exit;
    end;

    botoes(self);

    mudaedit(self, false);

    desabilitarpesq;

    try

        DMeclipse.TBlocacao.Post;

        DMeclipse.TBlocacao.Refresh;

    except

        on e: edatabaseerror do

            showmessage(e.Message);

    end;

end;

procedure TFrmLocLivro.BitBtn7Click(Sender: TObject);
begin

```

```

String_tb.Stabela := 'locacao';

String_tb.Scampocod := 'tombo_liv';

String_tb.Scamponome := 'pendencia_locac';

String_tb.Qrytb := DMeclipse.TBlocacao;

String_tb.DStb := DMeclipse.DStblocacao;

abrirpesq(self);

end;

procedure TFrmLocLivro.BitBtn6Click(Sender: TObject);

begin

    Close;

end;

procedure TFrmLocLivro.FormClose(Sender: TObject;

    var Action: TCloseAction);

begin

    DMeclipse.TBlocador.active:=true;

    dmeclipse.TBlivro.Active:=true;

    dmeclipse.TBlocacao.Active:=true;

    if menup='1900' then

    begin

        frmprincipal2.Enabled:=true;

    end

    else

```

```

begin

    application.CreateForm(TFrmPrincipal,FrmPrincipal);

    FrmPrincipal.Show;

end;

action:= caFree;

FrmLocLivro:= nil;

end;

procedure TFrmLocLivro.DBLookupComboBox1Exit(Sender: TObject);

begin

    if dmeclipse.TBlocador.FieldName('Block_loc').AsString='sim' then

        begin

            showmessage('Locador com pendências');

            dbedit1.SetFocus;

        end;

    end;

end;

procedure TFrmLocLivro.Button1Click(Sender: TObject);

begin

    String_tb.Stabela := 'Livro';

    String_tb.Scampocod := 'tombo_liv';

    String_tb.Scamponome := 'Nome_liv';

    String_tb.Qrytb := DMeclipse.TBlivro;

    String_tb.DStb := DMeclipse.DStblivro;

```

```

    abrirpesq(self);

    // DBEdit7.Text:=DMeclipse.TBlivrotombo_liv.AsString;

end;

procedure TFrmLocLivro.Button2Click(Sender: TObject);
begin
    String_tb.Stabela := 'Locador';

    String_tb.Scampocod := 'codigo_loc';

    String_tb.Scamponome := 'Nome_loc';

    String_tb.Qrytb := DMeclipse.TBlocador;

    String_tb.DStb := DMeclipse.DStblocador;

    abrirpesq(self);

    DBEdit1.Text:=DMeclipse.TBlocadorcodigo_loc.AsString;

end;


procedure TFrmLocLivro.desabilitarpesq;

begin

    BitBtn8.Enabled:=false;

    bitbtn9.Enabled:=false;

end;

procedure TFrmLocLivro.habilitarpesq;

begin

    BitBtn8.Enabled:=true;

```

```

    bitbtn9.Enabled:=true;

end;

procedure TFrmLocLivro.FormCreate(Sender: TObject);

begin

    DMeclipse.TBlocador.active:=true;

    dmeclipse.TBlivro.Active:=true;

    dmeclipse.TBlocacao.Active:=true;

end;

procedure TFrmLocLivro.FormCloseQuery(Sender: TObject;

    var CanClose: Boolean);

begin

    If messagedlg('Tem certeza que deseja fechar o formulário de cadastro de aluno?',

mtconfirmation, [mbyes, mbno], 0) = mryes then

    begin

        dmeclipse.TBlocacao.cancel;

        canclose:=true;

    end

    else

        canclose:=false;

end;

end.

unit Usobre;

interface

```

uses Windows, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
Buttons, ExtCtrls, jpeg, ComCtrls;

type

```
TFrmSobre = class(TForm)
    Panel1: TPanel;
    ProductName: TLabel;
    Copyright: TLabel;
    Comments: TLabel;
    OKButton: TButton;
    Label3: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Image1: TImage;
    Bevel1: TBevel;
    StatusBar1: TStatusBar;
    procedure FormShow(Sender: TObject);
    procedure OKButtonClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
    { Private declarations }
public
    { Public declarations }
end;
```

var

```
FrmSobre: TFrmSobre;
```

implementation

uses Uatualizar, UPrincipal2, UPrincipal, API_F11;

{ \$R *.dfm }

```
procedure TFrmSobre.FormShow(Sender: TObject);
```

var

```
    dateT : TDateTime;
```

begin

```
    StatusBar1.Panels[0].Text := 'Versão: ' + vers.versao;
```

```

    StatusBar1.Panels[1].Text := 'Data: ' + DateToStr(dateT);
end;
procedure TFrmSobre.OKButtonClick(Sender: TObject);
begin
    close;
end;
procedure TFrmSobre.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    if menup='1900' then
    begin
        frmprincipal2.Enabled:=true;
    end
    else
    begin
        application.CreateForm(TFrmPrincipal,FrmPrincipal);
        FrmPrincipal.Show;
    end;
    action:= cafree;
    FrmSobre:= nil;
end.

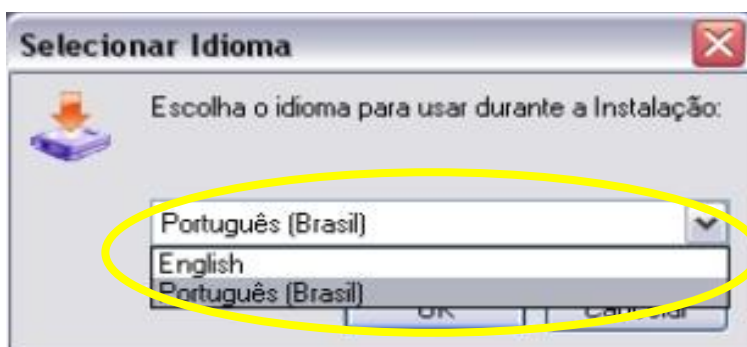
```


23. Manual Sistema Eclipse

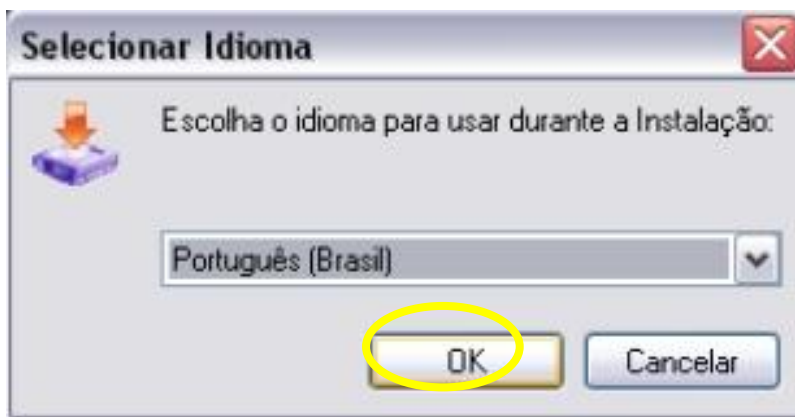
- Dê um clique duplo no ícone do instalador,



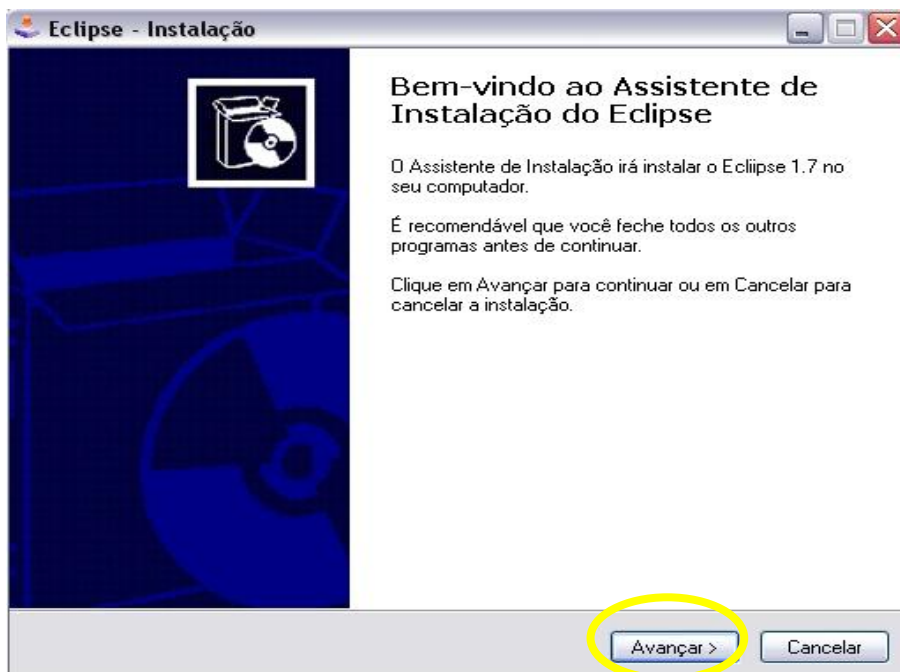
- Logo após ele lhe trará uma tela para que escolha o idioma,



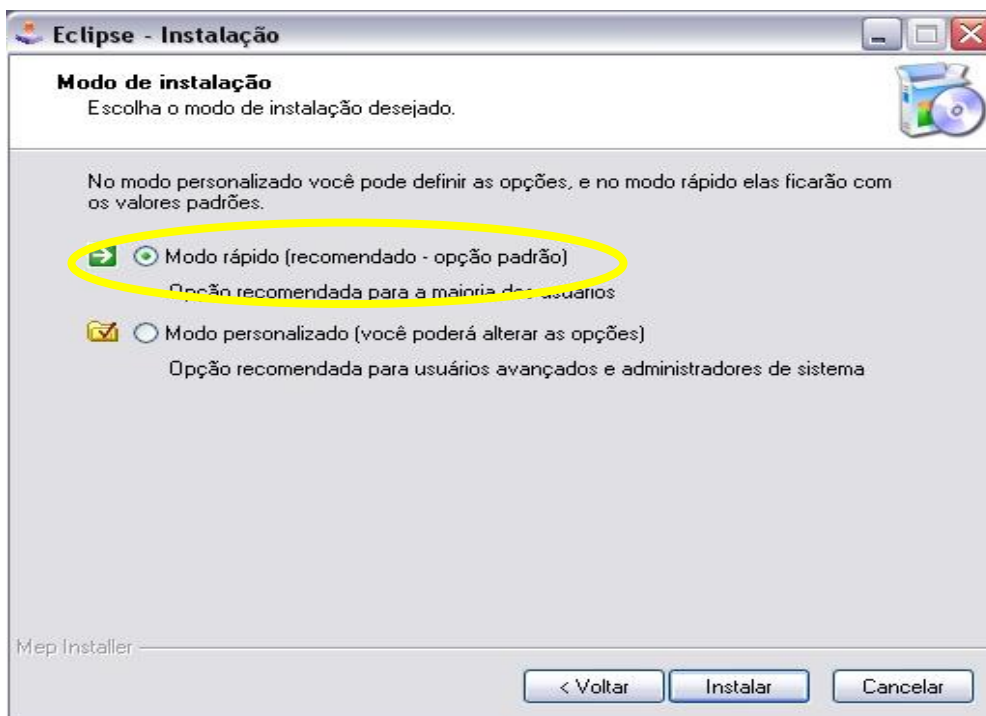
- Escolhido o Idioma Clique em Ok



- Dê um Avançar



- Escolha o Modo de Instalação (é indicado o Modo Clássico para a 1ª Instalação)



- Clique em Instalar



- Aguarde o processo decorrente da instalação



- Clique em Concluir



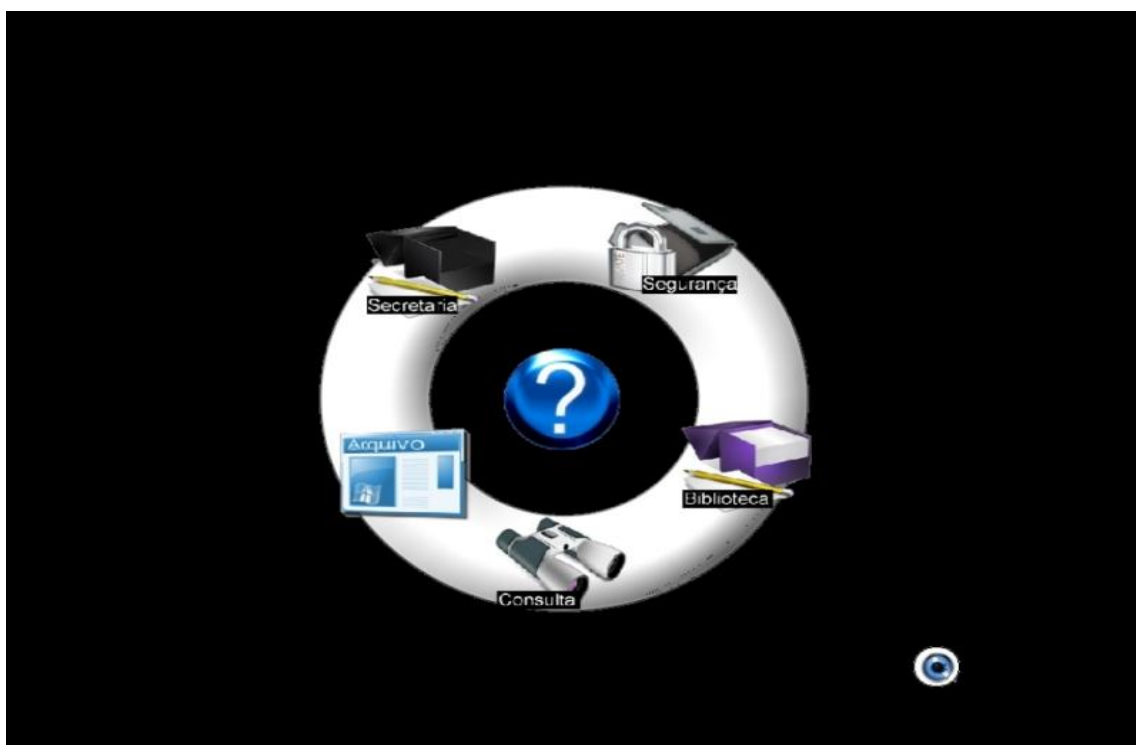
- Ao clicar em concluir automaticamente o Sistema Eclipse será aberto



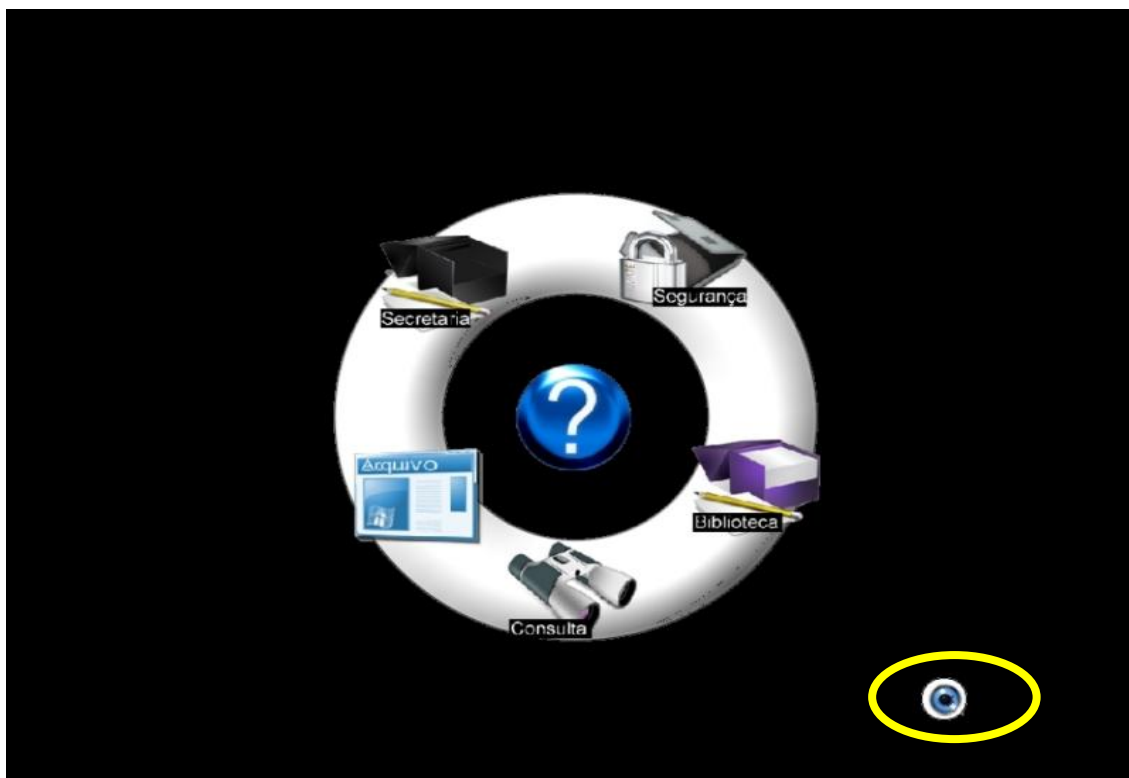
Digite o Usuário e a senha (O usuário inicial é Admin e a senha é Eclipse1)



- Após Logado o sistema trará a você a tela de Menu com visual compacto



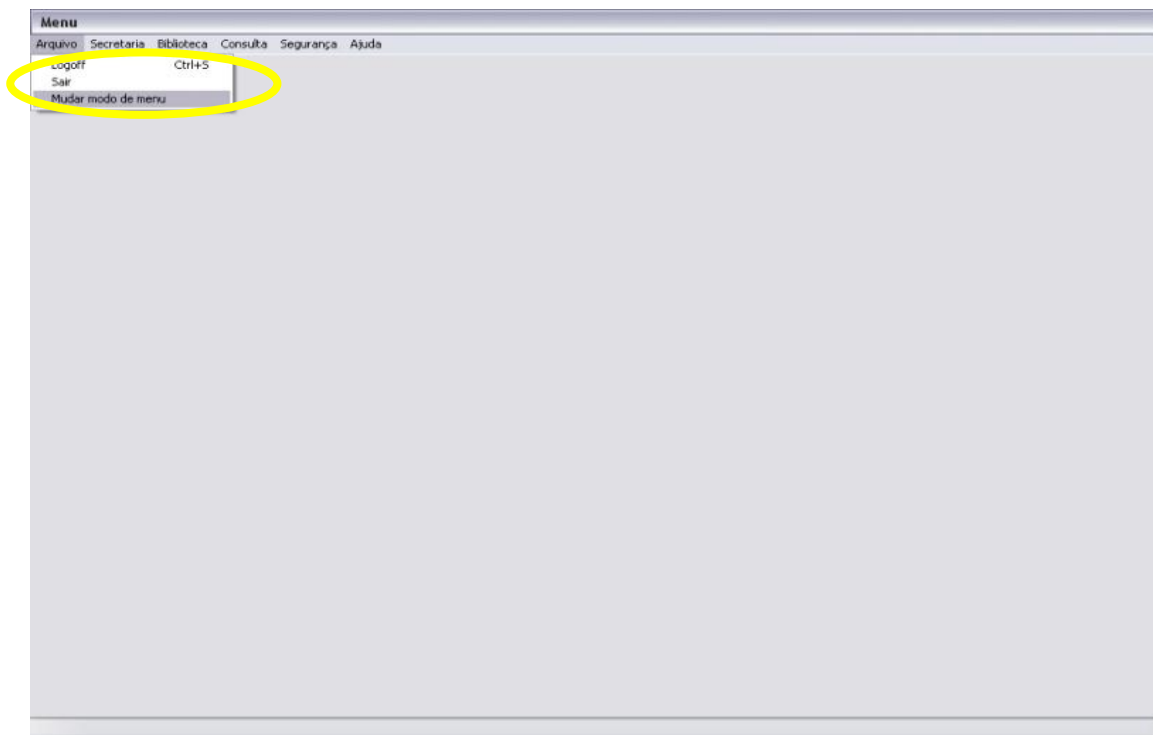
- Para alterná-la basta clicar no olho que se encontra a direita da área de trabalho



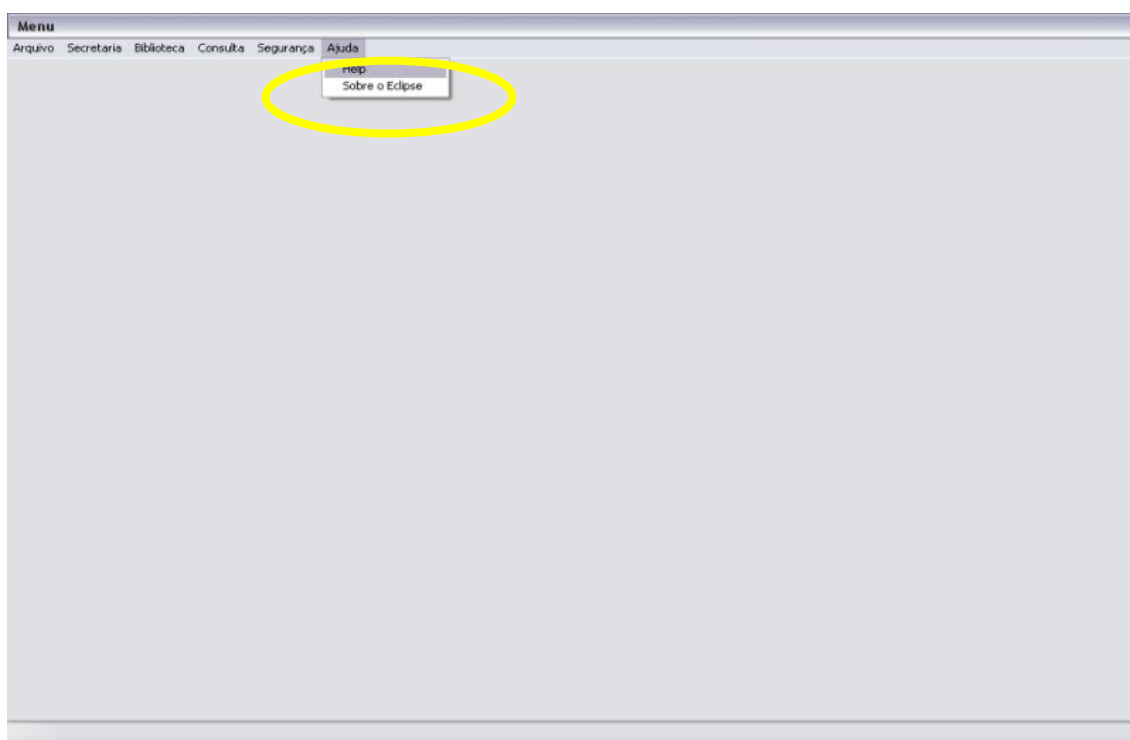
- Então abrirá a você o menu clássico



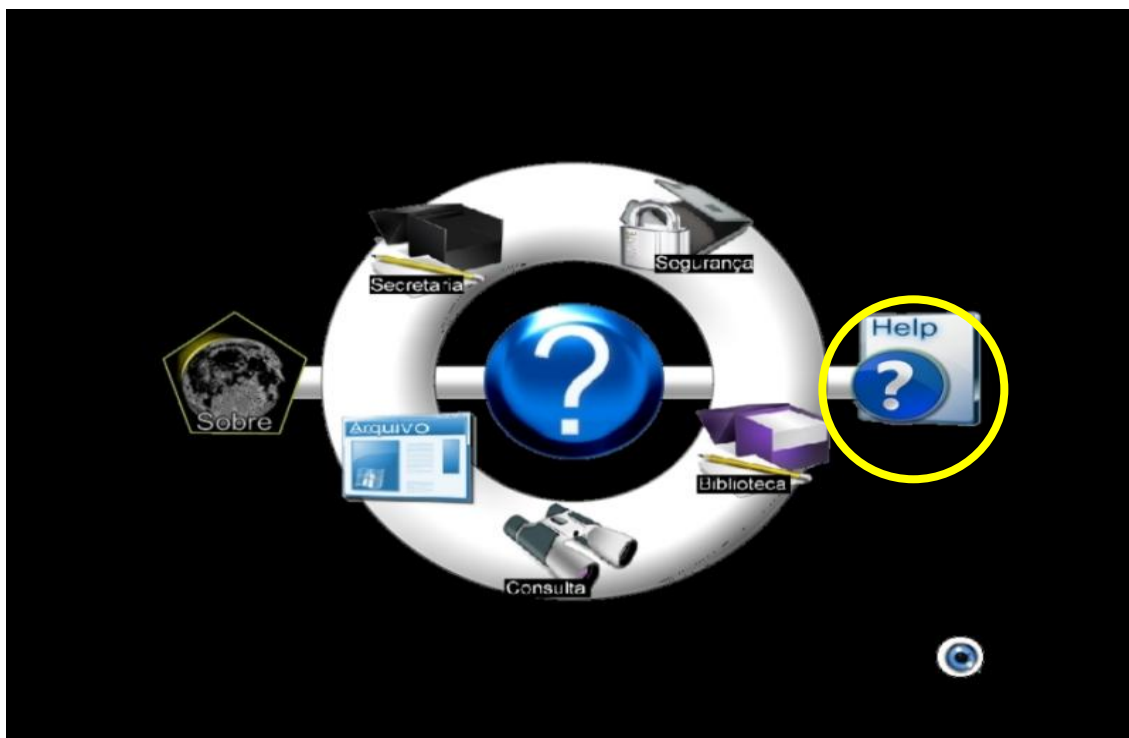
- Para voltar ao menu compacto basta clicar em Arquivo e a opção Mudar Menu



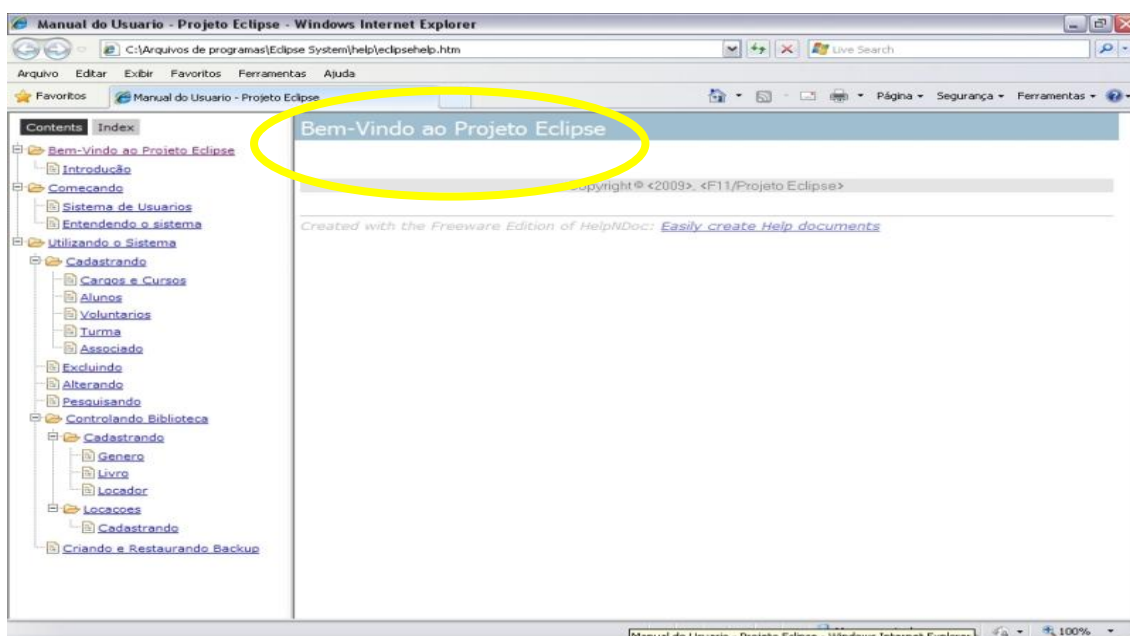
- No menu Clássico você pode acessar o Help que é o Manual do sistema eclipse mais detalhado ao usuário. Através da guia Ajuda e clique na opção Help



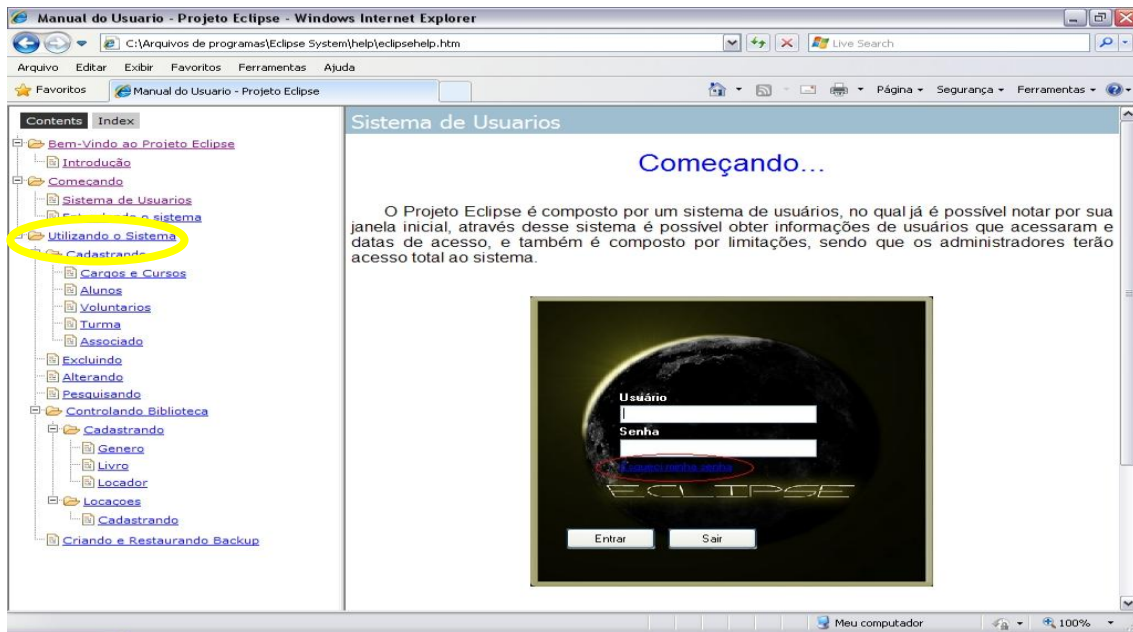
- Ou se preferir clique no símbolo de interrogação e logo após na opção Help



- Logo depois de clicado em um dos dois modos compacto ou clássico o Sistema lhe trará o manual online, com a tela de boas vindas.



- Clique no menu de opções lado esquerdo em Começando e obtenha as instruções de como utilizar o Sistema Eclipse



- Agora você pode usufruir as facilidades que o Sistema Eclipse proporciona a você

24. Manutenção

O período de manutenção oferecido pelo grupo F11 para o Espaço Cultural Flávio Fernandes de Matos é de 3 (três) meses, visando o conceito para aceitação do usuário final, em sua adaptação com metas de expansão para o software de modo que obtenha a total aceitação ao cliente.

25. Considerações Finais

Nosso projeto, que visa solucionar os problemas de armazenamento de dados e relatórios do Espaço Cultural Flávio Fernandes de Matos, foi terminado com sucesso. Nele está contido todo nosso esforço e conhecimento profissional para executá-lo de maneira conveniente.

Esperamos que este software seja de importância para as atividades de nosso cliente. Que torne os trabalhos eficientes, rápidos e seguros, de maneira que o andamento da empresa não seja afetado por qualquer problema que o método de trabalho antigo ocasionava.

Tendo a certeza de que foi implantado nesse projeto o essencial para o cliente, concluímos que este é o nosso melhor para o Espaço Cultural Flávio Fernandes de Matos. E nosso esforço ocasionou satisfação, pois adquirimos conhecimento, ao decorrer deste projeto.

26. Referências Bibliográficas

Apostilas

www.creupiapostilas.hpg.ig.com.br/analise_estruturada.doc 09/06/2009 às 09:16 am

<http://www.prodepa.gov.br/sbc2008/anais/pdf/arq0112.pdf> 07/06/2009 às 17:35 pm

www.inf.ufrgs.br/aulas/mlp/slides/MLPIntro1.ppt 30/05/2009 às 00:26 am

Internet

<http://support.microsoft.com/kb/208312/pt-br> 27/05/2009 às 23:00 pm

<http://www.pedagogiaemfoco.pro.br/met01.htm> 08/6/2009 às 14:30 pm

<http://br.geocities.com/perseuscm/espirtocientifico.html> 08/06/2009 às 14:50 pm

pt.tech-faq.com/data-dictionary.shtml 13/06/2009 às 14: 25 pm

[http://www.infopedia.pt/\\$protocolos-de-comunicacao](http://www.infopedia.pt/$protocolos-de-comunicacao) 14/06/2009 às 12:35 pm

http://www.mephost.com/software/mep_installer.htm 16/06/2009 às 00:22 am

<http://pt.wikipedia.org/wiki/Metodologia> 08/06/2009 às 8:20 am

<http://wiki.di.uminho.pt/twiki/bin/view/Education/ACS/WebHome> 08/06/2009 às 9:00 am

<http://www.infoescola.com/informatica/o-que-sao-linguagens-de-programacao>
/09/06/2009 às 16:00 pm

http://pt.wikipedia.org/wiki/Teste_de_software 05/06/2009 às 8:35 am

http://imasters.uol.com.br/artigo/9572/des_de_software/teste_de_software
04/06/2009 às 7:20 am

<http://www.xtibia.com/forum/e-Delphi-Historia-Delphi-t42588.html> 04/06/2009 às 9:35 am

<http://www.xtibia.com.br/xsite/component/xcontent/?task=post&id=249970>
04/06/2009 às 10:23 am

http://www.macoratti.net/proc_sw1.htm 06/06/2009 às 10:44 am

http://www.macoratti.net/vb_dfd1.htm 06/06/2009 às 11:23 am

http://www.wmtec.com.br/projeto_b2b/uml/homepage.html 06/06/2009 às 13:54 am

<http://www.scribd.com/doc/14853869/Analise-Estruturada-x-Orinetada-a-Objetos>
06/06/2009 às 14:22

<http://www.guiacarapicuiba.com/content/view/134/118/> 15/06/2009 às 7:45 am

Livros

The effective executive Por Peter F. Drucker

Edição: 2, revisada Publicado por Butterworth-Heinemann, 2007

Jalote, P. (1997), An integrated approach to software engineering, Springer, 2a. edição.

Alberto Heuser, Carlos (2001) Projeto de Banco de Dados, Sagra edição:1.

27. Anexos

27.1. Ata de Compromisso

Ata de Compromisso

O Espaço Cultural Flávio Fernandes de Matos concorda com a informatização do sistema de arquivos e banco de dados elaborados e executados pelo grupo F11. Este projeto não houve fins lucrativos para ambas as partes. Sendo que o período de assistência dado ao software será de 3 (três) meses.

São Paulo, _____ de _____ de 2009

Fernando Gomes Paredes
Assinatura Coordenador F11

Antônio Fernandes de Matos
Representante Espaço Cultural Flávio Fernandes de Matos