MARIA CAROLINA R. M. DA CUNHA

AUTORIA EM HIPERMÍDIA: O MODELO OOHDM APLICADO AO ENSINO DE LINGUAGENS DE PROGRAMAÇÃO

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências da disciplina Projeto Orientado, para a obtenção do título de Bacharel de Ciência da Computação.

Orientador

Prof. André Luiz Zambalde

LAVRAS
MINAS GERAIS – BRASIL
2002

MARIA CAROLINA R. M. DA CUNHA

AUTORIA EM HIPERMÍDIA: O MODELO OOHDM APLICADO AO ENSINO DE LINGUAGENS DE PROGRAMAÇÃO

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências da disciplina Projeto Orientado, para a obtenção do título de Bacharel de Ciência da Computação.

Aprovada em 27 de Março de 2002.

_____
Profa. Olinda Nogueira Paes Cardoso

_____
Prof. André Luiz Zambalde
(orientador)

Lavras
Minas Gerais - Brasil

Dedico este trabalho a mim.

AGRADECIMENTOS

A Deus, por tudo, ao orientador André Zambalde, pela paciência e ajuda neste trabalho. Agradeço também a minha mãe e irmãos pelo apoio e por suportarem meu eterno mau humor.

RESUMO

O presente trabalho descreve o processo de modelagem e implementação de uma aplicação hipermídia para o ensino de linguagem de programação. São apresentados os principais conceitos relativos à hipermídia. Realizou-se a modelagem da aplicação utilizando-se o OOHDM – "*Object Oriented Hypermidia Design Model*" e implementou-se um exemplo utilizando-se o software Toolbook. Concluiu-se que o modelo OOHDM, independente do processo de implementação, é capaz de estruturar complexas aplicações, facilitando a manutenção e reutilização.

Palavras-chave: modelagem OOHDM, ensino, programação, sistemas hipermídia.

ABSTRACT

This work describes a hypermidia application's process of modeling and implementing for programing language learning. The main concepts related to the hypermidia are presented. The application's modeling was done using the OOHDM – "Object Oriented Hypermidia Design Model". An exemple of implementation was built with the software Toolbook. It was concluded thah the model OOHDM, independently of the implementation's process, was able to structure complex applications, facilitating the maintenance and giving support to the reuse.

Key Words: *OOHDM modeling, learning, programing, hypermidia systems.*

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO I – INTRODUÇÃO

1.1 Considerações iniciais

Projetos que visam estimular o uso dos computadores como ferramenta de apoio à difusão do conhecimento e ao ensino através de sistemas hipermídia (mídias digitais e *internet*) são desenvolvidos em todo o mundo. A maioria desses projetos alcança aplicações em diversas áreas de conhecimento, tais como informática, medicina, história, geografia, matemática, português, inglês, biologia, entre outras [ZAL96]. O fato é que, a cada dia surgem novas áreas de aplicação que utilizam a flexibilidade dos hipertextos combinada com a riqueza dos tipos de dados da multimídia.

No entanto, é tarefa difícil construir grandes aplicações hipermídia. Geralmente, elas apresentam manutenção complicada e são construídas sem a preocupação fundamental com o reuso [Lim94].

Para desenvolver um sistema hipermídia que seja de qualidade e atenda aos requisitos básicos de manutenção e reuso é preciso que se busque um modelo de autoria que possa descrever toda a estrutura da aplicação hipermídia e sua semântica de navegação de uma forma independente da implementação. Este sistema deve fornecer construções de *design* de alto nível e mecanismos de abstração, bem como permitir uma transição tranqüila entre a abstração do domínio da aplicação e a implementação final, preenchendo assim, o espaço entre os modernos mecanismos de engenharia de *software* e as peculiaridades da hipermídia. Ele ainda deve ser capaz de fornecer meios para que se faça alterações no modelo para uma aplicação final (e vice-versa), tornando a manutenção e o reuso mais fácil.

Finalmente, ele deve ser independente do ambiente, isto é, deve possibilitar a construção de modelos abstratos para que se possa implementá-los em diferentes plataformas hipermídia [Lim94].

1.2 Autoria hipermídia e educação

A utilização da tecnologia hipermídia na educação é incentivada pela sua característica de permitir ao aprendiz a exploração livre de páginas com informações representadas por diversas mídias estruturadas através de ligações. Uma aplicação hipermídia (um hiperdocumento) fornece o material didático e proporciona uma forma de navegação através dele, com o controle da interação totalmente a cargo do usuário, permitindo que este tenha progresso de acordo com os seus interesses e objetivos. A característica pedagógica mais importante dessas aplicações é, portanto, a flexibilidade de exploração do material didático fornecido.

Entretanto, deve-se considerar que a hipermídia não foi projetada para a educação. O principal foco desta tecnologia tem sido a recuperação eficiente de informação e o entretenimento. Portanto, é necessário distinguir entre dois tipos de aplicações hipermídia [THH95]: as destinadas à pesquisa, recuperação e difusão de informações e as destinadas a tarefas relacionadas ao ensino e aprendizagem. As aplicações do primeiro tipo apresentam-se como base de dados que podem ser exploradas livremente por um leitor. Em contraste, as aplicações do segundo tipo têm o formato de documentos eletrônicos que procuram guiar intencionalmente os leitores através de um espaço de informações sobre um domínio de conhecimento.

Existe uma longa tradição nos projetos tradicionais, baseados em papel, mas existe pouca ou nenhuma para o projeto de hiperdocumentos [Str94]. O sucesso ou o fracasso da interação do leitor com um hiperdocumento é determinado pelas decisões feitas pelo autor sobre quais documentos devem ser unidos por ligações. Por isso, mais uma vez é enfatizada a necessidade da utilização de um modelo de autoria, para que o sistema hipermídia possa corresponder às expectativas envolvidas em seu projeto.

1.3 O modelo OOHDM

O OOHDM – *Object Oriented Hypermidia Design Model* – é um modelo de autoria que fornece mecanismos para a descrição das relações conceituais entre objetos do domínio, além de definir suas estruturas e comportamentos. Combinando classes com instâncias específicas de certas aplicações consegue-se preservar o poder de abstração sem perder a flexibilidade. O modelo combina as já conhecidas construções (classes, objetos) e os mecanismos de abstração (agregação, herança) da análise orientada a objeto com conceitos úteis da hipermídia (estruturas, hierarquias, perspectivas, contextos navegacionais, etc). É um modelo muito utilizado devido a duas de suas principais características: facilidade de manutenção e reutilização (reuso) [Lim94].

1.4 Objetivos do trabalho

O objetivo deste trabalho é o estudo do Modelo de Projeto de Aplicações Hipermídia Orientado a Objeto (OOHDM), e da metodologia associada, proposta por Daniel Schwabe e por Gustavo Rossi [SR94].

Pretende-se apresentar o processo de desenvolvimento de uma interface baseada no OOHDM. Este processo deverá envolver a modelagem de uma aplicação educacional direcionada ao ensino de linguagens de programação, seguida do projeto de navegação, *design* abstrato da interface (genérico) e da implementação fundamentada na linguagem DELPHI.

Entende-se que usando objetos não só como artefatos de implementação, mas principalmente como ferramentas de modelagem durante o processo de desenvolvimento de aplicações hipermídia, consegue-se solucionar os problemas de reuso e manutenção.

1.5 Estrutura do trabalho

No Capítulo I (Introdução), são apresentadas as considerações básicas e definição dos objetivos e composição do trabalho. No capítulo II (Referencial Teórico) realiza-se uma cuidadosa pesquisa bibliográfica sobre hipermídia, multimídia e hipertexto, além de um levantamento de informações sobre o OOHDM: seu histórico, características, princípios de funcionamento e descrição de sua utilização atual.

No Capítulo III (Metodologia) pretende-se especificar os elementos e ferramentas a serem utilizados do desenvolvimento deste trabalho, tais como ambiente de trabalho, sistemas de *hardware* e *software*.

Para se ter uma idéia real do modelo de aplicação, no Capítulo IV (Resultados e Discussão) apresenta-se todo o processo de desenvolvimento de uma aplicação hipermídia utilizando o modelo OOHDM, ou seja, a modelagem da aplicação, o projeto de navegação, o projeto abstrato da interface da aplicação, utilizando-se uma ferramenta de autoria hipermídia (Toolbook). Na seqüência, implementa-se um projeto específico, direcionado a área de ensino de uma linguagem de programação (DELPHI).

Como o método é sistemático e gradual, cada etapa do processo constrói a base para o desenvolvimento da etapa posterior. Assim, ao final do projeto será implementado um hiperdocumento coerente com suas características conceituais e de navegação, definidas nas diversas etapas e dentro dos padrões de manutenção e reuso desejáveis às aplicações hipermídia.

No Capítulo V (Conclusão), pretende-se apresentar as principais observações sobre o modelo OOHDM e suas aplicações no contexto hipermídia, com alguns comentários sobre interface homem-máquina.

Por último, no Capítulo VI (Referências Bibliográficas), encontra-se toda a bibliografia utilizada neste trabalho.

CAPÍTULO II – REFERENCIAL TEÓRICO

2.1 Hipertexto e Hipermídia: conceitos básicos

Segundo Zambalde [ZAL96], hipermídia constitui a junção dos tipos de dados na multimídia, com os mecanismos e semânticas dos hipertextos (navegação não seqüencial), ou seja, um aplicativo hipertexto que, além de texto, gráficos e figuras, suporta outros tipos de mídia, como vídeo e som.

Schneiderman [Sch98] define hipermídia como uma rede de nós (também chamados artigos, documentos, arquivos, cartões, páginas, *frames* e telas) contendo informações (em texto, gráficos, vídeo, som e outros) que são conectados por *links* (também chamados ponteiros, referências cruzadas e citações).

A *internet* estendeu a hipermídia para uma vasta rede de computadores nos quais milhões de usuários podem criar e reaproveitar materiais hipermídia em segundos.

Segundo Schneiderman [Sch98], para se ter um aplicativo hipermídia de qualidade, os autores devem escolher projetos apropriados, visando organizar seus artigos convenientemente e ajustar o seu estilo de escrita. O primeiro passo para criar uma hipermídia efetiva é escolher projetos que sigam as "regras de ouro do hiperdocumento (*golden rules of hipertext*)" que são as seguintes:

- um grande conjunto de informações organizadas em numerosos fragmentos;
- os fragmentos devem se relacionar uns com os outros;
- o usuário deve necessitar somente de uma pequena fração de fragmentos em qualquer tempo.

Os principais conceitos relacionados às estruturas que compõe a hipermídia são os seguintes [Lim94]:

- Nós: parte do documento hipermídia ou de hipertexto que contém um trecho da informação que se está apresentando. A forma de como são estruturadas as idéias, isto é, se um dado assunto corresponde a um único nó ou deve ser dividido em vários nós, hieraquircamente ou não, depende exclusivamente do autor da aplicação.

  Geralmente associa-se uma janela exibida na tela do terminal a um nó. A possibilidade de rolamento de um nó que exceder o tamanho de uma janela ou a limitação de tamanho de um nó depende da plataforma e da implementação usadas.

  Nós podem não ter tipo associado e aceitar qualquer conteúdo ou podem ser tipados e ter seus conteúdos validados por regras de restrição ou dirigidos por moldes.

- Elos: os relacionamentos existentes entre dois trechos de informação do hiperdocumento são representados por elos. A representação de um elo em uma aplicação hipermídia ou de hipertexto é feita com a exibição de uma ou mais palavras marcadas, em outra cor ou em vídeo reverso.

  Basta clicar o botão do mouse ou pressionar uma tecla para se ativar um elo. A ativação de um elo provoca a abertura de uma janela contendo o nó de destino do relacionamento representado por ele. Geralmente esta ativação deve ser registrada através de algum *feedback* visual (por exemplo, invertendo o vídeo), e em muitos casos também sonoro (um *bip*). Esta janela pode se sobrepor à original (em cascata), ser justaposta a ela (lado a lado), ocupar a tela inteira, ser combinada com a original

(dando a impressão que a janela de origem “esticou”) e pode também aparecer numa janela tipo *pop-up*.

- Âncoras: a percepção dos elos é, usualmente feita através de âncoras (ou botões). As âncoras são marcadores que indicam ao leitor a existência de um (ou mais) elos, e podem ser ativadas através de alguma operação, normalmente um clique do mouse.

  Segundo Schwabe [Sch93], as âncoras podem ser indicadas de diversas formas: símbolos ou ícones específicos que servem como marcadores; vídeo reverso; caixas ao redor do texto ou bordas ao redor dos gráficos ou imagens; indicações tipográficas (negrito, itálico, cor etc); mudanças na forma do cursor; piscamento. Todos estes devem ser pensados em função da capacidade do leitor em diferenciar os diversos tipos e distingui-los na tela.

  A figura 1 mostra o uso de âncoras. Cada botão é uma âncora, que se clicado, ativa o elo que remete o usuário à informação desejada.

Figura 1 – Janela mostrando algumas âncoras. (Implementação feita em Toolbook)

- Navegar / Folhear: o que o sistema de hipertexto ou hipermídia tenta é dar ao leitor a sensação de manuseio de um livro, ou outro documento de papel, quando este manipula as janelas de uma aplicação. Portanto, uma das mais importantes operações neste tipo de sistema é o folheio ou *browsing* das páginas do hiperdocumento. Alguns sistemas, além de permitir o *browsing*, fornecem também ao leitor um mapa, na forma de um grafo, com a topologia da rede (também conhecido como diagrama). Com isso, o leitor pode navegar tanto pelo mapa quanto pelos elos, que levam o leitor de um nó a outro dentro da aplicação.

- Autoria em ponto grande: a preocupação aqui é com os diversos tipos de nós e elos, ou seja, como os conceitos do domínio da aplicação são mapeados nas estruturas hipermídia, sem levar em conta detalhes de implementação.

- Autoria em ponto pequeno: depois de definidos nós e elos, e quais especificações (ou instâncias) destes tipos ocorrerão numa dada aplicação, temos que definir como o conteúdo dos nós será apresentado ao leitor, ou seja, como as informações que os nós trazem serão passadas ao leitor e como os elos serão assinalados. Este processo é chamado autoria em ponto pequeno. A preocupação maior é quanto aos aspectos perceptivos da interface.

  Procura-se sempre obter um projeto em ponto pequeno coerente com o projeto em ponto grande, para que os elementos definidos em ponto grande sirvam como unidades naturais para a definição dos parâmetros do projeto em ponto pequeno (por exemplo, definir que todos os nós de um mesmo tipo terão a mesma apresentação visual).

- Estruturas de acesso: conjunto de estruturas de acesso inicial e de roteiros. Uma aplicação hipermídia deve fornecer uma certa variedade de formas de acesso ao material da hiperbase (conjunto de informações que será usado na aplicação hipermídia), como por exemplo, menus ou índices hierárquicos que possibilitam ao leitor a escolha de um tópico de interesse dentro de uma taxonomia pré-estabelecida e os roteiros (ou *guided tours*) que são seqüências arbitrárias, pré-programadas ou não, de nós na hiperbase, que o autor deseja oferecer ao leitor, de acordo com seu perfil e das tarefas particulares que deve cumprir.

- Semântica de navegação: é a parte responsável pela especificação do comportamento dinâmico da aplicação, pois se tratando de uma aplicação interativa, tem-se que definir como o usuário irá perceber a ativação das diferentes partes do sistema (nós, elos, roteiros etc). Um exemplo é mostrado na figura 2, onde o usuário vai ativar diversas partes da aplicação hipermídia através do clique nas palavras sublinhadas.

Figura 2 - Representação de semântica (como o usuário vai interagir com a aplicação)

2.2 Construindo aplicações hipermídia

Em se tratando do processo de construção de aplicações hipermídia, pode-se afirmar que a aplicação consiste de uma hiperbase, de um conjunto de estruturas de acesso e de uma interface com o usuário. O projeto de uma aplicação deve cuidar de cada um destes elementos de processo, tratando-os de forma diferenciada, mas coordenada. Cabe observar que a existência prévia de material a ser utilizado deve ser levada em conta em cada um deles.

Schwabe [Sch93] afirma que o projeto de uma aplicação hipermídia deve cuidar da modelagem conceitual, isto é, do mapeamento da estrutura da informação a ser apresentada, abstraindo-se da parte de apresentação.Com isso o projetista permite que um mesmo projeto lógico possa ser utilizado para implementação de diversas plataformas de *hardware* e *software* (reuso).

Segundo o autor, após a modelagem da aplicação, deve vir o projeto navegacional, onde há a descrição das estruturas de acesso e dos contextos nos quais o usuário pode navegar, ou seja, a fase em que são definidos os *links* ou ligações a serem implementadas em cada página da aplicação. É importante notar que as estruturas definidas aqui são conceituais, no mesmo nível do domínio da aplicação (e das estruturas definidas na modelagem conceitual). A semântica de navegação de alto nível é também definida aqui, no mínimo em termos para estabelecer possíveis caminhos de navegação.

Na seqüência, tem-se o *design* abstrato da interface, onde se especifica a forma e o conteúdo da interface com o usuário, obedecendo às determinações da modelagem e navegação e mantendo-se o critério de abstração e reuso. A preocupação principal deste passo é a interação do usuário com a aplicação. Depois que este passo for concluído, tem-se informação bastante para implementar a aplicação usando um ambiente hipermídia. Em outras palavras, o

design abstrato da interface provê um mapeamento entre a abstração (o projeto da aplicação em alto-nível) e a implementação propriamente dita em um dado ambiente de *software* e *hardware*.

Finalizada esta etapa, o autor pode produzir um sistema hipermídia a ser executado.

Na implementação da aplicação, Nielsen [Nil93], chama a atenção para a interface com o usuário. Segundo o autor, a propriedade de uma interface que nos permite qualificá-la como adequada ou não é conhecida como usabilidade, definida tradicionalmente como a conjunção de cinco atributos:

- aprendizado fácil: o sistema deve permitir que o usuário aprenda a executar suas tarefas no tempo mais curto possível;
- eficiência: o sistema, uma vez dominado pelo usuário, permite um alto grau de produtividade;
- memorabilidade: o sistema deve ser lembrado facilmente mesmo pelo usuário casual, de forma que o retorno ao sistema não implique em um reaprendizado extensivo;
- minimização dos erros: o sistema deve ter uma taxa de erros baixa. Além disso, os erros dos usuários devem ser facilmente recuperáveis.
- satisfação: o sistema deve ser agradável de usar, ou seja, os usuários ficam subjetivamente satisfeitos com ele.

Geralmente os *softwares* utilizados para implementação de *design* abstrato da interface recebem a denominação genérica de “sistemas de autoria”. Alguns desses exemplos são o Toolbook, o Hypercard e o Director. Por outro lado, os métodos de apoio à modelagem e navegação mais conhecidos são o HDM – *Hypermidia Design Model* [GPS91] e o OOHDM – *Object Oriented Hipermidia Design Model* [Sch93, SR94].

2.3 Hipermídia na educação

Sob a ótica da psicologia cognitiva, aprender consiste na reorganização de estruturas de conhecimento. Tais estruturas poderiam ser modeladas por redes semânticas (idéias ligadas por associações), que possibilitam ao aprendiz raciocinar para combinar idéias, inferir e extrapolar. O aprendizado ocorre quando novas estruturas são construídas via a associação de novos nós com outros já existentes. Quanto mais ligações puderem ser feitas entre o conhecimento anterior e o novo, melhor a informação será compreendida e mais facilmente ocorrerá o aprendizado. Se indivíduos aprendem aumentando, combinando e rearranjando mapas cognitivos que são sobrepostos e interconectados, e se diferentes indivíduos possuem diferentes mapas cognitivos, então, sistemas hipermídia, com sua flexibilidade para oferecer ao aprendiz liberdade para criar nós e associações, parecem modelar bem o processo de aprendizagem [Car90]. Além disso, os sistemas hipermídia favorecem o aprendizado ao oferecer ao estudante controle sobre a experiência de aprendizagem. Não é necessário estabelecer uma seqüência fixa para cada estudante; cada um pode tomar suas próprias decisões para satisfazer suas necessidades e objetivos, no seu próprio ritmo e de acordo com seu conhecimento prévio. Essas vantagens devem ser bastante valorizadas num ambiente educacional. Ao ter controle sobre o material instrucional, o estudante naturalmente se sente mais motivado e aliviado de frustrações e ansiedades, uma vez que é capaz de ignorar material que ele já domina (ou pensa que domina) ou que não deseja ter acesso e, conseqüentemente, se concentrar naquilo que considera relevante [VHN97].

Nota-se a imensa vantagem de se utilizar hipermídia associada ao ensino. Mas é importante também destacar que a aplicação deve ser muito bem

elaborada. Não se pode deixar o aluno “solto” na aplicação. As informações contidas na aplicação devem estar distribuídas de maneira clara e eficaz. O que se deseja garantir ao autor é a possibilidade da elaboração cuidadosa de sua mensagem, permitindo-lhe incluir nela o que for essencial ou relevante, sem eliminar o secundário, o complementar, desde que esse não conflite com o essencial e que possa contribuir para os objetivos do usuário.

O que importa, de fato, é que a organização original possibilite uma seleção de elementos do roteiro que contemplem os requisitos de aprendizagem do usuário. Levando isto em conta, é sugerido que os autores destas aplicações sigam algum tipo de modelo. Este trabalho sugere o modelo OOHDM, que é um modelo robusto e eficaz para modelagem hipermídia.

2.4 OOHDM – *“Object Oriented Hypermida Design Model”*

2.4.1 Introdução

Nos últimos anos, vem desenvolvendo-se na comunidade hipermídia um debate interessante sobre a abrangência das aplicações hipermídia e os problemas prováveis de serem encontrados por criadores e leitores ao construir aplicações hipermídia. É ilustrativo ler algumas opiniões sobre estas questões:

-*"Enquanto os leitores podem encontrar dificuldades na compreensão de um hiperdocumento... seus autores têm de lutar com seus próprios problemas...Eles não possuem diretrizes orientando-os sobre como deve ser um hiperdocumento."* [THH 91]

-*"Muitas aplicações hipermídia lidam com domínios de aplicação muito complexos... onde a complexidade é ampliada por exigências adicionais de consistência"* [GPS 91]

- *"Por algum tempo, os problemas de navegação têm sido assuntos em voga na área de hipertextos... este problema não é inerente ao conceito de hipertexto, mas deve-se à fraqueza dos mecanismos de especificação de estruturas..."* [NN91]

-*"Um modelo formal ajuda, na medida em que permite a abstração e a separação entre a estrutura e o conteúdo do hipertexto; ele nos proporciona uma melhor compreensão da semântica de navegação e fornece uma interpretação consistente para propósitos de implementação"* [ZP92]

Pode-se perceber pelas opiniões anteriores que autores, para desenvolver aplicações hipermídia eficazes, para que os usuários pudessem se satisfazer, precisavam de uma "ferramenta" que lhes pudesse auxiliar. Vários *browsers*, mapas, entre outros recursos, foram propostos, mas ainda não se tinha chegado ao objetivo proposto. Ainda se comentava:

- *"Mesmo assim, não possuímos diretrizes e ferramentas para projetar e desenvolver aplicações hipermídia. Isto se aplica particularmente aos sistemas de escala comercial que freqüentemente envolvem intercâmbio de informações. Na falta de tais diretrizes e ferramentas de projeto, a sempre crescente rede de aplicações interligadas vem tornando-se, além de progressivamente parecida com um prato de espaguete, de difícil manutenção.* [BK95]

-*"Desenvolvedores que possuem uma visão limitada de hiperdocumentos podem dispensar a possibilidade de suporte de hipertexto em aplicações computacionais. Infelizmente, muitos desenvolvedores de sistemas de informática vêem o hipertexto apenas com fins de acesso e gerenciamento de documentos... e... o hipertexto deve ser integrado no projeto de uma aplicação computacional"*. [BK95]

Embora se possa encontrar diferentes questões nas citações anteriores, é interessante observar que a maioria delas lida com problemas de especificação e de projeto, ainda que de formas diferentes. Sem surpresa alguma são encontradas poucas referências à manutenção de aplicações hipermídia. Isto deve-se à pouca idade da hipermídia e ao fato de que muitos sistemas são vendidos como programas fechados (em formato de CD-ROM, por exemplo), sem possibilidade de serem modificados de acordo com as necessidades do usuário.

Na medida em que a hipermídia ganha projeção, novos problemas são enfocados: a explosão da web, por exemplo, fez emergir inúmeras questões previamente discutidas. Além disto, as aplicações hipermídia necessitam de manutenção e seus projetistas começam agora a enfrentar este problema.

Pode-se perceber, portanto, a necessidade de um tipo de modelo para o desenvolvimento de aplicações hipermídia. Um modelo que possibilite uma abordagem sistemática e abrangente, na qual todos os aspectos do empreendimento de projeto hipermídia sejam considerados e as decisões de projetos sejam gravadas com o intuito de serem mais tarde rastreadas. É nesse cenário que o modelo OOHDM surge. Ele foi proposto por Gustavo Rossi em sua tese de mestrado, orientado pelo professor Daniel Schwabe em 1994, na PUC-RJ.

O OOHDM é primariamente um mecanismo de modelagem, que permite a descrição do domínio da aplicação usando mecanismos de alto nível e independente do sistema de hipertextos já existente ou a ser projetado. O modelo faz uma evolução desde o modelo abstrato até uma descrição em alto nível da aplicação hipermídia e mais tarde em aplicações concretas, que não necessariamente precisam ser implementadas num sistema orientado a objeto. Ele ainda permite o projeto conceitual da aplicação sem que o autor tenha que se

preocupar demasiadamente com detalhes de implementação, servindo também como forma de comunicação entre projetistas, implementadores e usuários.

Como o OOHDM é orientado a objeto, ele pode ser usado como uma poderosa estrutura para reutilização de componentes hipermídia.

2.4.2 Descrição do modelo

OOHDM considera o processo de desenvolvimento da aplicação hipermídia como um processo de quatro atividades, desempenhadas em uma mistura de estilos iterativos e incrementais de desenvolvimento; em cada etapa um modelo é construído ou enriquecido.

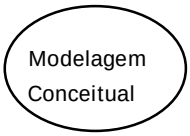
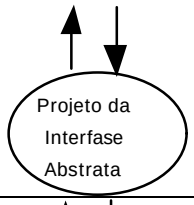
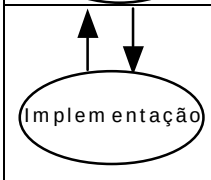
Atividades	Produtos	Mecanismos	Interesses do Projeto
	Classes, sub-sistemas, relacionamentos, perspectivas de atributos	Classificação, composição, generalização e especialização	Modelagem da semântica do domínio de aplicação
	Nós, elos, estruturas de acesso, contextos de navegação, transformações navegacionais	Mapeamento entre objetos conceituais e de navegação. Padrões de navegação para a descrição da estrutura geral da aplicação.	Leva em conta o perfil do usuário e a tarefa; ênfase em aspetos cognitivos e arquiteturais.
	Objetos de interface abstrata, reações a eventos externos, transformações de interface	Mapeamento entre objetos de navegação e objetos de interface.	Modelagem de objetos perceptíveis,implementa metáforas escolhidas. Descrição de interface para objetos navegacionais
	Aplicação em execução	Aqueles fornecidos pelo ambiente alvo	Desempenho, completitude

Tabela 1 – Esboço da metodologia OOHDM

A tabela 1 resume as quatro atividades do modelo OOHDM: modelagem de domínio ou conceitual, projeto navegacional, projeto da interface abstrata e implementação, assim como os produtos gerados em cada uma e os mecanismos de abstração utilizados ao longo do processo. As setas de avanço demonstram a evolução natural do empreendimento de projeto, enquanto as setas de retorno expressam não apenas a possibilidade de *feedback* (volta no processo), mas também a existência de um modelo simples, porém potente, de rastreamento que permite mapear as modificações em um modelo de projeto para partes de um outro modelo (rastreamento para trás) [SR94].

Durante a modelagem conceitual, um modelo do domínio da aplicação é construído utilizando-se princípios bem conhecidos de modelagem orientada a objetos aumentados com algumas primitivas tais como perspectivas de atributo e sub-sistemas. Classes conceituais podem ser construídas utilizando-se hierarquias de agregação e de generalização/especialização. Não há, nesta etapa preocupação com os tipos de usuários e tarefas, apenas com a semântica do domínio da aplicação. O produto desta etapa é um esquema de classes e objetos construído a partir de Sub-Sistemas, Classes e Relações [SR94].

Uma das características das aplicações hipermídia é a noção de navegação. No OOHDM, uma aplicação é vista como uma visão navegacional do modelo conceitual. Esta visão é construída durante o Projeto Navegacional levando-se em conta os tipos de usuários aos quais a aplicação se destina e o conjunto de tarefas que deverão desempenhar utilizando-o. Diferentes modelos navegacionais podem ser construídos para o mesmo esquema conceitual, expressando, desta forma, diferentes visões (aplicações) no mesmo domínio [SR94].

Durante o Projeto de Interface Abstrata um modelo de interface é construído. Este modelo especifica que objetos de interface serão vistos pelo

usuário e, particularmente, a forma que tomarão diferentes objetos navegacionais, que objetos de interface ativarão a navegação, a maneira como os objetos de interface multimídia serão sincronizados e que transformações ocorrerão na interface [SR94].

Finalmente, mapeando os modelos navegacionais e de interface ao ambiente de implementação escolhido, o autor produz o sistema real de hipermídia a ser rodado. Em particular, o modelo de interface pode ser implementado de forma direta sobre plataformas hipermídia disponíveis tais como Hypercard, Toolbook, MacWeb [SR94].

Analisando as informações acima, pode-se afirmar que o OOHDM melhora os processos de construção de aplicações hipermídia com a introdução de um novo conceito: a utilização de objetos. Segundo Schwabe e Rossi [SR94], as vantagens de se utilizar objetos na modelagem de aplicações hipermídia são:

- oferece um referencial natural para raciocinar acerca de entidades do mundo real, objeto da maior parte das aplicações hipermídia;
- fornece mecanismos de abstração adequados ao nosso empreendimento como Agregação e Especialização/Generalização;
- desde que estrutura e comportamento sejam encapsulados por objetos, seremos capazes de elaborar não apenas aplicações hipermídia "convencionais" (como *Art Gallery* ou *Dinossaurs* da Microsoft), mas também aplicações sofisticados combinando navegação hipermídia com diferentes tipos de processamento de informação, como ocorre, por exemplo, nos ambientes de Engenharia de *Software*;
- são utilizadas as mesmas primitivas de modelagem (objetos, classes), simplificando a transição de uma atividade para outra;
- ao longo do processo utilizamos os mesmos mecanismos de abstração, isto é, agregação, classificação e generalização/especialização.

- pelo fato de os objetos serem artefatos reativos, podem ser construídas aplicações sofisticadas baseadas em hipermídia, definindo-se padrões de comportamento e de comunicação entre objetos;
- aplicações projetadas e construídas em torno de objetos tendem a ser mais robustas e fáceis de modificar, tanto pelo uso de polimorfismo e herança como por composição;
- construir novas aplicações reutilizando componentes existentes é altamente viável quando os componentes são descritos como objetos.

2.4.3 Modelagem Conceitual

No OOHDM, um esquema de modelagem de aplicações hipermídia orientado a objeto é construído a partir de objetos, classes, relacionamentos e sub-sistemas. As classes são descritas em modelos orientados a objetos, mas seus atributos podem ser multi-tipados, representando diferentes perspectivas da mesma entidade real. Usa-se uma notação semelhante a OMT de Rumbaugh [RB91], enriquecida com informações de subsistema. Agregação e Generalização/Especialização são utilizadas para aumentar o poder de abstração do sistema. Além disto, utiliza-se Cartões de Classes e Relacionamentos, para assistir na documentação. Tais cartões auxiliam na tomada de decisões, tanto "para frente" como "para trás", a serem feitas no decorrer do projeto. Apesar de o processo de modelagem de domínio ser basicamente o mesmo em diversas aplicações (mesmo as aplicações “não hipermídia”), algumas decisões tomadas na etapa de modelagem podem refletir na estrutura de uma aplicação hipermídia. O projeto cuidadoso de estruturas agregadas por exemplo, facilitará a definição do estilo de navegação [SR94].

- Classes, relacionamentos e sub-sistemas

Um esquema consiste em um conjunto de objetos e classes conectadas por relacionamentos. Os objetos são instâncias de classes, e assim, quando um relacionamento existe entre classes, ele abstrai um relacionamento objeto – objeto. Quando é possível, relacionamentos do tipo classe – classe são usados. Classes podem ser relacionadas com sub-sistemas (abstrações de todo o esquema hipermídia). A figura 3 mostra parte de um esquema de uma aplicação para uma faculdade. Note que há a representação de um sub-sistema **departamentos** e também da relação entre duas classes **faculdade** e **fotos da faculdade**.

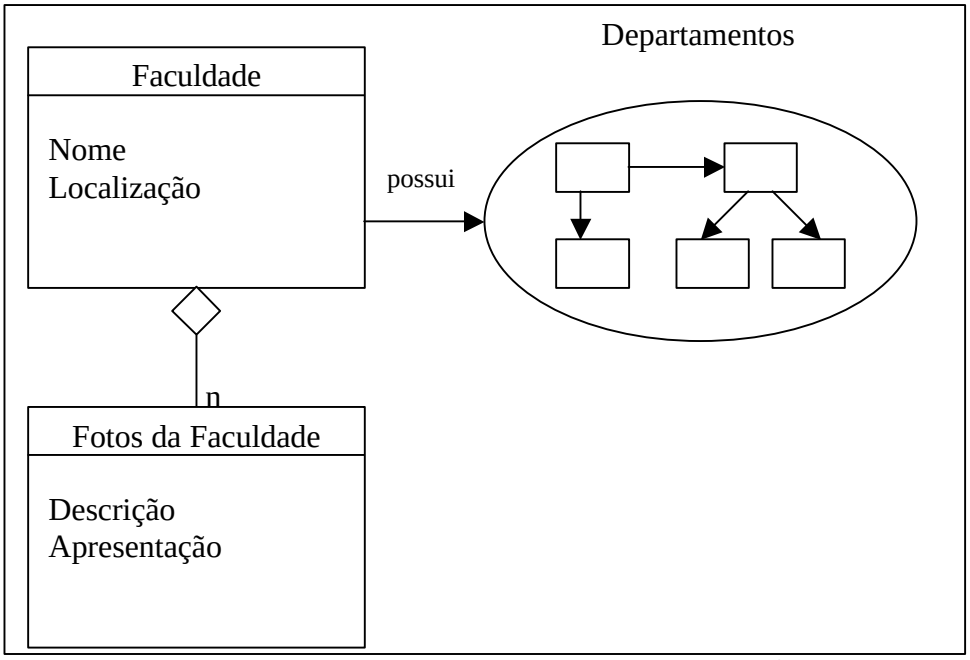


Figura 3 - Parte de um esquema de uma aplicação para uma faculdade

Assim como é usual em modelos orientados a objeto, classes são descritas como um conjunto de atributos e métodos (implementando o comportamento dos objetos) e depois organizados em hierarquias.

Antes de se falar mais detalhadamente em classes é importante definir primeiramente o que constitui uma classe: atributos, tipos e perspectiva.

Um atributo é algum dado (informação de estado) para o qual cada objeto em uma classe tem seu próprio valor. O tipo (ou classe) de um atributo irá representar a aparência retórica ou visual na aplicação final [SR94].

Cada possibilidade de aparência é chamada de perspectiva. Alguns exemplos desse mecanismo são os seguintes: o atributo descrição em **Fotos da Faculdade** (figura 3) pode ser visto como texto (uma descrição das fotos). A descrição de uma lei pode ser vista em vários estilos: texto oficial, uma interpretação etc.

Em OOHDM a representação de um atributo segue o seguinte formato:

Forma Geral: nome-do-atributo: [tipo]

Exemplos:

descrição: [*Image,Text*]

apresentação: [*Image+,text,Sound*]

Quando múltiplas perspectivas existem, usa-se a notação "[...]" e se uma delas é perspectiva padrão (*default*), marca-se ela com um "+". No exemplo dado, pode-se ver que o atributo **apresentação** pode ser visualizado como imagem (perspectiva padrão), texto ou som. Somente a perspectiva *default* tem que estar presente em todas as instâncias, enquanto que as outras podem ou não ser implementadas.A definição de qual atributo estará perceptível aos usuários é feita no segundo passo de construção de aplicações (Projeto de Navegação).

Uma relação ou relacionamento é um modelo dos mapeamentos do domínio do problema que um objeto precisa ter com outros objetos para poder cumprir com suas responsabilidades.

Na figura 4, por exemplo, o relacionamento **possui** descreve o fato de que obras de arte têm donos. Uma implementação hipermídia apresentaria um nó representando a **Obra** de arte e outro representando a **Pessoa** que possui a obra.

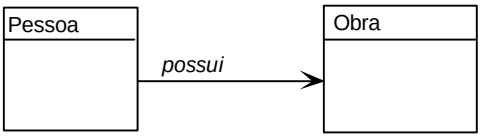


Figura 4 - Um relacionamento entre classes

Sub-sistemas que representam esquemas hipermídia completos são usados com parte em outros esquemas. É uma abstração de um sistema conceitual completo (empacotamento) [SR94]. Eles podem ser usados quando outro esquema ou aplicação hipermídia já existe ou que pode ser definida oportunamente (como o sub-sistema **Departamentos** da figura 3). Usualmente sub-sistemas contém pontos de entrada, ou seja, classes do sub-sistema que podem ser acessadas de outra classe fora dele. Pontos de entrada são completamente definidos enquanto se define a semântica de navegação. A figura 5 mostra um exemplo. Nela existem dois pontos de entrada, um a partir da classe **Exibição** e outro a partir da classe **Coleção**.

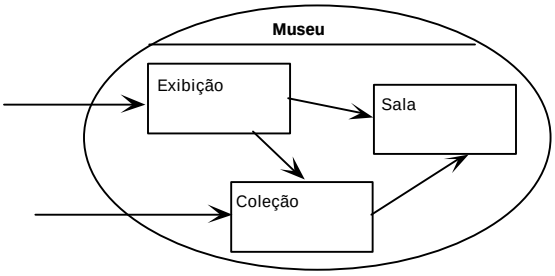


Figura 5 - Pontos de entrada do sub-sistema Museu

Usa-se cartões de classes, relacionamentos e sub-sistemas para documentar o modelo. Eles incluem informações sobre os artefatos

documentados e permitem rastrear informações. Cartões são fáceis de se manipular e podem ser automatizados em estilo hipermídia [SR94]. Na figura 6 encontra-se a representação de um cartão para um sub-sistema.

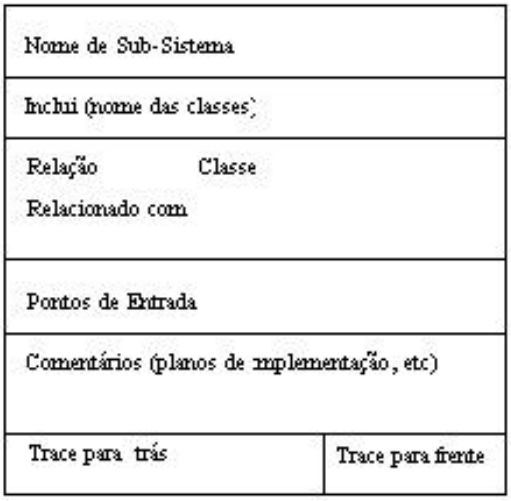


Figura 6 - Cartão para sub-sistemas

- Mecanismos de abstração

Schwabe [SR94] fornece dois tipos de construções abstratas para lidar com a complexidade das aplicações: agregação e generalização/especificação. A primeira é muito útil para descrever classes complexas como uma agregação de outras mais simples, e a segunda para construir hierarquias de classes usando herança como mecanismo de compartilhamento. A noção de sub-sistema pode ser vista como um terceiro mecanismo de abstração de alto nível.

Uma agregação, em orientação a objeto, geralmente, pode ser visto como estrutura Todo-Parte. Uma estrutura Todo-Parte representa um dos três métodos básicos de organização que permeiam os pensamentos humanos. Um exemplo disto é: o todo **Veículo** com a parte **Motor**. Menos formalmente, uma estrutura Todo-Parte é considerada como uma estrutura "tem um", por exemplo: um

Veículo tem um **Motor**. Definir bem as estruturas de agregação é importante em hipermídia porque suas especificações podem ser úteis quando definir os contextos e transições navegacionais. Entender a exata natureza de uma agregação, isto é, que tipo de composição de objetos ela representa, é importante para se poder construir boas estruturas navegacionais [SR94]. A figura 7 mostra um exemplo de agregação. Ela pode ser interpretada da seguinte forma: **ImagemCidade** é parte de **Cidade** ou **Cidade** tem uma **ImagemCidade**. Neste tipo de estrutura as classes herdam seus atributos, relacionamentos e comportamento de suas super-classes. Percebe-se então que **ImagemCidade** herda os atributos já definidos na classe **Cidade**.

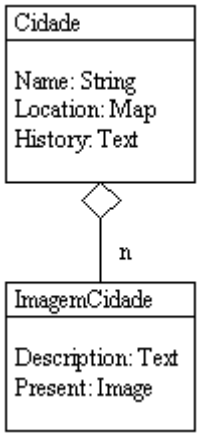


Figura 7 - Representação de agregação entre classes

A estrutura de generalização/especialização pode ser vista como parte da distinção entre classes. Um exemplo de generalização é **Veículo** e de especialização é **VeículoCaminhão**. Menos formalmente, esta estrutura é considerada como uma estrutura "é um" ou "é um tipo de", por exemplo, **VeículoCaminhão** "é um (é um tipo de)" **Veículo**. A figura 8 representa um caso de generalização/especialização. Na figura, o relacionamento entre a classe **AtraçãoTurística** e **Cidade** reflete o fato de que cada sub-classe de

AtraçãoTurística (**Igreja** e **Monumentos**) estarem localizadas em uma **Cidade**. A classe **Igreja** e **Monumento** compartilham entre os atributos herdados da classe **AtraçãoTurística**. A classe **AtraçãoTurística** sofreu uma generalização/especialização.

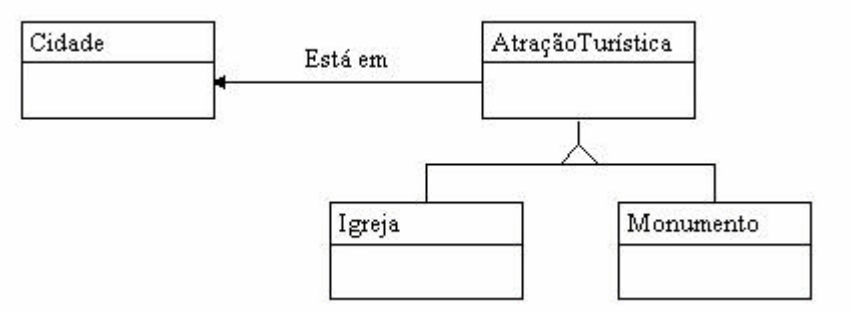


Figura 8 - Exemplo de generalização/especialização

- Significado do comportamento das classes

No modelo OOHDM, a definição do comportamento das classes é útil para [Lim94]:

- Computar elos, isto é, para expressar aqueles elos que não são definidos explicitamente e devem ser computados em tempo real. Isto acontece em aplicações nas quais a hiperbase se atualiza por meio do usuário (por exemplo, num ambiente CASE). Este comportamento abstrato é implementado através da atribuição de métodos à classes e relacionamentos.

- Para prover acesso aos atributos das classes (como em outros modelos orientados a objeto), tanto para satisfazer consultas sobre os atributos de um objeto, quanto para permitir consultas baseadas em acessos e para atualizar um objeto. Este último caso é usado em aplicações que permitam atualizações durante a navegação.

- Produto final da modelagem conceitual

- Definição de classes conceituais, isto é, classes que contém informações previamente divididas, mas que ainda não contém ligações entre si (isto será feito no projeto de navegação).

- Um esquema conceitual. (Na verdade, um conjunto de esquemas conceituais, sendo um para cada subsistema).

- Um conjunto de definições de subsistemas, objetos e relacionamentos.

2.4.4 Projeto de Navegação

As aplicações de hipermídia são projetadas para efetuar navegação através de um espaço de informações. Por isto, o projeto da estrutura de navegação de tais aplicações é a etapa crucial no empreendimento de desenvolvimento [SR94].

Apesar de ser comprovada a necessidade de se terem formalismos adequados para acabar com problemas originados de estruturas de navegação mal feitas, os métodos atuais de projeto de hipermídia são falhos. O modelo OOHDM vem corrigir essa situação. Ele propõe a construção de um modelo conceitual compartilhado no qual focalizamos os objetos e relacionamentos do domínio. Após essa etapa, no projeto navegacional, diferentes visões navegacionais serão “derivadas” deste modelo, levando em conta os perfis dos futuros usuários. O modelo conceitual funcionará como um repositório compartilhado de modelagem, a partir do qual se construirá diferentes visões (na verdade, visões navegacionais) do domínio do problema; cada uma delas constituirá um tipo distinto de aplicação hipermídia [SR94].

Dentro do Projeto Navegacional, estabelece-se as estratégias de navegação bem como as visões que um determinado usuário terá ao navegar pela aplicação. Ao projetar a estrutura de navegação de um aplicativo de hipermídia serão considerados aspectos como :

- Quais objetos serão navegados e que atributos possuem? Quais os relacionamentos entre estes objetos e aqueles definidos no esquema conceitual?

- Qual é a estrutura subjacente de navegação? Em que contextos o usuário irá navegar?

- Quais as estruturas de elo existentes entre os objetos que serão navegados (elos, caminhos, índices, roteiros guiados, etc.)?

Dentro do projeto navegacional são desenvolvidas as seguintes atividades: O esquema das classes navegacionais e o esquema de contextos navegacionais.

- Estruturas de acesso e classes navegacionais

As estruturas de acesso são caracterizadas por um conjunto de seletores, um conjunto de objetos de destino (usualmente objetos no esquema) e um predicado nestes objetos [Lim94]. Estas estruturas podem conter outros atributos, com informações sobre a própria estrutura ou destinada a serem apresentadas ao usuário final.

As estruturas de acesso são um tipo específico de nós, onde as âncoras são definidas por seletores e os elos para objetos destino são definidos implicitamente. O predicado expressa quais objetos serão acessíveis em termos de suas propriedades. Seletores usualmente esperam por algum atributo dos objetos de destino. Em ambos os casos eles tem que ser mencionados explicitamente na definição das estruturas de acesso. As estruturas de acesso funcionam como índices ou dicionários, e ajudam o usuário final a encontrar a informação desejada.

Usa-se como formalismo para expressar contextos navegacionais e suas semânticas a idéia de classe navegacional, que representa nós, elos e outras estruturas para o *design* de aplicações hipermídia. Nós possuem atributos, âncoras para elos e estruturas de acesso. Elos contêm informações sobre os elos

de destino e origem, seus próprios atributos e comportamento. Classes navegacionais são organizadas em hierarquias e definem contextos para a navegação na aplicação.

As classes navegacionais são usadas para se definir objetos navegáveis, e também para representar estruturas de acesso, *browsers* e outros tipos de facilidades de navegação não definidas previamente, assim como as interfaces de consultas. Elas enriquecem o modelo e fornecem uma transição tranqüila do *design* de baixo nível, onde os aspectos da interface são definidos, até a implementação [Lim94].

Agora será mostrado um exemplo para ilustrar a idéia de classes navegacionais. Este exemplo mostra o mapeamento direto da classe conceitual (produto final do processo de modelagem conceitual) para uma classe navegacional.

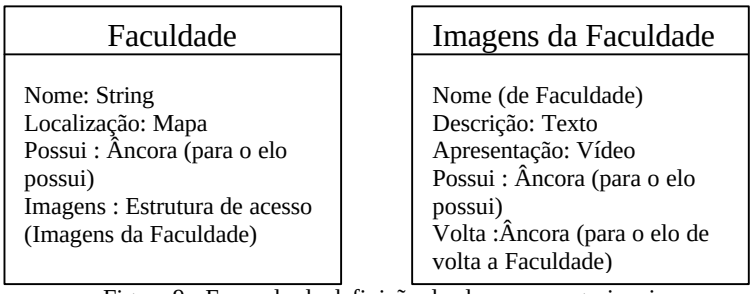


Figura 9 - Exemplo de definição de classes navegacionais

Suponha as duas classes definidas no exemplo da figura 3: **Faculdade** e **Imagens da Faculdade**. Define-se então agora, as classes navegacionais (que representam objetos navegacionais, que subdividem-se em classes de nós e classes de elos). Estas classes navegacionais estão definidas na figura 9.

A classe de nós **Faculdade** tem os seus atributos advindos diretamente dos atributos e relacionamentos gerados na classe conceitual: **nome** e **localização** são copiados diretamente; **possui** é induzido pelo relacionamento

correspondente, enquanto **imagens** é induzido diretamente pela agregação correspondente nas classes conceituais. O atributo **imagens** contém uma estrutura de acesso para os nós que descrevem as imagens da atual faculdade. Note que seu tipo é estrutura de acesso e não foi definido aqui; esta classe pode ser implementada usando-se diferentes (possivelmente primitivas) estruturas de dados, como listas, árvores etc.

A classe de nós **Imagens da Faculdade** foi definida de maneira similar, onde os atributos e relacionamentos vieram diretamente das classes conceituais. A única diferença aqui é a classe navegacional herdou os atributos **nome** e **possui**. Isto acontece porque é comum este tipo de herança na composição de objetos e cuja semântica permite naturalmente uma navegação na estrutura de nós que compõe a agregação.

- Contexto navegacional

Qualquer aplicação hipermídia bem projetada deve levar em conta o modo como o usuário explora o espaço hipermídia. Deve-se evitar a apresentação de informações redundantes e ajudar, de forma consistente e controlada, o usuário a escolher a maneira como navegará [SR94]. Nós e elos, infelizmente, não são suficientes para alcançar este objeto. É nesse contexto que surge a idéia de contexto navegacional.

A idéia de contexto navegacional nada mais é que a premissa de que os elementos de informação em um hiperdocumento (como os nós, por exemplo) são mais facilmente entendidos quando são apresentados em um contexto, e que o uso de contextos minimiza os problemas da desorientação, comum em aplicações hipermídia complexas. Pode-se também definir contexto navegacional, de uma forma mais simples, como a maneira pela qual o usuário vai navegar pela aplicação hipermídia. Um contexto navegacional é constituído

por um conjunto de nós, elos e outros contextos navegacionais. Inclui, também, um caminho pré-definido entre seus elementos.

2.4.5 Interface Abstrata

A construção de uma interface hipermídia é um aspecto crítico da criação de uma grande aplicação hipermídia. Os métodos atuais de projeto de interface hipermídia tendem a negligenciar esta questão e as interfaces humano-computador de aplicações hipermídia costumam ser totalmente construídas com uso de ferramentas dependentes da implementação e do ambiente.Tais métodos, freqüentemente dão ênfase aos projetos conceitual e navegacional, sem considerar os problemas associados à interface.

Para resolver esta questão, o OOHDM propõe um modelo formal de projetos que deve ser usado anteriormente à implementação, de modo a maximizar a independência de diálogo e o reuso em grande escala dos componentes de interface. Propõe também um modelo de projeto que pode suportar comunicação aperfeiçoada entre projetistas e implementadores: quando as decisões de projeto de interface são documentadas, podem ser usadas tanto como um campo de testes para validar a implementação, quanto como uma referência durante a manutenção [SR94].

Para especificar um modelo abstrato de interface é necessário definir metáforas de interface e descrever suas propriedades estáticas e dinâmicas e seus relacionamentos com o modelo navegacional de uma forma independente de implementação. Para isto é necessário especificar:

- A aparência de cada objeto navegacional que será percebido pelo usuário, isto é, a representação de seus atributos (incluindo as âncoras). O mesmo objeto navegacional pode ter diferentes representações de interface em diferentes situações. Por exemplo, um nó pode ter a

representação de uma fotografia de um monumento ou a de um ícone em um mapa que atue como um índice para monumentos;

- outros objetos de interface para oferecer as diversas funções do aplicativo, como barras de menus, botões de controle e menus;
- os relacionamentos entre os objetos de interface e navegacionais, tais como o modo com que um evento externo, como o fato de o usuário "clicar" o mouse afetará a navegação;
- as transformações de interface que ocorrem pelo efeito da navegação ou de eventos externos no computador de diferentes objetos de interface;
- a sincronização de alguns objetos de interface deve ser considerada, especialmente quando há meios dinâmicos, como áudio e vídeo envolvidos.

Para ilustrar a idéia de interface abstrata, será mostrado um exemplo. Na figura 10 se encontra a interface abstrata para uma aplicação **Flores**. Já na figura 11, tendo a interface abstrata como base, tem-se a aplicação **Margaridas**.

Figura 10 - Interface abstrata “Flores”

Figura 11 - Interface construída a partir da interface abstrata

2.4.6 Implementação

É a última etapa do processo de construção de aplicativos hipermídia. Este processo visa "traduzir" as informações geradas pelos processos anteriores (modelagem conceitual, projeto de navegação e interface abstrata), para que se possa implementar uma aplicação hipermídia.

A implementação de uma aplicação hipermídia de modo que esta resulte numa aplicação usável não é tarefa simples. Muitas questões técnicas e não-técnicas devem ser resolvidas. Uma vez que o ambiente de implementação tenha sido escolhido, o projeto deve ser mapeado para artefatos de implementação e todos os componentes hipermídia têm que ser instanciados.

No Capítulo V será dado um exemplo de construção de uma aplicação hipermídia usando todos os passos OOHDM, onde o produto final (a implementação) será visualizado.

2.5 O Modelo OOHDM e sua utilização na atualidade

O modelo OOHDM hoje em dia é bastante difundido e utilizado. Devido às suas facilidades, muitos autores de hipermídia o têm utilizado. Exemplos da utilização do OOHDM podem ser encontrados em [BS98]. Os autores descrevem vários exemplos de aplicações de *e-commerce* utilizando a modelagem OOHDM. Em [BS98] podem ser encontrados também vários exemplos de *sites* na *internet* que utilizam o Modelo OOHDM em sua confecção. Em [SR94] encontram-se mais dois exemplos: a utilização do modelo para o projeto de uma nova aplicação (aplicação Portinari) e a reengenharia de uma aplicação comercial (Le Louvre).

CAPÍTULO III – METODOLOGIA

A metodologia utilizada para o desenvolvimento deste trabalho fundamenta-se na pesquisa bibliográfica e implementação prática de uma aplicação hipermídia.

Os principais temas de pesquisa bibliográfica foram: hipertexto, hipermídia, hipermídia na educação, OOHDM e interface homem-máquina. A pesquisa foi realizada na *internet* e em periódicos disponíveis na Biblioteca Central da Universidade Federal de Lavras.

Na abordagem prática, procurou-se desenvolver um modelo para uma aplicação educacional, direcionado a linguagens de programação. Na seqüência, implementou-se o *design* abstrato, utilizando-se a ferramenta de autoria Toolbook, versão 3.0, instalada num computador com processador k6-2, 64 de RAM, 8Gbytes de memória e sistema operacional Windows 98.

Em relação ainda a parte prática, como o OOHDM é um modelo sistemático e gradual, cada etapa do processo de construção da aplicação hipermídia, constrói um modelo que é apoiado no anterior. Assim, ao final do projeto, terá sido implementado um hiperdocumento coerente com suas características conceituais e de navegação, definidas em fases anteriores. Este hiperdocumento específico diz respeito a um curso sobre linguagem de programação DELPHI.

CAPÍTULO IV – RESULTADOS E DISCUSSÃO

Neste capítulo será mostrado como é o desenvolvimento de uma aplicação hipermídia utilizando o OOHDM.

O desenvolvimento da aplicação se baseia numa série de dados e informações sobre o ensino virtual de uma linguagem de programação. O objetivo é a criação de uma aplicação genérica, ou seja, explorando os mecanismos de abstração que uma modelagem em OOHDM fornece, onde se criará uma aplicação hipermídia que possa ser usada tanto no ensino da linguagem DELPHI, quanto de qualquer outra, como PASCAL ou C++, bastando para isso acrescentar apenas textos e figuras correspondentes a cada linguagem.

4.1 A modelagem

A figura 12 mostra o resultado do primeiro passo de construção da aplicação hipermídia.

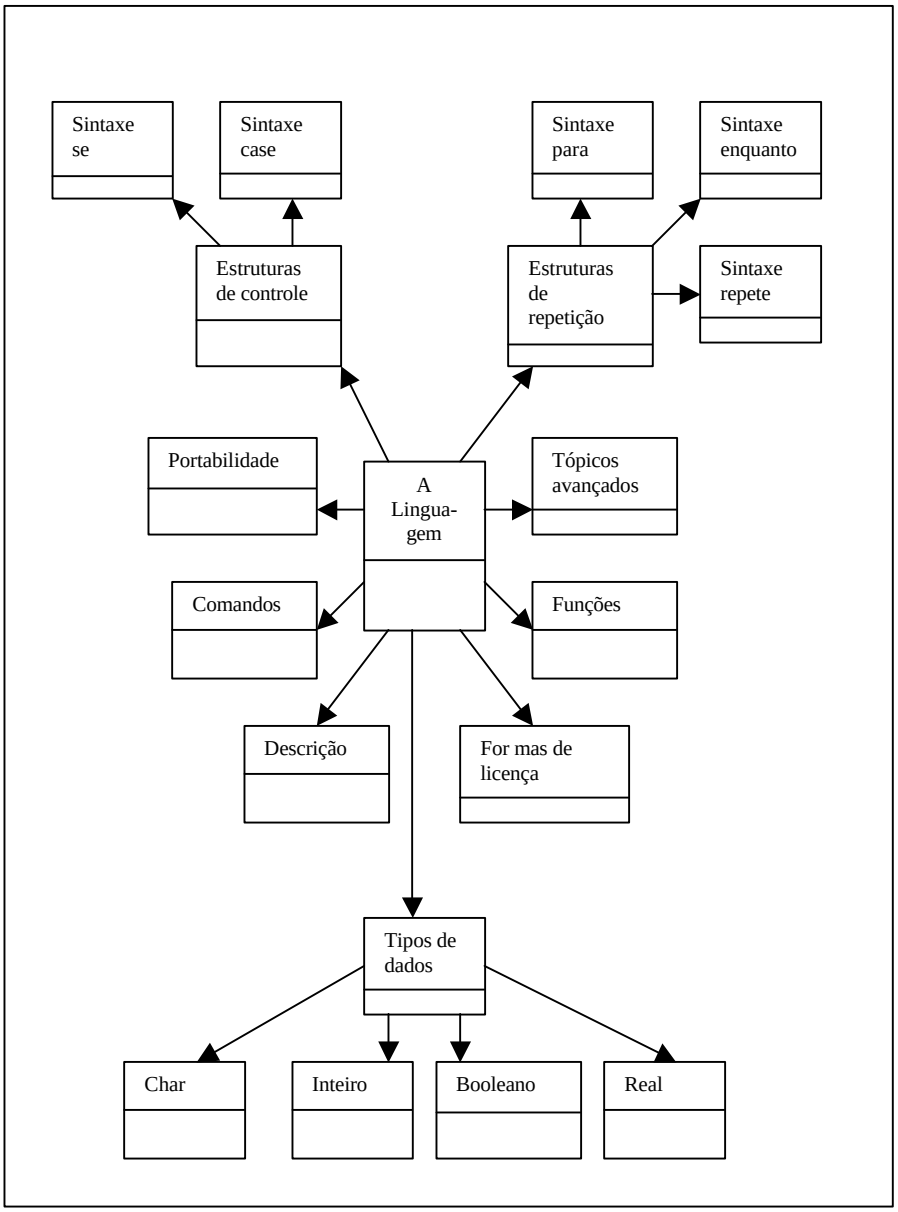


Figura 12 – Esquema de classes da aplicação

O esquema mostrado na figura 12 possui uma estrutura de acesso global chamada **A linguagem**, que no caso, é o índice principal, através do qual se tem acesso às principais classes do esquema.

A partir de agora será mostrado mais detalhadamente as classes presentes na figura 12, seus atributos, perspectivas, sub-classes etc.

A primeira classe a ser analisada é a classe denominada **Histórico**. Nela há uma **descrição** sobre o histórico da linguagem. Além disso, há também um texto no qual se encontram informações sobre as **versões** encontradas da linguagem de programação.

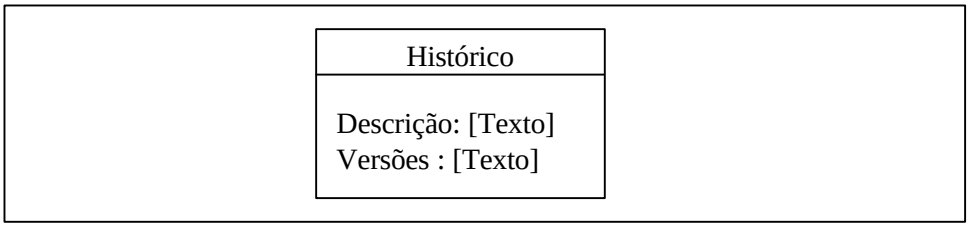


Figura 13 – A classe "Histórico" é mostrada em detalhe junto com seus atributos

A próxima classe a ser analisada é a classe **Funções**. Nela se encontram os atributos **Nome**, que especifica o nome da função, o atributo **Descrição**, onde há uma descrição sobre a função e sua utilização. Além desses dois atributos também há o atributo **Exemplos**, onde se poderá encontrar exemplos utilizando as funções especificadas. A figura 14 ilustra a representação desta classe.

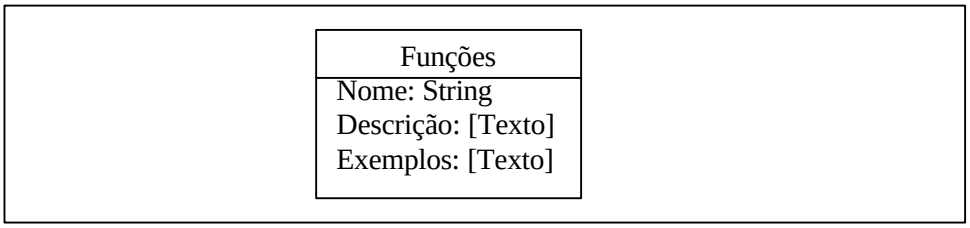


Figura 14 – Representação da classe "Funções" e seus atributos

A figura 15 representa a classe **Formas de Licença**. Nesta classe se encontra uma **descrição** sobre as várias formas de licença de utilização que uma linguagem pode ter.

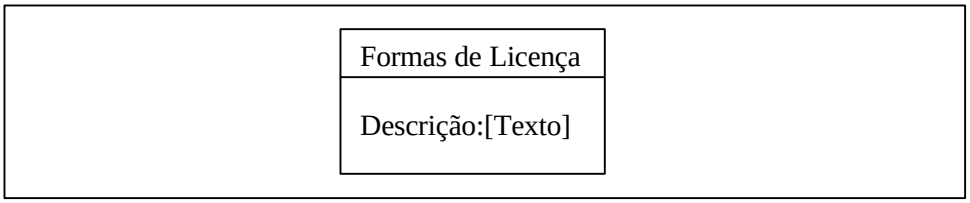


Figura 15 – Representação da classe “Formas de Licença” e seus atributos

A próxima classe representada é **Tópicos Avançados**. Ela contém uma **descrição** sobre peculiaridades avançadas da linguagem, como por exemplo sua utilização em banco de dados e sua configuração em plataformas cliente/servidor.

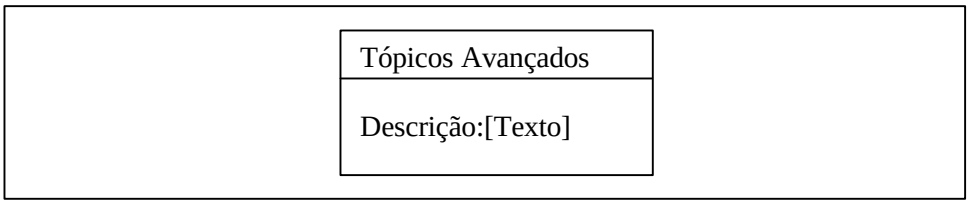


Figura 16 – Representação da classe “Tópicos Avançados” e seu atributo

A figura 17 representa a classe **Portabilidade**. Esta classe contém uma **descrição** sobre a portabilidade da linguagem, ou seja, em quais sistemas operacionais ela funciona ou não.

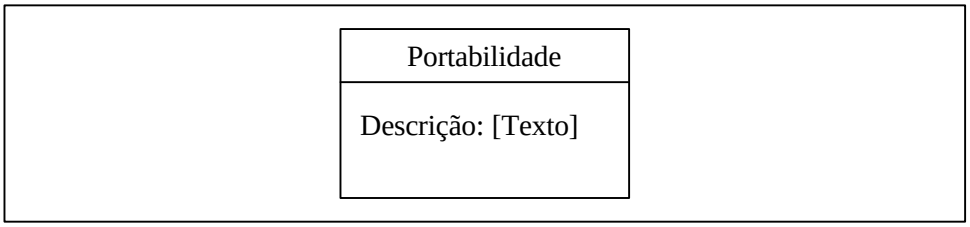


Figura 17 – Representação da classe “Portabilidade” e seus atributos

A classe **Comandos**, representada na figura 18, possui os atributos **Nome** (onde é indicado o nome da função), **Descrição** (onde há uma breve descrição sobre o comando) e **Exemplos** (onde se encontra exemplos da utilização das funções).

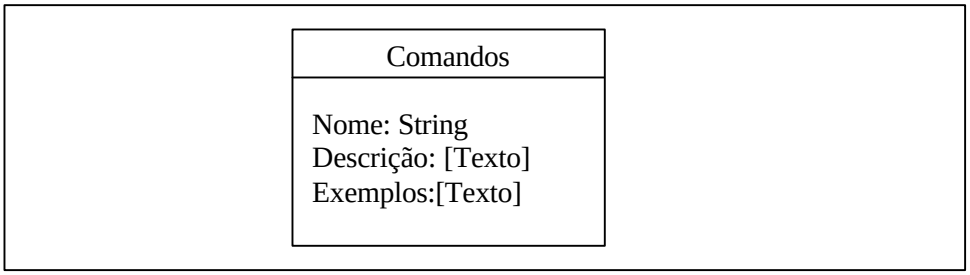


Figura 18 – Representação da classe "Comandos" e seus atributos

A Classe **Estruturas de Controle** é representada na figura 19, juntamente com suas sub-classes **Sintaxe Case** e **Sintaxe Se**. Na classe **Estruturas de Controle**, encontra-se uma **descrição** sobre o que são estruturas de controle e quando devem ser usadas. As sub-classes **Sintaxe Se** e **Sintaxe Case**, além de terem também uma **descrição**, há também **exemplos** de sua utilização. As estruturas de controle são partes muito importantes na linguagem de programação.

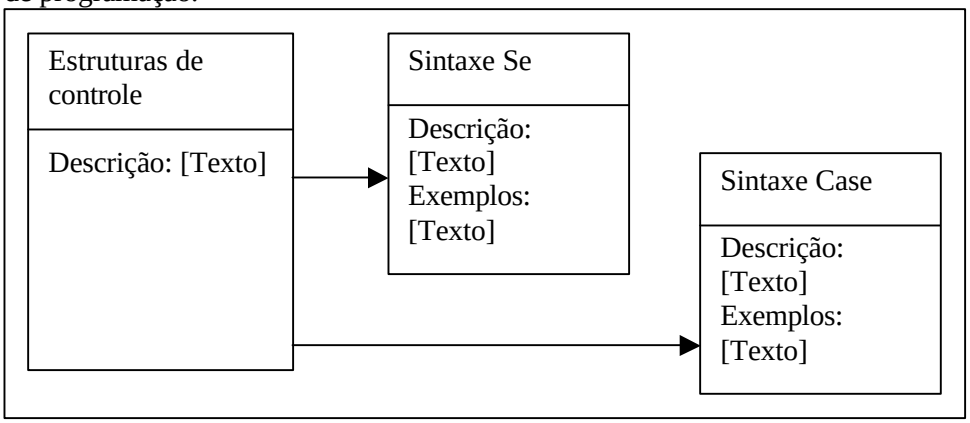


Figura 19 – Representação da Classe "Estruturas de Controle" e suas sub-classes "Sintaxe Se" e "Sintaxe Case"

A figura 20 representa a classe **Estruturas de Repetição** e suas sub-classes **Sintaxe Enquanto**, **Sintaxe Para** e **Sintaxe Repete**. Na classe **Estruturas de Repetição** se encontra uma **descrição** sobre o que são estas estruturas e quando usá-las. As sub-classes **Sintaxe Enquanto**, **Sintaxe Para** e **Sintaxe Repete** contém além da **descrição** de sua utilização, **exemplos** nos quais o aluno poderá ver a utilização prática destas estruturas.

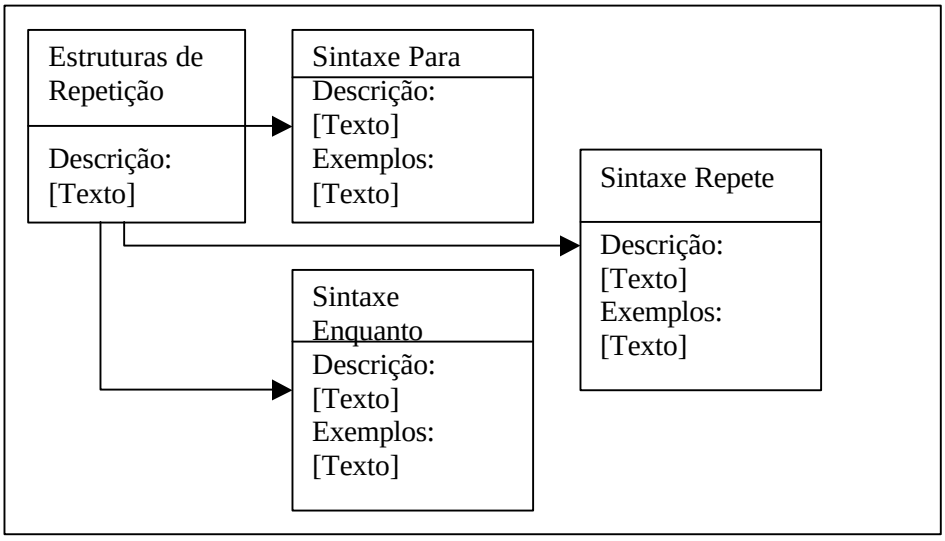


Figura 20 – Representação da classe "Estruturas de Repetição" e suas sub-classes "Sintaxe Repete", "Sintaxe Enquanto" e "Sintaxe Para"

A última classe representada é **Tipos de Dados**. Além dela estão representadas também suas sub-classes **Char**, **Inteiro**, **Real** e **Booleano**. Na classe **Tipos de Dados**, se encontra uma **descrição** sobre o que são tipos de dados e como são usados. As sub-classes **Char**, **Inteiro**, **Real** e **Booleano** além de conterem uma **descrição** sobre sua utilização, também tem **exemplos** que ilustram a sua utilização na linguagem de programação.

É importante notar que as sub-classes das classes **Tipos de Dados**, **Estruturas de Controle** e **Estruturas de Repetição** são todas especializações das classes principais. Foi usada a herança no projeto deste esquema de classes.

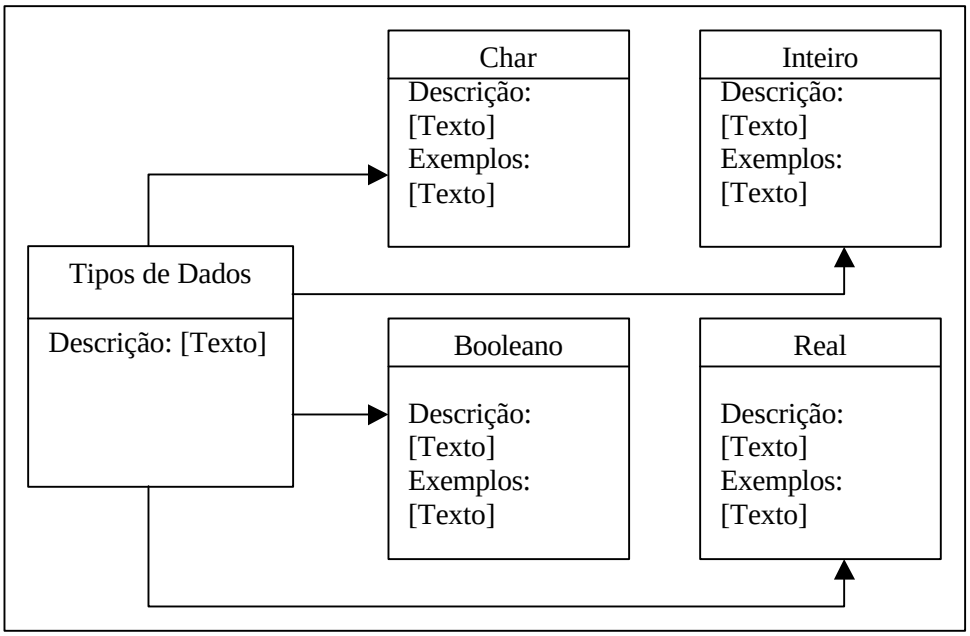


Figura 21 – Representação da classe “Tipos de Dados” e suas sub-classes “Char”, “Inteiro”, “Real” e “Booleano”

A classe **A Linguagem** (principal nesta aplicação hipermídia) representada na figura 12, apenas possui seletores através dos quais se tem acesso às principais classes do esquema. Ela será mostrada no projeto de navegação.

4.2 Projeto de Navegação

O segundo passo no desenvolvimento de uma aplicação hipermídia, depois de já feita a modelagem conceitual, é a definição da semântica de navegação, ou *design* da navegação. Agora serão definidos como as informações

serão acessadas na aplicação. Para isso, haverá a especificação das classes navegacionais da aplicação.

Baseado na classe conceitual **Histórico**, tem-se a classe navegacional representada pela figura 22.

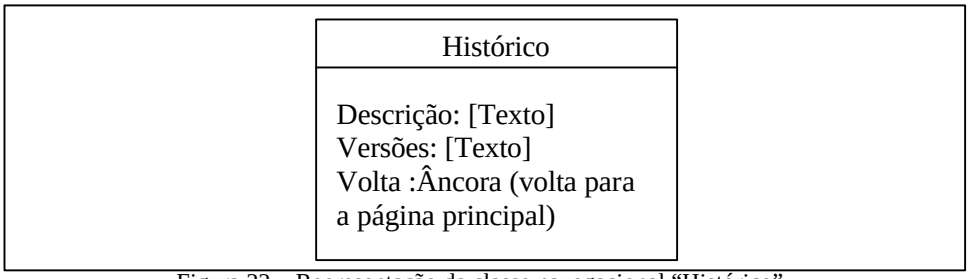


Figura 22 – Representação da classe navegacional "Histórico"

É importante ressaltar que na classe navegacional **Histórico** e em todas as classes navegacionais seguintes, haverá a presença do atributo **Volta**. Este atributo representa um elo que volta à classe anterior dentro do contexto navegacional.

As figuras 23, 24, 25, 26 e 27 representam as classes navegacionais originadas a partir das classes conceituais **Funções**, **Formas de Licença**, **Tópicos Avançados**, **Portabilidade** e **Comandos**. Assim como na classe navegacional **Portabilidade**, a única diferença destas classes para suas classes conceituais é o acréscimo do atributo **Volta**.

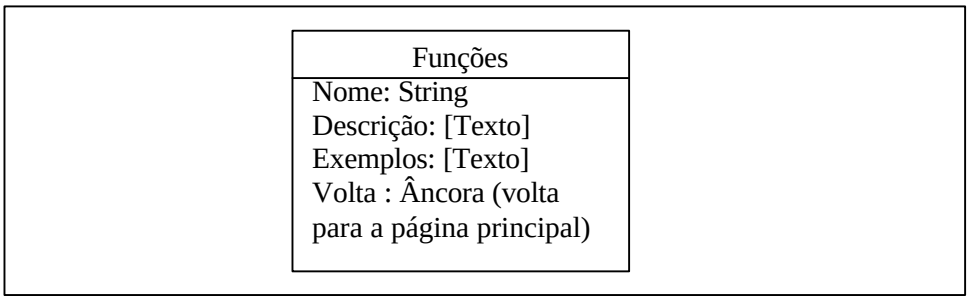


Figura 23 – Representação da classe navegacional "Funções"

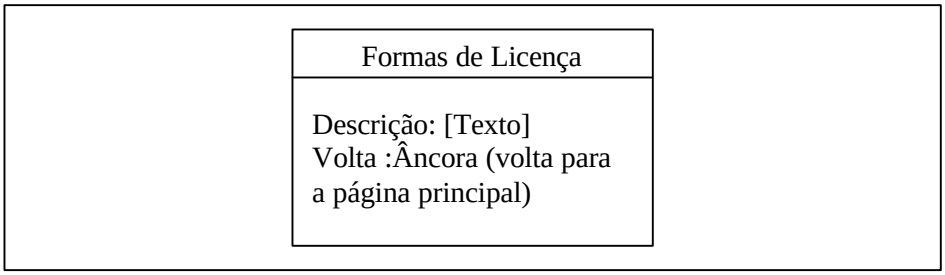


Figura 24 – Representação da classe navegacional “Formas de Licença”

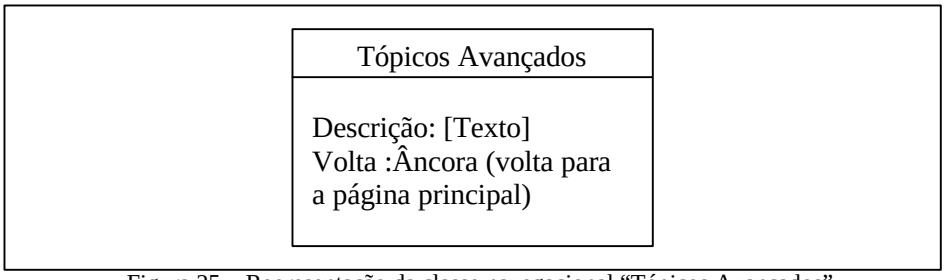


Figura 25 – Representação da classe navegacional “Tópicos Avançados”

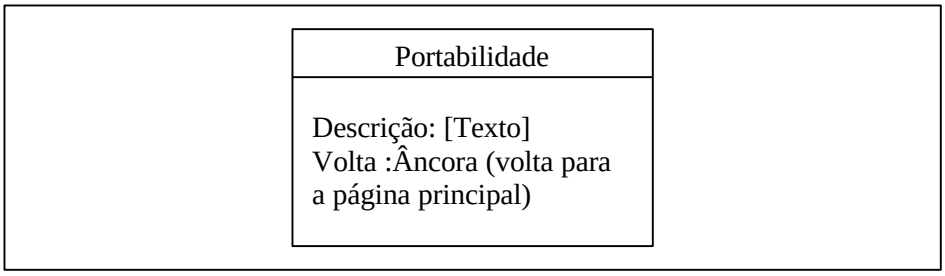


Figura 26 – Representação da classe navegacional “Portabilidade”

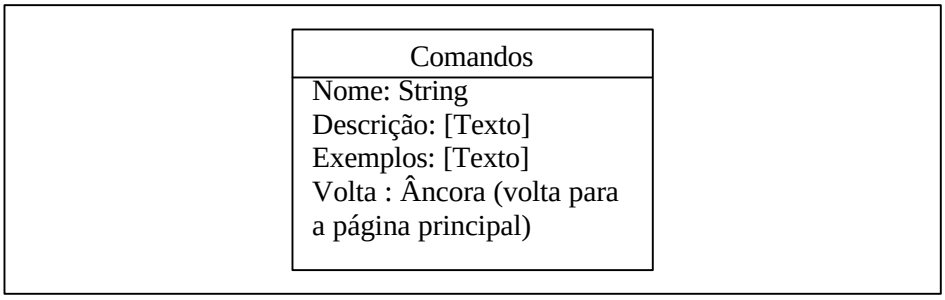


Figura 27 – Representação a classe navegacional “Comandos”

Agora será mostrado as classes navegacionais originadas das classes conceituais **Estruturas de Controle**, **Estruturas de Repetição** e **Tipos de Dados**, representadas através das figuras 28, 29 e 30. Note que nas suas sub-classes, o atributo **Volta**, é um elo de ligação com a classe anterior, ou seja, a classe "mãe", e não um elo com a classe principal, que é o menu principal da aplicação. Já nas classes "mães" (**Estruturas de Controle**, **Estruturas de Repetição** e **Tipos de Dados**), o atributo **Volta** funciona como um elo de ligação com a classe principal, a classe **A Linguagem**. Além disso, as classes "mães" possuem um atributo chamado **Itens**, que é uma estrutura de acesso para as suas sub-classes e pode ser melhor visualizado como um índice ou menu.

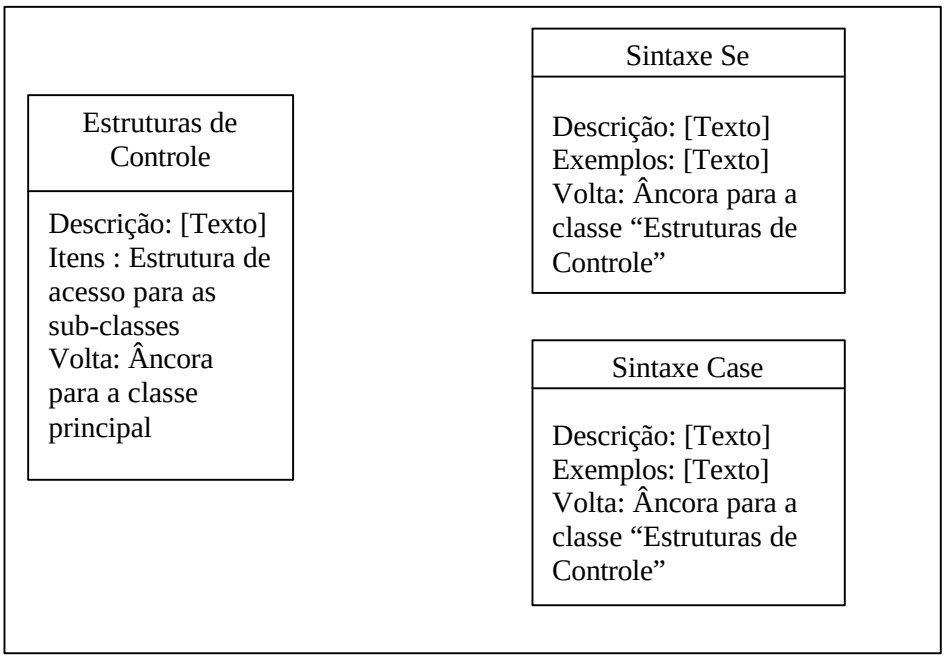


Figura 28 – Representação da classe navegacional "Estruturas de Controle" e suas sub-classes

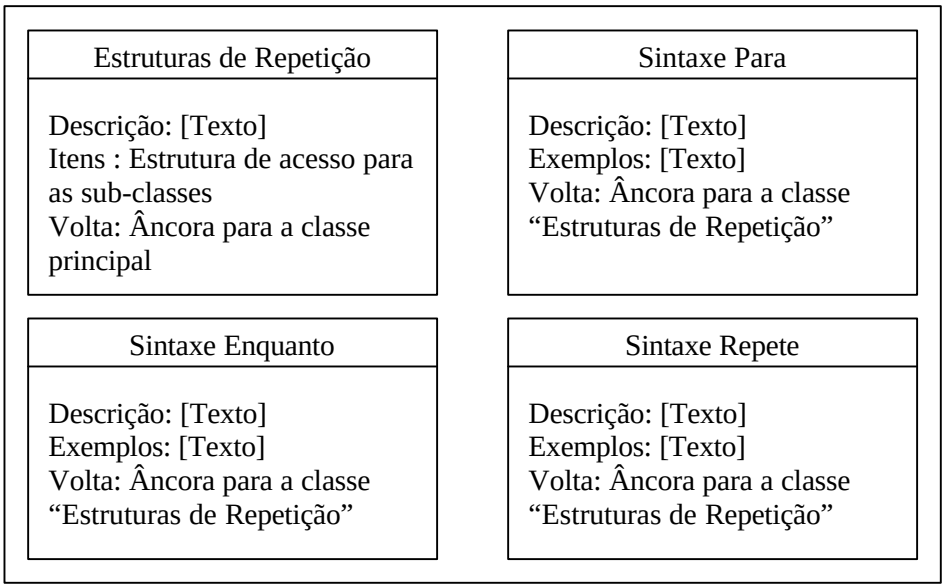


Figura 29 – Representação da classe navegacional "Estruturas de Repetição" e suas sub-classes

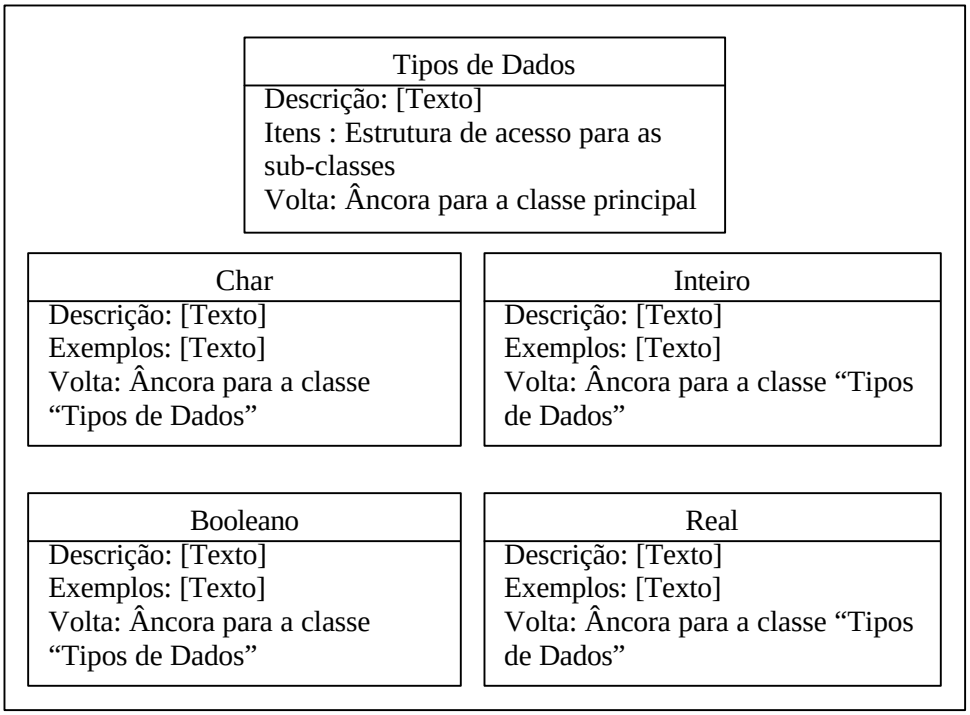


Figura 30 – Representação da classe navegacional "Tipos de dados" e suas sub-classes

A classe **A Linguagem**, que á a classe principal desta aplicação hipermídia, é representada possuindo o atributo **Índice**, que é uma estrutura de acesso a todas as outras classes da aplicação. Ele pode ser visualizado como um menu de opções.

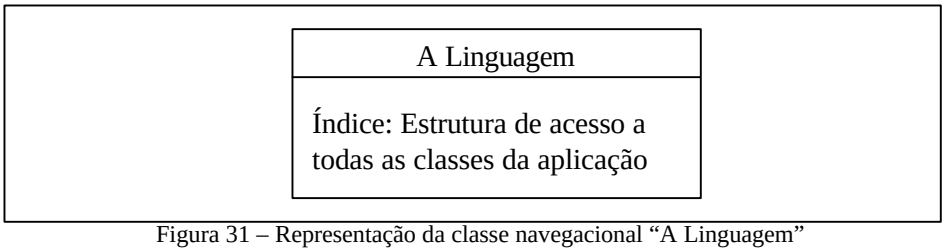


Figura 31 – Representação da classe navegacional “A Linguagem”

4.3 *Design* Abstrato da Interface

Depois de concluídos os dois primeiros passos do processo de construção de aplicações hipermídia, inicia-se o terceiro passo, ou seja, baseado nas estruturas definidas anteriormente (as classes conceituais e navegacionais), especifica-se como será o comportamento dinâmico da aplicação e como será feita a interface com o leitor.

Para a construção da interface, cria-se telas ou nós genéricos, determinando a posição das âncoras, dos títulos, dos espaços reservados aos atributos das classes e do comportamento da aplicação quando o usuário selecionar uma âncora.

A primeira tela a ser definida será a tela **A Linguagem**. Ela representará a estrutura global de acesso chamada **Índice**. O conjunto de botões representa os seletores das estruturas de acesso, cada um com o nome da classe a qual está associado.

Figura 32 – Interface da estrutura de acesso global “Índice”

Nesta interface, a frase **A Linguagem** pode ser substituída pelo nome da linguagem a ser realmente implementada. O retângulo onde contém a palavra “figura” é onde vai se colocar alguma figura que represente a linguagem a ser implementada.

Para o comportamento da aplicação quando for pressionado algum botão ou âncora, define-se uma regra geral que servirá para qualquer parte da aplicação:

- Após se pressionar um botão ou âncora, a tela ou nó correspondente deverá se sobrepor à tela original. Por exemplo, se clicar no botão **Histórico**, a tela correspondente à classe **Histórico** deverá se sobrepor à tela correspondente a tela anterior (que no caso é a tela do índice).

A próxima tela mostrada é a tela **Histórico**, através da figura 33. Observe o botão volta, no canto inferior direito da tela. Este botão será uma

constante nas telas que serão apresentadas, pois ele aparece em todas as classes navegacionais. O comportamento deste botão seguirá o comportamento geral definido.

As telas **Funções**, **Formas de Licença**, **Tópicos Avançados**, **Portabilidade** e **Comandos**, terão o mesmo comportamento da tela "**Histórico**" (elas serão acessadas através do índice principal e terão em suas telas o botão **Volta**). Elas serão representadas através das figuras 34, 35, 36, 37 e 38.

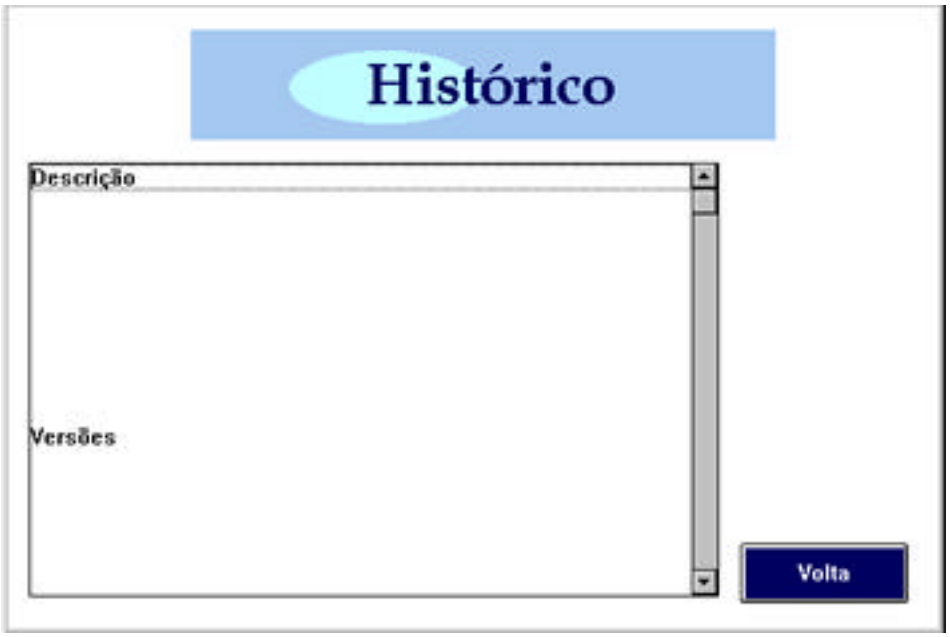


Figura 33 – Representação da tela abstrata "Histórico"

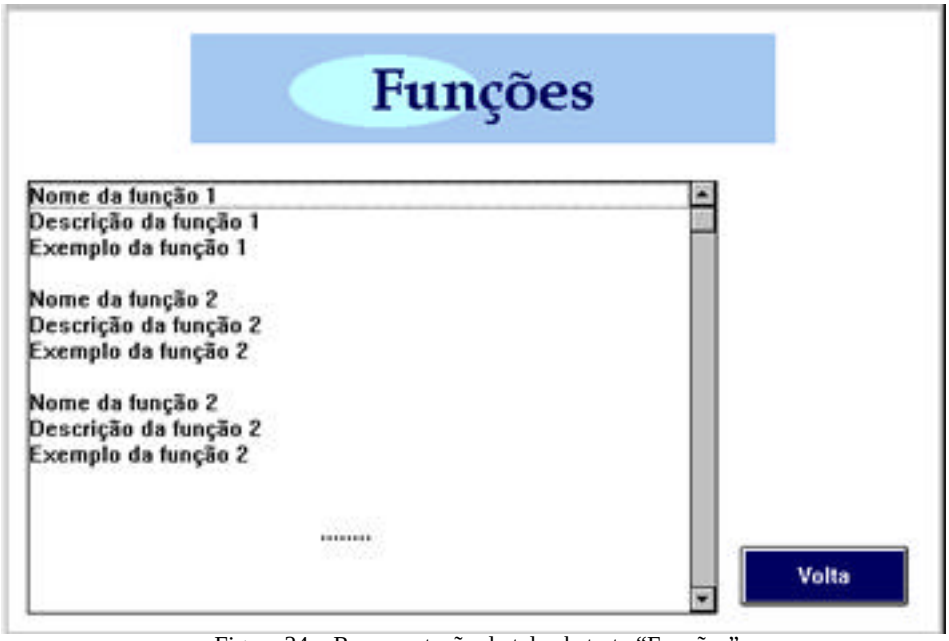


Figura 34 – Representação da tela abstrata “Funções”

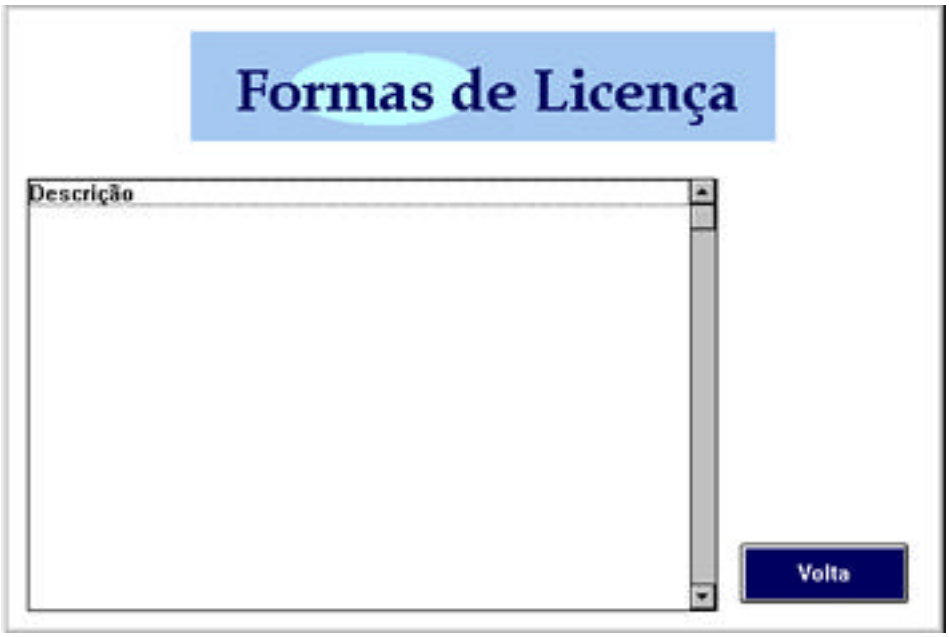


Figura 35 – Representação da tela abstrata “Formas de Licença”

Figura 36 – Representação da tela abstrata “Tópicos Avançados”

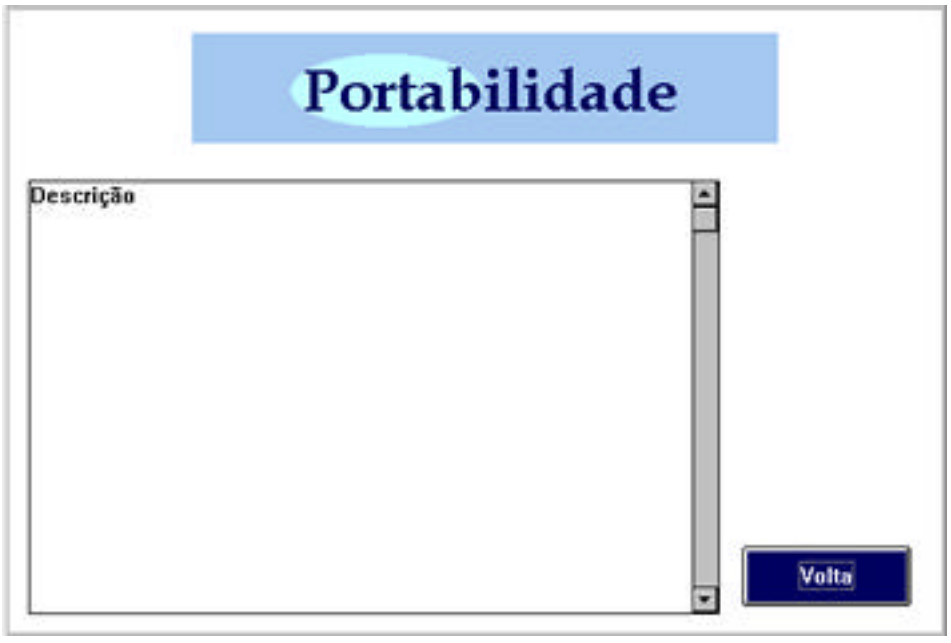


Figura 37 – Representação da tela abstrata “Portabilidade”

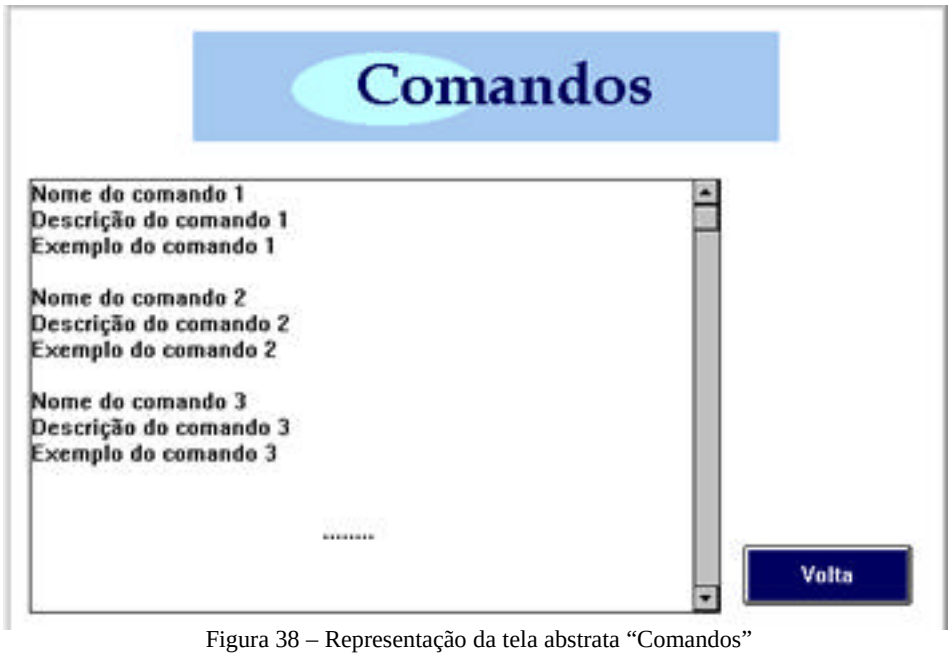


Figura 38 – Representação da tela abstrata “Comandos”

Algumas telas apresentadas a partir de agora terão uma pequena particularidade. As telas originadas a partir das classes de navegação **Estruturas de Controle**, **Estruturas de Repetição** e **Estruturas de Dados** serão acessadas a partir do índice principal, como as outras apresentadas. A diferença está nas telas originadas das sub-classes das classes referidas anteriormente. As telas originadas dessas classes serão acessadas através de menus especiais, contidos nas telas das suas classes “mães”, no lado direito superior. O botão **Volta** dessas telas, ao invés de fazer ligação com a tela do índice principal, vai fazer ligação com a tela anterior, ou seja, com a tela da classe que deu origem a sub-classe.

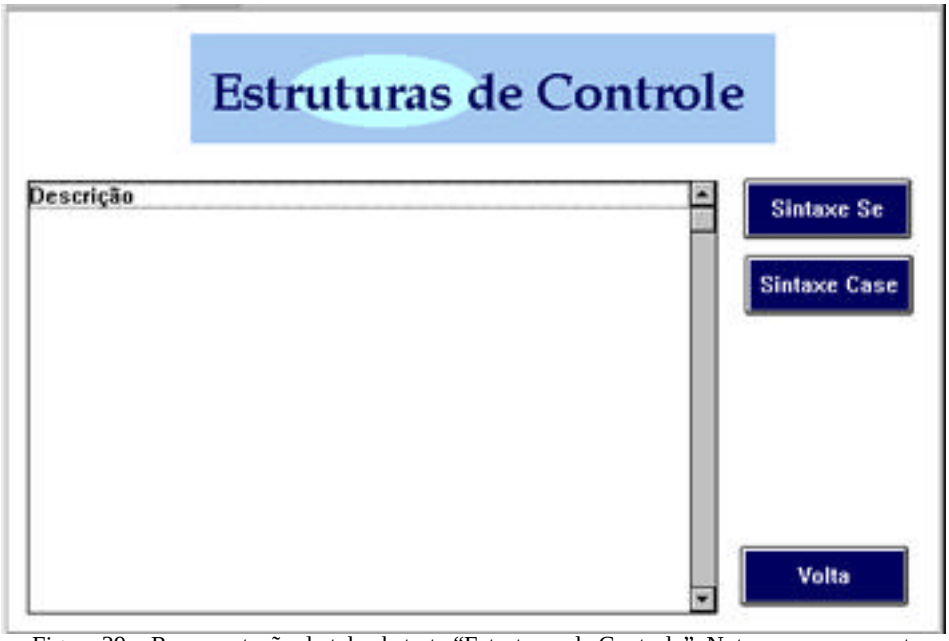


Figura 39 – Representação da tela abstrata “Estruturas de Controle”. Note o menu no canto superior direito

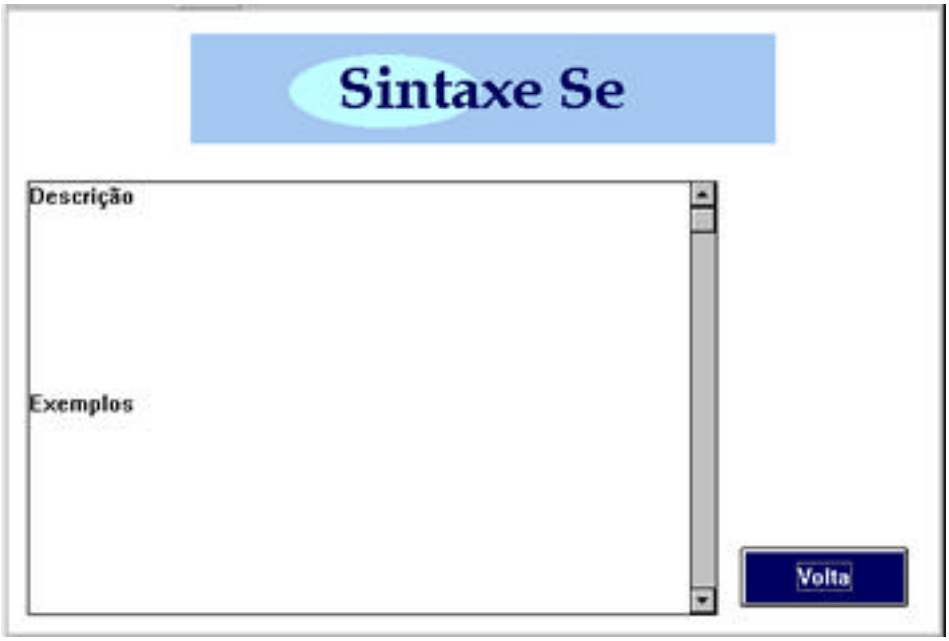


Figura 40 – Representação da tela abstrata “Sintaxe Se”

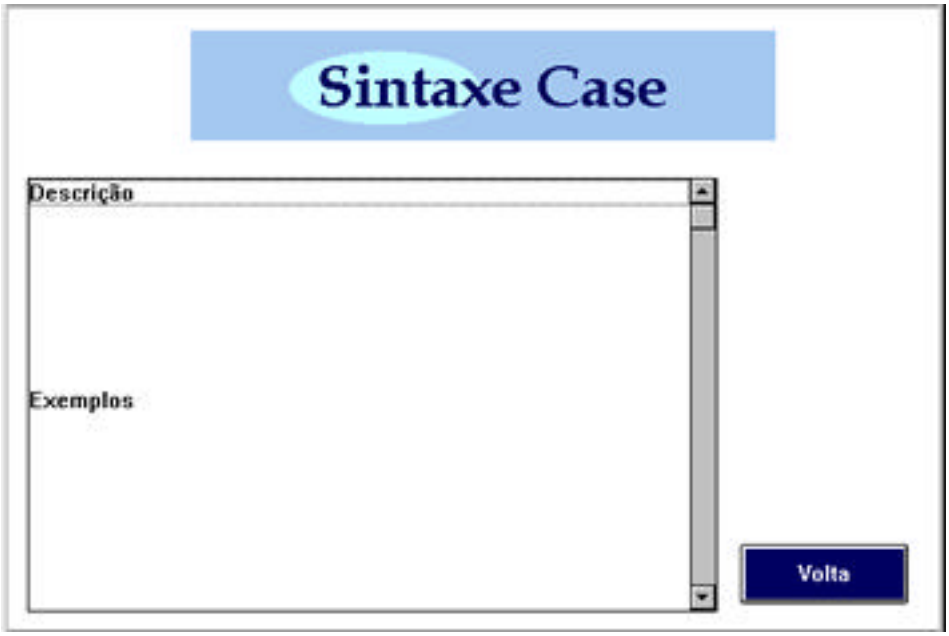


Figura 41 – Representação da tela abstrata “Sintaxe Se”

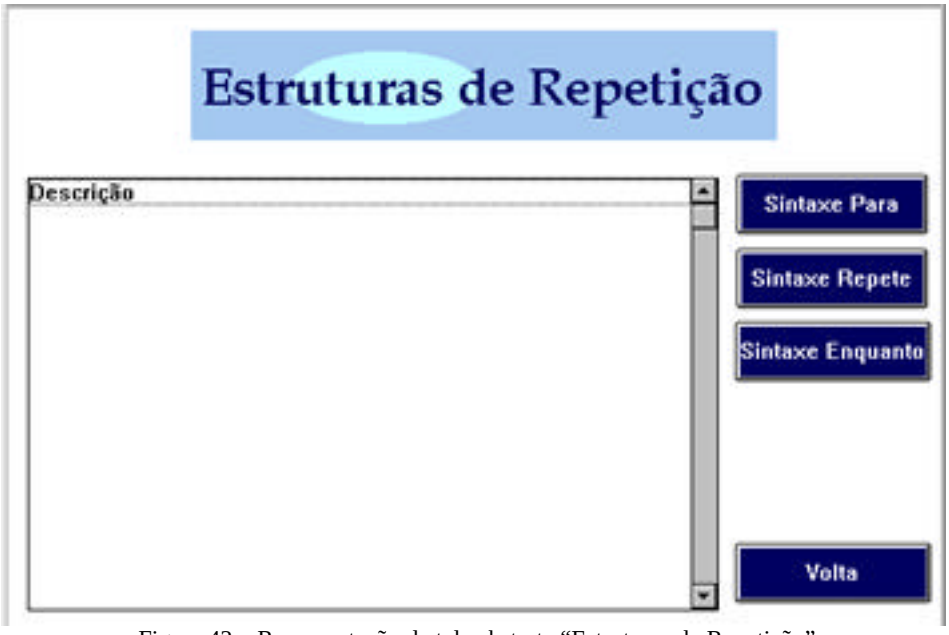


Figura 42 – Representação da tela abstrata “Estruturas de Repetição”

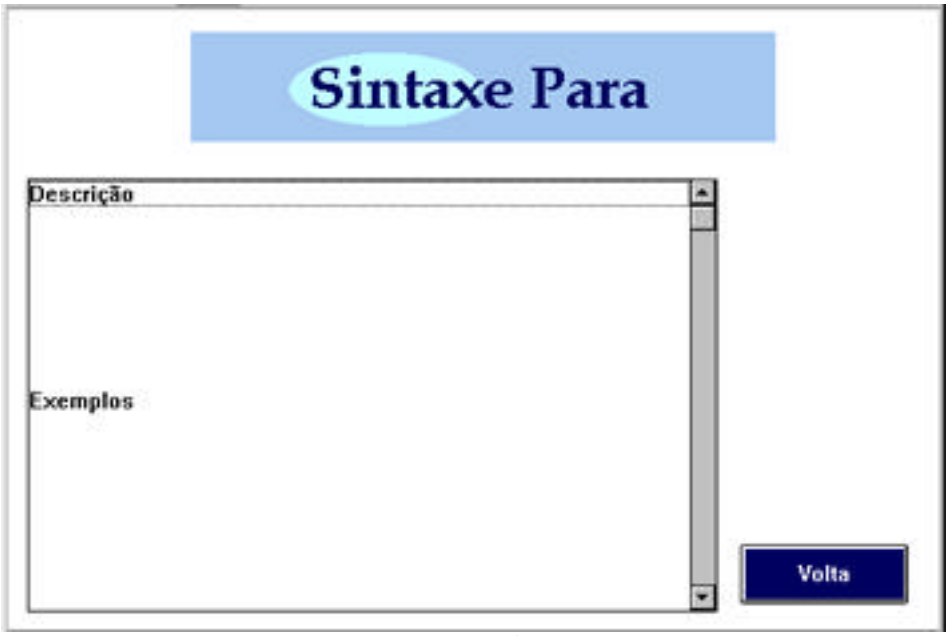


Figura 43 – Representação da interface abstrata “Sintaxe Para”

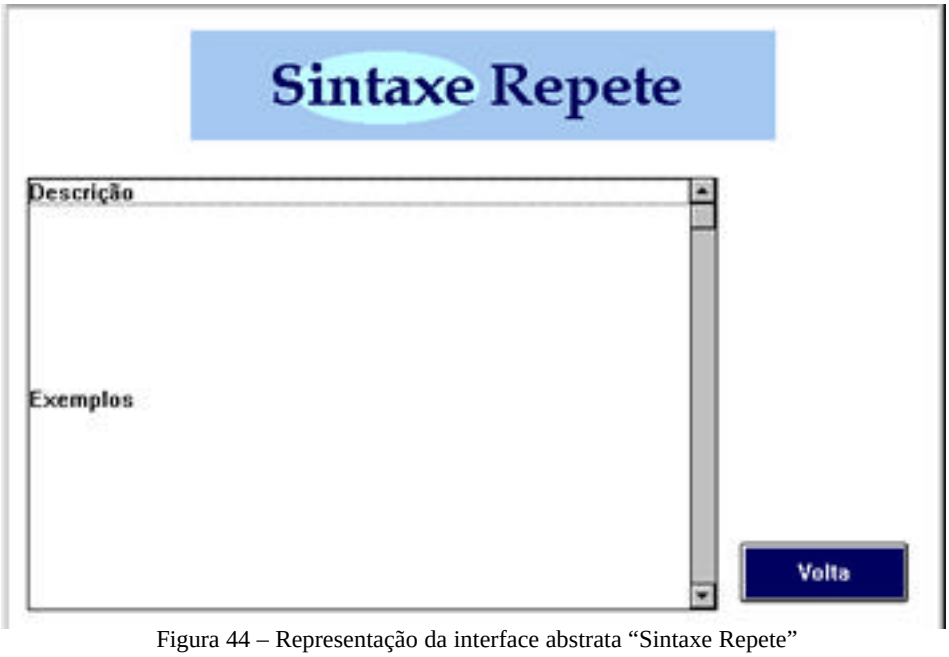


Figura 44 – Representação da interface abstrata “Sintaxe Repete”

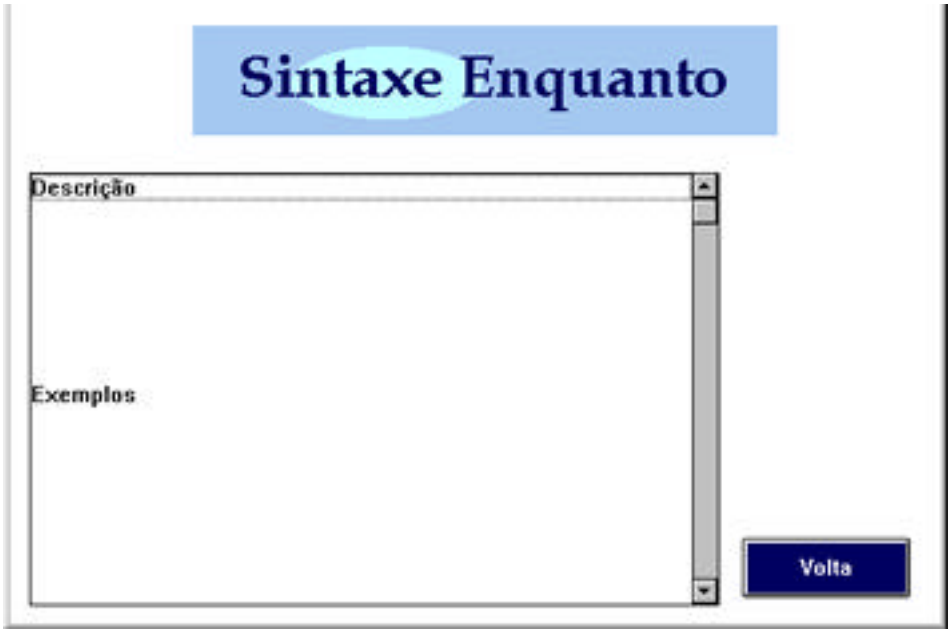


Figura 45 – Representação da interface abstrata “Sintaxe Enquanto”

Figura 46 – Representação da interface abstrata “Tipos de Dados”

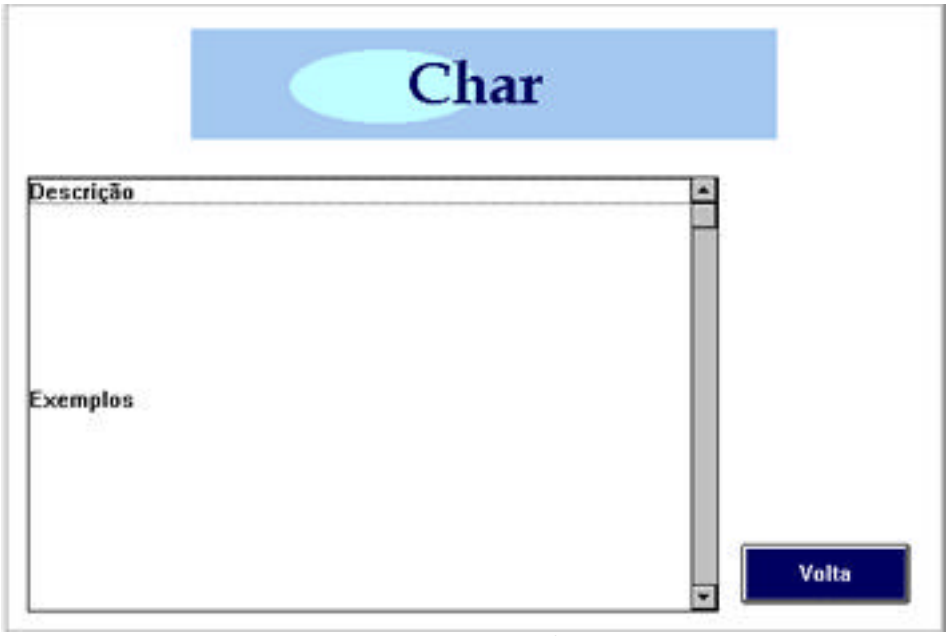


Figura 47 – Representação da interface abstrata “Char”

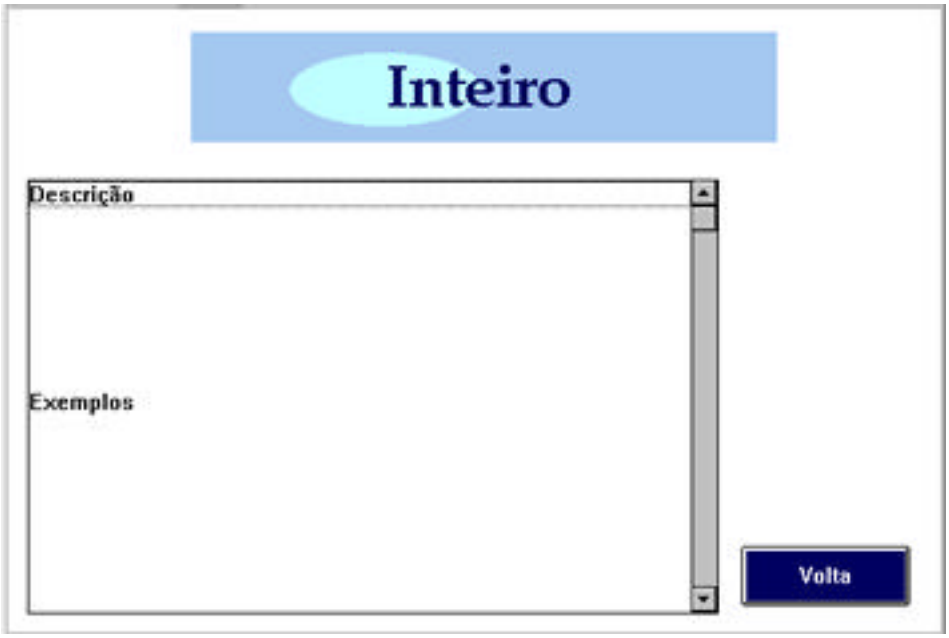


Figura 48 – Representação da interface abstrata “Inteiro”

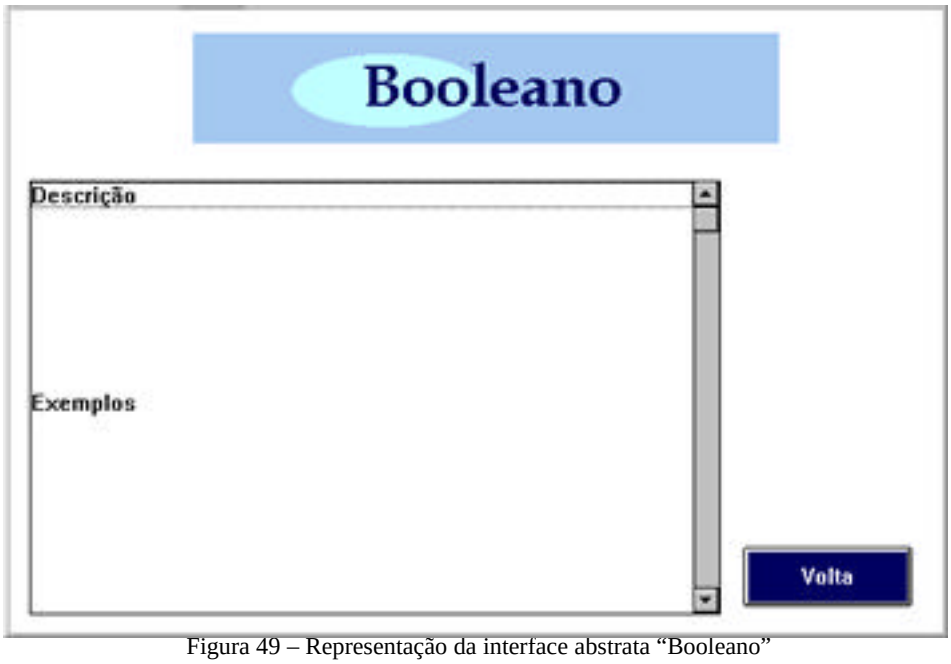


Figura 49 – Representação da interface abstrata “Booleano”

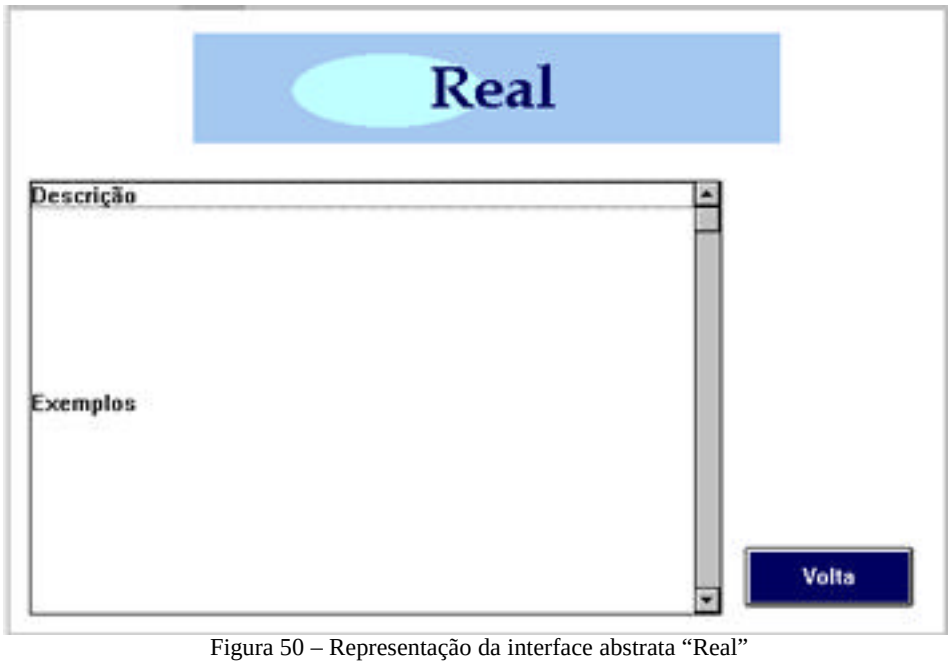


Figura 50 – Representação da interface abstrata “Real”

4.4 Implementação

Para dar ao leitor a idéia de como ficaria uma implementação baseada nas estruturas definidas nos tópicos anteriores, alguns textos e desenhos foram acrescentados às telas já projetadas. Além disso, foram feitas ligações (elos) entre elas.

A aplicação criada em Toolbook, para o ambiente Windows, se refere ao ensino da linguagem de programação DELPHI.

A interface definida para a estrutura de acesso global, o índice principal da aplicação, ficou como mostrado na figura 51. Note que o título (DELPHI) e uma figura que representa a linguagem foram acrescentados.

Nesta parte da implementação nem todas as telas serão mostradas. As figuras 51 e 52 mostram as telas **Portabilidade** e **Tópicos Avançados** respectivamente. As figuras 53, 54 e 55, mostram as telas **Estruturas de Controle**, **Sintaxe Se** e **Sintaxe Case**. As figuras 56 e 57 mostram as telas **Tipos de Dados** e **Booleano**.

Espera-se que o leitor, após serem mostradas algumas telas da aplicação hipermídia de ensino de linguagem de programação, perceba como é tranqüila a passagem de um passo a outro dentro do processo de construção de aplicações apresentado. Veja como é relativamente simples a implementação de uma outra aplicação sobre o ensino de uma outra linguagem de programação qualquer, tendo em mãos toda a documentação (objetos, classes, atributos, elos etc) definidos nos três primeiros passos de modelagem.

Figura 51 – Primeira tela da aplicação “DELPHI”

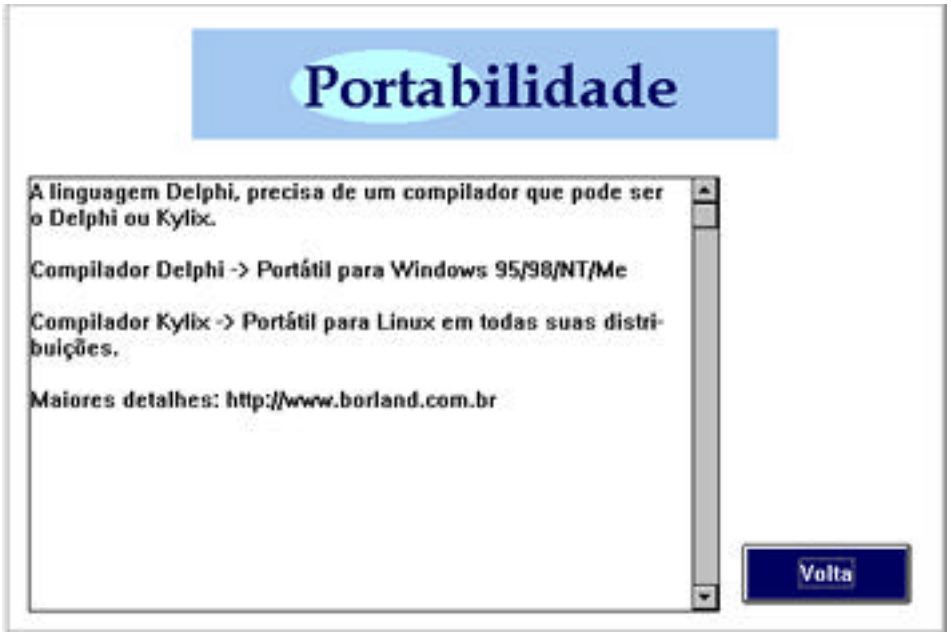


Figura 52 – Tela “Portabilidade” da aplicação “DELPHI”

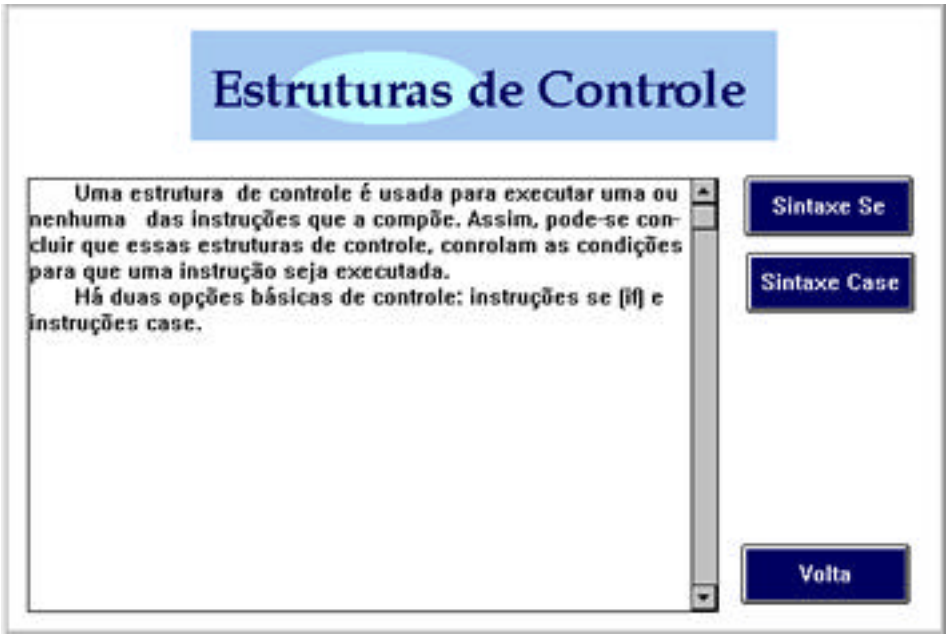


Figura 53 – Tela “Estruturas de Controle” da aplicação “DELPHI”

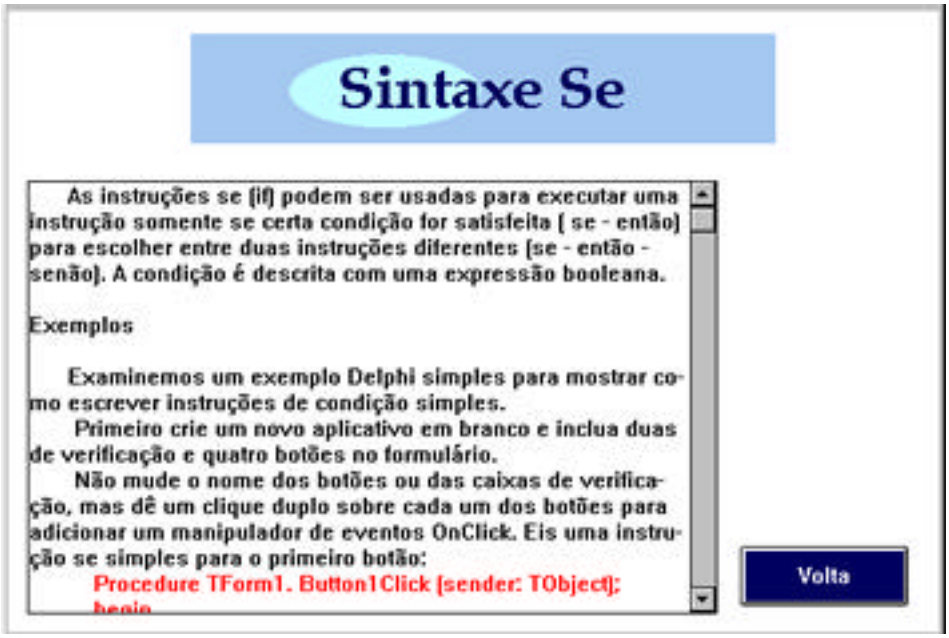


Figura 54 – Tela “Sintaxe Se” da aplicação “DELPHI”

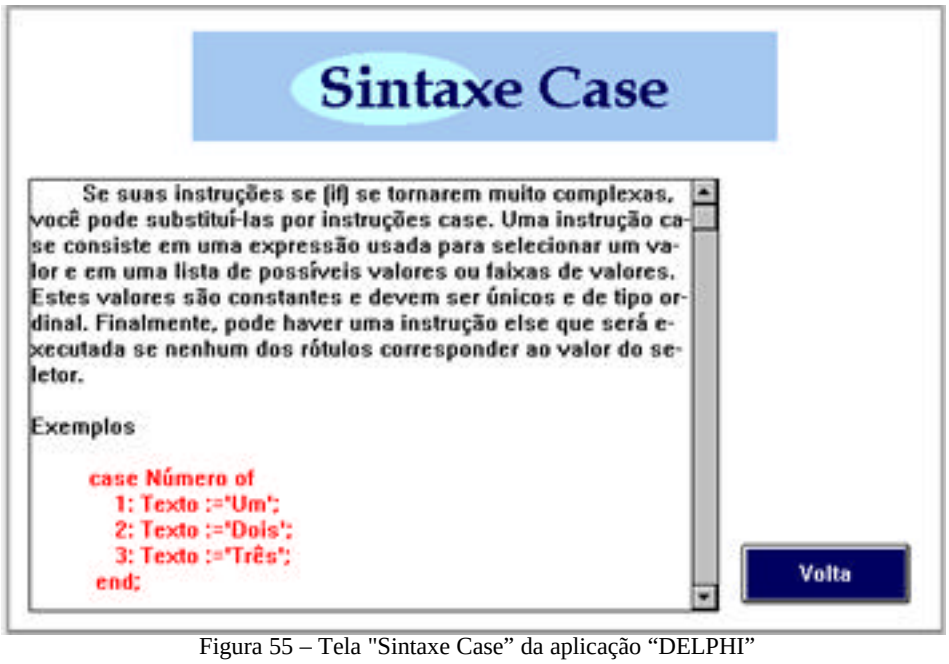


Figura 55 – Tela "Sintaxe Case" da aplicação "DELPHI"

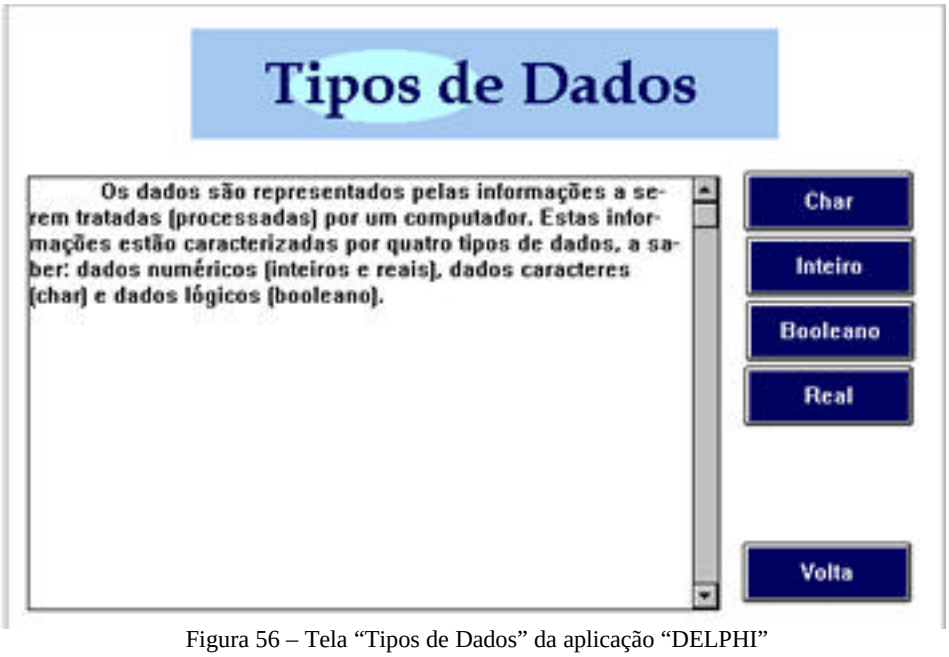


Figura 56 – Tela "Tipos de Dados" da aplicação "DELPHI"

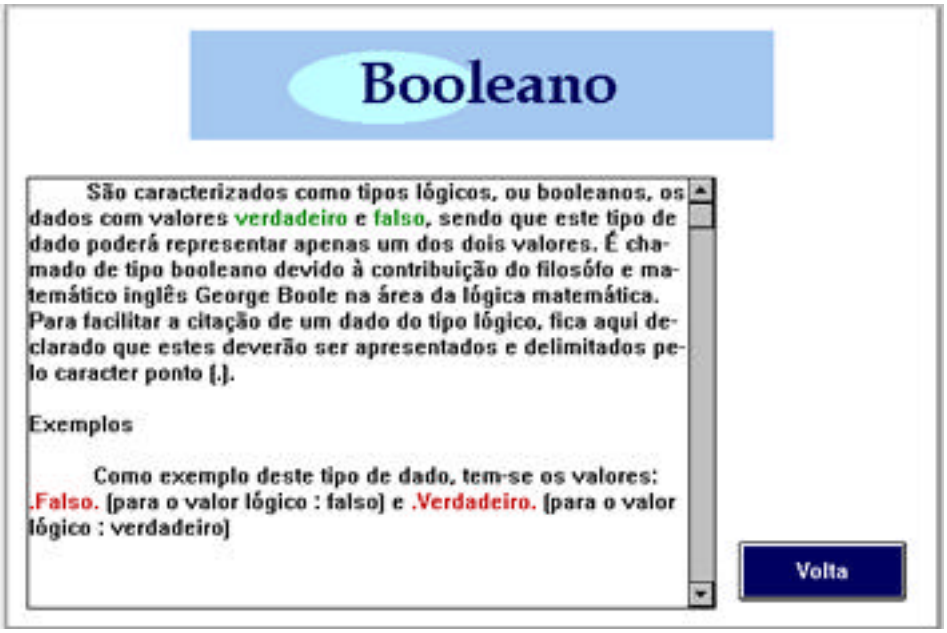


Figura 57 – Tela “Booleano” da aplicação “DELPHI”

CAPÍTULO V – CONCLUSÃO

No presente trabalho foram apresentados os principais conceitos relacionados à hipermídia e modelagem OOHDM. Desenvolveu-se uma aplicação hipermídia educacional, baseada em reuso e facilidade de manutenção.

Ao desenvolver a aplicação hipermídia, o OOHDM permitiu uma transição tranqüila desde a modelagem, passando pelo projeto de navegação e *design* abstrato da interface, até chagar-se à implementação propriamente dita, que inclusive pode ser feita em qualquer plataforma.

Uma característica muito importante do OOHDM vista ao final deste trabalho, foi a documentação gerada pelo processo de modelagem. Esta documentação é muito importante para um autor de hipermídia, pois, com ela, qualquer alteração a ser feita na aplicação ou, até mesmo, a sua reutilização, fica mais fácil de ser feita. O modelo OOHDM, independente da implementação, é capaz de estruturar aplicações grandes e complexas, facilitando a manutenção e dando suporte à reutilização.

Outra característica que também deve ser destacada é a flexibilidade que um esquema em OOHDM fornece, pois é possível o enriquecimento de aplicações particulares simplesmente com o uso de instâncias. Esta flexibilidade é conseguida devido ao poder de abstração que o modelo possui.

Ao longo de todo o estudo, observou-se que estruturar e implementar grandes aplicações hipermídia é ainda um campo aberto a estudos e que os mecanismos de análise orientada a objeto são uma boa forma de ser resolver os principais problemas que encontram-se quando se desenvolve aplicações deste tipo.

Espera-se com este trabalho, divulgar e proporcionar a outras pessoas um maior contato com o OOHDM e mostrar do que ele é capaz. Espera-se atender às diversas áreas de desenvolvimento, principalmente aquelas baseadas na *web*.

CAPÍTULO VI – REFERÊNCIAS BIBLIOGRÁFICAS

[BK95] Bieber, M. e Kacmar, C.: "Designing hypertext support for computational applications", Comm ACM, Agosto 1995, pp. 99-107

[BS98] Barroso, N.G. e Schwabe,D.: "Projeto de Navegação em aplicativos hipermídia orientado ao usuário", PUC-RJ, Março 1998.

[Car90] Carlson, P.A.: "Square books and round books: cognitive implications of hypertext",Academic Computing 4, 1990, pp.26-31.

[GPS91] Garzotto, F. , Paolini, P. e Schwabe, D.: "HDM - A Model for the Design of Hypertext Applications", Proceedings of Hypertext, 1991, ACM Press, pp. 313-320.

[Lim94] Lima,V.M.B.: "Autoria em hipermídia: o OOHDM", UFRJ, Dezembro 1994.

[Nil93] Nielsen, J.: "Usability Engineering", Chestnut Hill, 1993

[NN91] Nanard, J. e Nanard, M.: "Using Structured Types to Incorporate Knowledge in Hypertext", Third ACM Conferences on Hypertext Proceedings, Hypertext'91, ed. ACM Press, pp. 329-334.

[RB91] Rumbaugh, J. e Blaha, M.: " Object Oriented Modeling and Design", Prentice Hall Inc., 1991.

[Sch93] Schwabe, D.: “Autoria em hipermídia”, Versão preliminar PUC-RJ, Junho 1993.

[Sch98] Schneiderman, B.: “Design the user interface”, Addison-Wesley, 1998.

[SR94] Schwabe, D. e Rossi, G.: “OOHDM: An Object Oriented Hypermidia Design Model”, PUC-RJ , 1994.

[Str94] Streitz, N. A. : “Foundations of Hypermedia Design”, Designing user interfaces for hypermedia, Berlim, 1994.

[THH91] Thuring, M. , Haake J. M. e Hannemann, T.: "What's ELIZA doing in the Chinese room? Incoherent hyperdocuments and how to avoid them”, ACM Press, New York 1991, pp.161-177.

[THH95] Thüring, M., Hannemann, J., e Haake, J. M.: “ Hypermedia and Cognition: Designing for Comprehension”, Communications of the ACM, 1995.

[VHN97] Vieira, F., Hasegawa, R. e Nunes, M.G.: “SASHE: autoria de aplicações hipermídia para o ensino”, ICMSC-USP, 1997.

[ZAL99] Zambalde, A. L., Alves, R.M. e Lopes, M.A.: “Modelagem, autoria e análise de usabilidade de aplicação hipermídia direcionada ao setor agropecuário”, UFLA, 1999.

[ZP92] Zheng, Y. e Pong, M.C.: “Using Statecharts to Model Hypertext”, Proceedings of the ACM European Conference on Hypertext, Milano, Dezembro 1992.