UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO

APLICAÇÃO DA ENGENHARIA DA USABILIDADE PARA O DESENVOLVIMENTO DE UM SOFTWARE MULTIMÍDIA.

ROBERTO KLOSOWSKI MACHADO

BLUMENAU
2004

2004/1-32

ROBERTO KLOSOWSKI MACHADO

APLICAÇÃO DA ENGENHARIA DA USABILIDADE PARA O DESENVOLVIMENTO DE UM SOFTWARE MULTIMÍDIA.

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Carlos Eduardo Negrão Bizzotto, Dr. – Orientador

BLUMENAU
2004

2004/1-32

APLICAÇÃO DA ENGENHARIA DA USABILIDADE PARA O DESENVOLVIMENTO DE UM SOFTWARE MULTIMÍDIA.

Por

ROBERTO KLOSOWSKI MACHADO

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Carlos Eduardo Negrão Bizzotto, Dr. – Orientador, FURB
Membro:	Prof. Paulo de Tarso Mendes Luna, Ms
Membro:	Prof. Luiz Bianchi, Ms

Blumenau, 4 de agosto de 2004

Dedico este trabalho aos meus pais e a todos os amigos, especialmente aqueles que me ajudaram diretamente na realização deste.

“Não sabendo que era impossível, ele foi lá e fez.”

Lao-Tsé

AGRADECIMENTOS

Todo projeto é resultado da interação entre muitas pessoas. Amigos e amigas que contribuíram com seus conhecimentos e empolgações para o sucesso dessa jornada.

O professor Carlos Eduardo Negrão Bizzotto foi mais do que um orientador, um verdadeiro companheiro de jornada, indicando o "caminho das pedras".

Agradeço especial aos meus pais, por sempre acreditarem na minha capacidade e perseverança, motivando-me e auxiliando-me em todos os momentos da minha vida, dando-me toda estrutura necessária para minha formação.

Um agradecimento em especial a compreensão de minha namorada, pelas sucessivas ausências durante a realização deste trabalho, e agradeço-a também, por sempre me motivar e acreditar no meu potencial.

Aos meus amigos, meu sincero agradecimento e gratidão, pela amizade, carinho e respeito, que se tornaram presentes durante estes longos anos.

RESUMO

O presente trabalho aborda o desenvolvimento de um software na forma de um jogo multimídia que tem como finalidade auxiliar no processo de ensino-aprendizagem da informática, utilizando técnicas de engenharia da usabilidade. Estas técnicas propõem modificações para melhorar a qualidade e ergonomia, ampliando horizontes de controle da interação por parte dos usuários, através de telas e objetos que sejam compatíveis com as necessidades das tarefas a serem realizadas. A avaliação da adequação destas técnicas foi feita através de um sistema para o aprendizado do *mouse*. Para isso, foram utilizados para realização deste trabalho o software de autoria Macromedia Director 8 e a ferramenta Rational Rose.

Palavras chaves: Monografia; Usabilidade; Ergonomia; *Mouse*.

ABSTRACT

The main objective of this assignment is the development of a software in a form of multimedia game witch has the finality to support in computers teaching-learning process, using engineering techniques of usability. This techniques propose changes to increase the quality and the ergonomics, extending the horizon of interaction users controls in a user point of view, through of screens and objects which will be compatibles with the task needs. The validation of this techniques was done through a mouse learning system, to implement this was used the Macromedia Director and the CASE tool Rational Rose.

Key-Words: Monograph, Usability, Ergonomics, Mouse.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

ACE– Associação Catarinense de Ensino

AADAV - Associação Assistencial dos Deficientes Auditivos e Visuais

IHC – Interação Homem-Computador

SUMÁRIO

1 INTRODUÇÃO

A sociedade atual vem passando por grandes transformações, dentre as quais pode-se destacar a relação entre trabalho e educação. Há poucas décadas, era comum aos profissionais das diferentes áreas atuarem no mercado de trabalho com os conhecimentos que adquiriam na universidade. Atualmente, no entanto, esta abordagem não tem produzido resultados positivos em função, dentre outros fatores, da velocidade da evolução da tecnologia.

Nesse sentido, não há mais uma separação temporal entre formação (educação) e trabalho, ou seja, a atualização permanente deixa de ser um diferencial para se transformar em uma necessidade. Em função disso, diferentes metodologias, modalidades e tecnologias vêm sendo desenvolvidas de forma que um maior número de profissionais possam se atualizar.

Dentre as opções disponíveis, pode-se destacar o uso de softwares educacionais, na forma de jogos, como ferramentas auxiliares no processo de ensino-aprendizagem. Estes tipos de softwares educacionais podem melhorar a motivação do aluno, aumentando assim, a qualidade do processo de aprendizagem.

Segundo Abt (1974), a motivação em jogos de simulação educacional é amplamente reconhecida. Torna-se importante ressaltar, no entanto, que a motivação por parte do aluno é necessária, mas não suficiente para que a aprendizagem ocorra. Nesse sentido, para contribuir para o desenvolvimento do aluno, um software educacional, na forma de jogo, deve estar fundamentado em uma pedagogia, cujos princípios sejam compatíveis com aqueles adotados em Bizzotto (2003). Caso contrário, o computador assume o papel de uma máquina de ensinar, onde a interação entre o aprendiz e o computador consiste na leitura da tela ou escuta da informação fornecida e avanço pelo material, apertando a tecla ENTER ou usando o *mouse* para escolher a informação.

Por envolver o aprendiz em uma competição, os jogos educacionais podem dificultar o processo de aprendizagem uma vez que, enquanto estiver jogando, o interesse do aprendiz está voltado para ganhar o jogo e não em refletir sobre os processos e estratégias envolvidos no mesmo, conforme ressalta Wendt (2002). Sem essa consciência é difícil uma transformação dos esquemas de ação em operação. Para evitar esse problema, é importante que o jogo combine atividades que façam o aluno refletir sobre suas ações e estratégias.

Sendo assim, conforme Martins (2002), na quase totalidade das vezes, os jogos educacionais são utilizados com crianças e adolescentes. No entanto, os jogos também podem ser uma grande ferramenta para o aprendizado de adultos. Nesse sentido, podem substituir os tutoriais no ensino dos aspectos básicos de informática e na utilização de softwares.

É dentro deste contexto que se enquadra o presente trabalho, cujo objetivo é o desenvolvimento de um software educacional, na forma de jogo, o qual abordará o uso do *mouse*. No aprendizado do uso do *mouse* serão desenvolvidos jogos que permitam o desenvolvimento da coordenação motora do aluno para clique, clique duplo e clique mover. Assim em todas as atividades, o software fará um monitoramento do desempenho do usuário, de forma a sustentar a repetição ou início de uma nova atividade.

No desenvolvimento do software proposto serão utilizados os fundamentos da Engenharia da Usabilidade, conforme proposto por Cybis (2002). O objetivo da utilização destes fundamentos é o desenvolvimento de um software que seja fácil de aprender e de utilizar.

Para que se possa validar sua adequação ao usuário final, o software será aplicado junto a um grupo de usuários em potencial. Com isso, será possível verificar a efetividade tanto do software desenvolvido quanto da metodologia utilizada para o desenvolvimento.

1.1 OBJETIVOS DO TRABALHO

O presente trabalho tem como objetivo o desenvolvimento de um software educacional interativo, na forma de jogo, que auxiliará no aprendizado do uso do *mouse*, tomando como base fundamentos teóricos da engenharia da usabilidade.

Além desse objetivo, o trabalho possui também os seguintes objetivos específicos:

a) estimular o aprendizado no uso do *mouse*, o que inclui o clique, clique duplo e clique mover;
b) monitorar o desempenho do aprendizado em cada um dos módulos;
c) utilizar e empregar conceitos propostos pela engenharia de usabilidade;
d) análise/teste da efetividade do aprendizado com o software produzido.

1.2 ESTRUTURA DO TRABALHO

Este trabalho está dividido em seis capítulos:

O primeiro capítulo apresenta a introdução do trabalho, justificando a criação dos objetivos proposta pelo mesmo.

O segundo capítulo faz uma abordagem sobre Software Educacional, abrangendo seu histórico, a evolução dos jogos e os tipos de softwares educacionais.

O terceiro capítulo apresenta uma visão geral sobre os aspectos de ergonomia e interface, focando conceitos básicos, princípios de abordagens e deficiências de aplicações.

O quarto capítulo apresenta toda fundamentação teórica, demonstrando as etapas e métodos de aplicação da engenharia da usabilidade.

O quinto capítulo apresenta uma breve abordagem da ferramenta CASE Rational Rose utilizada para especificação do software, além dos conceitos e técnicas sobre o ambiente de desenvolvimento utilizado: o Director 8. Neste mesmo capitulo serão apresentados às etapas para construção do software, especificando a estrutura do sistema, seu processo de desenvolvimento e algumas telas representativas do software.

O sexto capítulo traz as conclusões e sugestões para futuros trabalhos.

2 SOFTWARES EDUCACIONAIS

Segundo Ferreira (1995) o software educacional é geralmente classificado como subconjunto de sistemas de treinamento. Existe claramente uma interseção entre os dois conjuntos, mas a maior parte dos softwares educacionais não pode ser exatamente descrita como sistemas de treinamento, e vice-versa.

Conforme Campos (1994) o software educacional vem entrando no mercado mundial de forma muito acelerada. Inúmeros países como Inglaterra, França e EUA, entre outros, desenvolvem projetos de uso de microcomputador em educação e, conseqüentemente necessitaram desenvolver produtos de software específicos para suas necessidades.

O mesmo tem ocorrido no Brasil, onde diversos projetos de pesquisa vêm sendo desenvolvidos não só relacionados ao uso do microcomputador e sala de aula como, também, ao desenvolvimento de software para os mais diversos conteúdos programáticos. Alguns grupos de pesquisa utilizam o termo software educacional, ou software educativo, outros o termo *courseware*, outros, ainda, o termo programas educativos por computador. Todos estes termos possuem um mesmo significado: material educacional para microcomputadores.

O objetivo do presente capítulo é apresentar uma breve introdução da evolução histórica da informática na educação, o conceito de multimídia, as etapas de como um software educacional vai se desenvolvendo, qualidade de software educacional, além de descrever e exemplificar alguns tipos de softwares educacionais que existem atualmente.

2.1 HISTÓRIA DA INFORMÁTICA NA EDUCAÇÃO

Desde a década de 70, a educação no Brasil tenta alcançar o *status* de tratamento científico com o uso de tecnologia instrucional. Muito se tentou, para realizar o ensino individualizado, onde cada aluno seguiria de acordo com seu próprio ritmo e tivesse à sua disposição um *feedback* sobre as respostas desejadas, conforme ressaltados por Fagundes (1992).

Segundo Peixoto (1995, p. 01), no Brasil, o primeiro evento que tratou do tema Informática e Educação foi um seminário sobre a utilização de computadores no ensino de física, sob a assessoria de um especialista da Universidade de Dartmouth (EUA), realizado em 1981, na Universidade de São Carlos, São Paulo. No entanto, ações de abrangência nacional

só ocorreram a partir de 1981, com a realização do 1º Seminário Nacional de Informática na Educação, em Brasília.

Conforme Peixoto (1995, p. 1), a realização de pesquisas sobre o ensino por computador aparece como resultado dos subprojetos financiados pelo Projeto EDUCOM. Criado em 1983 pela Secretaria Especial de Informática da Presidência da República e pelo Ministério da Educação, sua implantação definitiva ocorreu apenas em 1985, com a instalação de cinco centros de pesquisa em universidades: Universidade Federal de Pernambuco, Universidade Federal de Minas Gerais, Universidade Federal do Rio de Janeiro, Universidade de Campinas e Universidade Federal do Rio Grande do Sul.

Assim este Projeto denominado EDUCOM [1] originou o Programa Nacional de Informática Educativa – PRONINFE, lançado em 1989, para apoiar o desenvolvimento e a utilização de novas tecnologias de informática no ensino fundamental, médio e superior e na educação especial. Tanto o EDUCOM quanto o PRONINFE não chegaram à escola de ensino básico, permanecendo no campo experimental em universidades, secretarias de educação e escolas técnicas, conforme ressaltados por MEC (2001).

2.2 EVOLUÇÃO DOS SOFTWARES EDUCACIONAIS

Segundo Silveira (2002), muitos softwares disponíveis no mercado que são ditos educacionais, não apresentam relevância pedagógica. Possuem, muitas vezes, apenas umas interfaces agradáveis, coloridas, mas são pobres no seu conteúdo. Além disso, exploram, na maioria das vezes, a repetição de exercícios, caracterizando os chamados CAI (*Computer Aided Instruction*) - Instrução Auxiliada por Computador, que são classificados como os mais simples softwares educacionais. Um dos tipos de software educacional que tenta fugir à monotonia dos CAI's são os jogos educativos computadorizados. Através de um jogo, o aluno pode "aprender" pelo conteúdo embutido no jogo. Os jogos educativos computadorizados são preferidos pelos alunos por despertar neles o interesse em "brincar", ou seja, o aluno "aprende brincando".

Conforme Ferreira (1995), multimídia computacional é a união de várias mídias, com o intuito de fixar ou transmitir mais facilmente uma informação. Assim o Software

[1] Nome dado ao projeto criado em 1983 pela Secretaria Especial de Informática (SEI) visando a implantação de centros-piloto em universidades públicas, voltados à pesquisa no uso de informática educacional.

Multimídia pode ser considerado também como um "livro eletrônico", pois, contém textos, imagens, sons, animações e filmes interligados.

Ainda segundo Ferreira (1995), a utilização da multimídia no desenvolvimento de software educacional trouxe inúmeras vantagens. Um software educacional com recursos multimídia pode tornar o aprendizado mais agradável e interessante, devido à possibilidade da inclusão de sons, fotos, imagens e animações, entre outras mídias. Este tipo de software pode contribuir no processo de ensino e aprendizagem, tornando as aulas menos monótonas e despertando no aluno o interesse à investigação e à descoberta. Um software com recursos multimídia também pode ser elaborado para que o aluno “aprenda brincando”. Isto se torna possível através da utilização de jogos educativos. Estes jogos educativos podem utilizar-se de recursos multimídia e tornam o aprendizado muito mais interessante e divertido. Estes jogos são elaborados para divertir os alunos e aumentar a chance de aprendizagem de conceitos, conteúdo e habilidades embutidas no jogo.

2.3 TIPOS DE SOTWARES EDUCACIONAIS

Os tipos mais comuns de softwares educacionais ou, também chamados de ambientes de aprendizagem interativa são: tutorial, exercício e prática, ambientes de autoria, simulação e jogos. A seguir será abordada uma breve explicação de cada um dos tipos de software.

2.3.1 SIMULAÇÃO

Os softwares educacionais do tipo simulação reproduzem situações reais para facilitar o aprendizado do aluno. Através de demonstrações que imitam a realidade, o aluno poderá aprender virtualmente. Uns exemplos de software educacional de demonstração são Simuladores de Vôos, aplicados atualmente no instituto da aeronáutica. Este multimídia tem por objetivo reproduzir, com total fidelidade, uma coletânea das principais situações e experiências que podem ocorrer durante um vôo. Cada experiência vem acompanhada de um breve resumo teórico, de animações e de vídeos ilustrativos. O usuário terá ao seu dispor os manuais de operação dos instrumentos utilizados nas experiências, assim como uma descrição dos materiais que as compõem. A figura 1 mostra exemplo de um simulador aplicado para testes na aeronáutica.

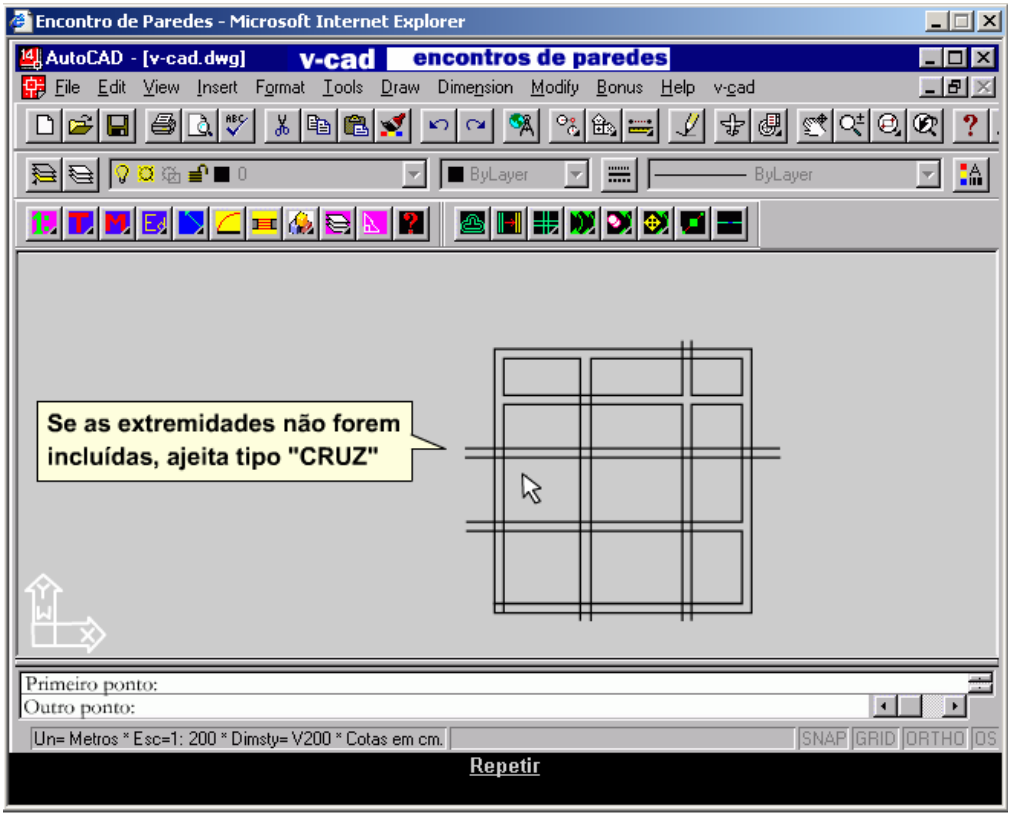
Fonte: CPV (2004)

Figura 1 - Tela simulador: parte interna BOING 727

2.3.2 TUTORIAL

Os programas tutoriais, conforme ressalta Valente (1993), constituem uma versão computacional da instrução programada. A vantagem dos tutoriais é o fato de o computador poder apresentar o material com outras características que não são permitidas no papel como: animação, som e a manutenção do controle da performance do aprendiz, facilitando o processo de administração das lições.

Segundo Wendt (2002, p. 9), a informação que está disponível para o aluno é definida e organizada previamente, assim o computador assume o papel de uma máquina de ensinar. A interação entre o aprendiz e o computador consiste na leitura da tela ou escuta da informação fornecida e avanço pelo material, apertando a tecla ENTER ou usando o *mouse* para escolher a informação. A figura 2 mostra exemplo de um tutorial.



Fonte: V-CAD (2000)

Figura 2 – Tutorial na Internet do Software Autocad

2.3.3 EXERCÍCIO E PRÁTICA

O software educacional de exercício e prática tem como característica ensinar através da repetição de exercícios propostos, fazendo com que o aluno aprenda praticando e visualizando seu desempenho no término de cada exercício. Um exemplo interessante é o software que aborda a atividade de completar espaços. Este software é produzido pelo Software Clic, próprio para ensinar na alfabetização de crianças, onde o aluno pode também programar atividades de ditado sonoro, pois o programa aceita vários tipos de sons. A figura 3 mostra uma das telas desta aplicação.

Fonte: Centrorefeducacional (1998)

Figura 3 – Tela do software – atividade de completar espaços

2.3.4 AMBIENTE DE AUTORIA

Os Softwares de Autoria desenvolvem a criatividade do aluno que trabalha como o Autor. Neste tipo de material pode-se trabalhar tanto com a exposição de dados, quanto com a construção do conhecimento.

Os próprios professores ou alunos, com esse tipo de software, podem desenvolver suas aplicações sem que seja necessário conhecer nenhuma linha de código de programação.

Conforme Valente (1993), um bom exemplo de software educacional do tipo autoria é o *LOGO*, onde não é só o nome de uma linguagem de programação, mas também de uma filosofia que lhe é subjacente. A filosofia surgiu dos contatos de Papert com a obra de Piaget e dos estudos sobre o problema da inteligência artificial.

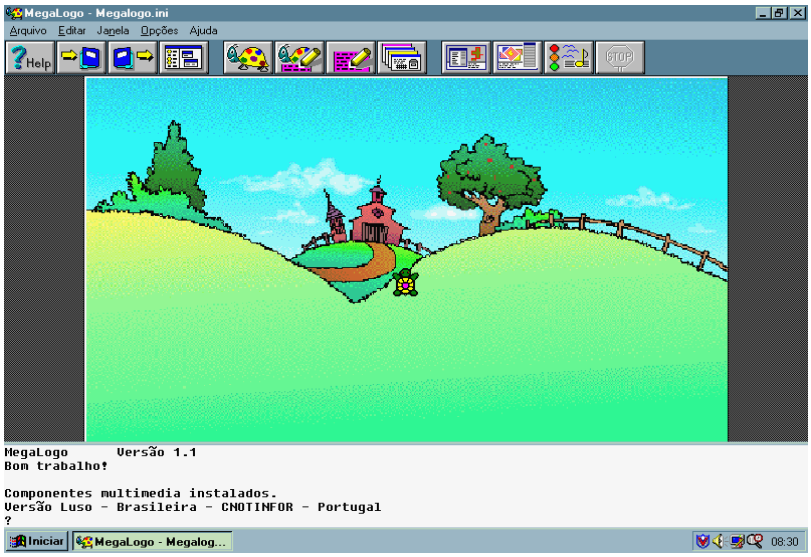
A visão que Papert tem do homem e do mundo situa-se numa perspectiva interacionista, sendo o conhecimento o produto dessa interação, que é centrada nas formas com que o mundo cultural age e influencia o sujeito em interação com o objeto. Ao contrário de Piaget, Papert enfatiza que aquilo que aprendemos e o como aprendemos depende dos materiais culturais que encontramos à nossa disposição.

Conforme Valente (1993), o *LOGO* propõe um ambiente de aprendizagem no qual o conhecimento não é meramente passado para o aluno, mas, uma forma de trabalho onde esse aluno entre em interação com os objetos desse ambiente, possa desenvolver outros conhecimentos, por exemplo, conceitos geométricos ou matemáticos.

Propicia ao aluno a possibilidade de aprender fazendo, ou seja, ensinando a tartaruga a resolver um problema, seguindo a linguagem de programação. O aluno pode ao ver o resultado da execução, comparar suas expectativas originais com o produto obtido, analisando suas idéias e os conceitos que usou.

Se houver um erro o aluno pode depurar o programa e identificar a origem do erro, usando o erro de modo produtivo, para entender melhor suas ações.

O aprendizado proporcionado pelo ambiente *LOGO,* (hoje não é mais a única opção) também pode ser obtido através de outros softwares como os sistemas de autoria e softwares abertos como planilhas, banco de dados e simulações, dependendo de como o professor irá utilizá-los. A figura 4 mostra uma das telas desta aplicação.



Fonte: Wendt (2002, p. 11)

Figura 4 – Tela software Megalogo

2.3.5 JOGOS EDUCATIVOS

Segundo Wendt (2002, p. 12), os softwares educacionais tipo jogos educativos utilizam recursos multimídia e tornam o aprendizado muito mais interessante e divertido. Estes jogos são elaborados para divertir os alunos enquanto eles aprendem.

Conforme Ferreira (1995), um dos primeiros a conseguir popularidade foi à série *Carmem Sandiego* (desenvolvido pela Broderbund), que pretendia ensinar conceitos de história e geografia a partir de uma "caçada" a uma ladra internacional e sua quadrilha ao

redor do mundo. Nele as pistas eram resolvidas com aplicação de conhecimentos nessa área, despertando o interesse de crianças por essas matérias. Apesar de ser voltado para crianças, o jogo obteve grande popularidade entre os adultos e adolescentes. Exemplos de telas do jogo *Carmem Sandiego* podem ser vistos na figura 5.

Fonte: Wendt (2002, p. 13)

Figura 5 - Tela jogo Carmem Sandiego

Desta forma os jogos possuem um importante papel no desenvolvimento infantil, pois atuam como motivadores no processo de ensino-aprendizagem e mantêm uma estreita relação com a construção do conhecimento das pessoas. O brincar, através do jogo, é uma atividade natural do ser humano. Entretanto, é importante pensar no jogo como um meio educacional, deixando de lado a idéia do jogo pelo jogo, e observando-o como um instrumento de trabalho e um meio para atingir objetivos preestabelecidos, tal como o desenvolvimento de conteúdos curriculares.

Os jogos por computador, segundo Battaiola (2000, p. 84), são produtos muito populares, especialmente entre os jovens. O grau de evolução tecnológica deste tipo de *softwar*e, nos últimos anos, tem sido expressivo.

O jogo é um vínculo que une a vontade e o prazer durante a realização de uma atividade e possibilita que o ensino, utilizando meios lúdicos, crie ambientes gratificantes e atraentes, servindo de estímulo para o desenvolvimento integral do aluno.

Segundo Battaiola (2000, p. 87), embora os jogos de maior sucesso comercial atualmente sejam os que melhor combinam violência com efeitos visuais sofisticados, um ponto positivo nesta área é a possibilidade de combinar entretenimento com educação. Estima-se que menos de 20% dos jogos disponíveis tem algum enfoque educacional. Em

geral, os jogos mais utilizados só auxiliam o desenvolvimento da rapidez de raciocínio e reflexos, abrindo um grande campo para pesquisas nesta área.

Um jogo por computador pode ser definido como um sistema composto de três partes, conforme ressalta Battaiola (2000, p. 100):

a) enredo: define o tema, a trama, objetivo(s) do jogo e qual será a série de passos que o usuário deverá realizar para atingir o objetivo estabelecido. Desta forma, a definição do enredo não envolve apenas a criatividade e a pesquisa sobre o tema, mas também a interação com especialistas no assunto a ser enfocado pelo jogo;
b) motor do jogo: pode ser definido como o mecanismo que controla a reação do jogo em função de uma ação do usuário. A implementação do motor envolve diversos aspectos computacionais, tais como a escolha da linguagem de programação, o desenvolvimento de algoritmos específicos, o tipo de interface com o usuário entre outros;
c) interface interativa: trata-se da parte que controla a comunicação entre o motor do jogo e o usuário, reportando graficamente um novo estado do jogo. Contudo a interatividade, proporcionada através da interface e que requer a movimentação de personagens e objetos, apresenta-se como um dos principais atrativos dos jogos por computador. O desenvolvimento da interface envolve aspectos artísticos, cognitivos e técnicos, onde o valor artístico de uma interface está na capacidade que ela tem de valorizar a apresentação do jogo, atraindo usuários e aumentando a sua satisfação ao jogar.

O desenvolvimento de jogos por computador geralmente envolve uma série de importantes etapas que devem ser planejadas e executadas buscando as definições de cada uma das três partes caracterizadas anteriormente, de modo a possibilitar o desenvolvimento de um jogo que atenda a todos os objetivos esperados.

3 ERGONOMIA E INTERFACE

3.1 ERGONOMIA

Na informática, a ergonomia tem contribuído de varias maneiras para obtenção de melhores resultados, desde a concepção ou correção dos postos de trabalho, atuando nas relações físicas entre o operador e a máquina até a concepção ergonômica de interface. Portanto os conceitos aqui apresentados objetivam mostrar evolução da ergonomia nos diversos campos de estudos e áreas de atuação.

3.1.1 CONCEITOS DE ERGONOMIA

Os conceitos tradicionais abordam claramente a preocupação com a ergonomia clássica cujo interesse fundamental é principalmente nos processos físicos do homem.

> "Ergonomia é o estudo do trabalho ao homem, onde o objetivo central é o ser humano, suas habilidades, capacidades e limitações de posse, no qual destes conhecimentos pode-se dizer quais as ferramentas e materiais, os métodos de trabalho, o arranjo dos instrumentos e do local de trabalham que se melhor adaptam." (Rigui (1993 *apud* Stein 1994, p. 51).

Segundo Laville (1977, p. 6), conceitua-se "ergonomia como estudo do comportamento do homem em relação com seu ambiente especial. A pesquisa ergonômica é usada na adaptação das condições de trabalho à natureza física e psicológica do homem, e isto resulta no mais importante princípio de ergonomia, que se torna adaptar a tarefa ao homem."

Para certos especialistas a ergonomia é a ciência do trabalho, ou o conjunto das ciências do trabalho (fisiologia, psicologia, sociologia do trabalho). Para outros, a ergonomia é uma tecnologia cujo objeto é o arranjo dos sistemas homem-máquina, ou mais amplamente das condições de trabalho, em função de critérios dos quais as mais importantes caracterizam o bem-estar dos trabalhadores (saúde, segurança, satisfação, conforto, etc.) Dentro da primeira concepção, faz sentido falar de ergonomia aplicada, mas não no segundo caso, onde a ergonomia é por natureza aplicada.

Conforme Moraes (1989), o estudo científico da relação entre o homem e seu ambiente de trabalho, pode neste sentido apresentar que o termo ambiente não se refere apenas ao entorno ambiental, no qual o homem trabalha, mas também às suas ferramentas, seus métodos de trabalho e à organização deste, considerando-se este homem, tanto como um indivíduo quanto como participante de um grupo de trabalho. Finalmente, tudo isto se

relaciona com a natureza do próprio homem, com suas habilidades, capacidades e limitações. Na periferia da ergonomia, sem, no entanto, até o presente momento, fazer parte do seu campo, estão as relações do homem com seus companheiros de trabalho, seus supervisores, gerente e sua família. Estes pontos são usualmente considerados como parte das ciências sociais, mas eles não devem ser ignorados, já que podem desempenhar um importante papel na solução de alguns problemas de ergonomia.

Os conceitos de ergonomia da informática abordam a preocupação do homem como um todo, onde, além dos aspectos físicos são relevantes os aspectos cognitivos. Assim, a ergonomia evidencia que é necessário conhecer o processo cognitivo do pensamento humano para que se construa sistemas amigáveis. Talvez a melhor solução seja o trabalho conjunto entre usuários e analistas, podendo assim, melhorar a maneira de executar a tarefa.

> "As principais qualidades ergonômicas recomendadas para um sistema são relativamente conhecidas; ele deve ser simples para aprender e para utilizar, fácil de memorizar, rápidos, confiáveis, positivos e auxiliar o usuário a resolver suas dificuldades." (Rigui (1993 *apud* Stein 1994, p. 53).

Segundo Odebrecht (1993), a ergonomia não é revolucionária, ela caracteriza-se como reformista. Reformar não significa acabar com o antigo, com o existente, mas dar nova forma ou reconstruir para melhorar nos diversos aspectos o que existe, e para isso propõe uma série de recomendações.

3.1.2 DIFICULDADES NA UTILIZAÇÃO DOS CONHECIMENTOS ERGONÔMICOS

Cybis (1994) afirma que, apesar dos ergônomos estarem cientes da importância das contribuições que a ergonomia pode trazer para o aumento da produtividade e melhoria das condições de trabalho do usuário e de seus sistemas, os programadores e analistas, permanecem distantes desta disciplina. Isto, por não terem acesso nem aos conhecimentos necessários nem as ferramentas adequadas.

A dificuldade para a aplicação da ergonomia ocorre porque as contribuições não são rapidamente assimiláveis, pois fazem referencia a uma vasta área de conhecimento, e a uma metodologia especifica. Isso tem exigido, a participação de programadores e de analistas em cursos de treinamento em ergonomia. Da mesma maneira, que os ergonomistas estão tendo que se especializar nas questões de informática, conforme ressalta Cybis (1994).

Conforme o mesmo autor muitos programadores e analistas consideram que o estudo ergonômico atrasa o ritmo de desenvolvimento de seus sistemas, isso porque ocorrem

modificações nos planos e metodologias já estabelecidas, e a necessidade de um trabalho de equipe com freqüentes reuniões de coordenação e validação da tarefa.

3.1.3 PRINCÍPIOS ERGONÔMICOS PARA GERAÇÃO DE SOFTWARE

Diferentemente dos métodos tradicionalistas que moldam a realização da tarefa às características e potencialidades dos equipamentos e que apenas consideram superficialmente as características dos recursos humanos envolvidos na implementação do sistema durante a fase de projeto lógico, a abordagem ergonômica para geração de software privilegiada o conhecimento do usuário e o conhecimento da tarefa ou do trabalho como elementos básicos.

Por essa abordagem o ergonomista pode ser considerado como mediador entre o criador de software e o usurário, pois conhece os métodos que permitem reunir as informações e exigências do usuário pertinentes à tarefa e tem experiência sobre as diversas maneiras pelas quais essas exigências podem ser traduzidas na concepção do software.

Conforme Odebrecht (1993), alguns pontos devem ser considerados em uma abordagem ergonômica:

a) uma adequada análise de trabalho anterior à informatização, seguindo métodos e técnicas da ergonomia, o que pode relevar os modelos mentais montados pelos profissionais, bem como identificar o fluxo das informações tratadas;
b) concepção de nova estrutura do trabalho informatizado, procurando adequar-se ao fluxo de informações e aos modelos mentais dos operadores;
c) concepção da interface segundo nova estrutura do trabalho com critérios quanto à apresentação da informatização, uso de cores, ícones, janelas, etc.;
d) prototipagem e ensaios com interfaces antes de seu uso comercial com amostras do usuário da população alvo, procurando identificar as telas mais problemáticas para as devidas revisões de projeto;
e) análise final da interface prevendo sua implantação no trabalho informatizado.

Segundo Cybis (1994), a ergonomia tem como principal objetivo à adequação das exigências da tarefa ao homem. Em sistemas informatizados é necessário que sejam consideradas as habilidades e capacidades perceptivas e cognitivas humanas, assim como, os aspectos ligados à tarefa a ser desenvolvida. Levando-se em consideração estas características é possível conceber sistemas adaptados aos usuários e suas tarefas, proporcionando desta forma um rápido aprendizado, facilidades de uso e baixas taxas de erro. Neste sentido, a

Ergonomia será empregada em três momentos distintos: na concepção do sistema, na escolha das interfaces e nas condições de utilização do sistema.

Conforme ressalta Cybis (1994), é função da ergonomia cognitiva analisar as características internas do software, enquanto a ergonomia de interfaces preocupa-se com a forma de apresentação das informações ao usuário, facilitando a utilização do sistema e garantindo que as habilidades, capacidades e necessidades do mesmo, sejam levadas em consideração no projeto de cada parte do sistema.

Um dos paradoxos da ergonomia de concepção é estudar uma determinada atividade para uma situação de trabalho que ainda não foi implementada. Nos sistemas de ajuda de forma geral, e mais especificamente na aprendizagem, a principal limitação encontrada é de natureza sensorial.

Para as máquinas, ainda não é possível captar dos usuários os sinais de dúvida, hesitação ou compreensão, que seriam as informações retidas na memória sensorial para posterior tratamento na memória de curto termo. Cabe à memória de curto termo ou de trabalho, guardar as informações sobre um diálogo ou parte de um problema. Para o desenvolvimento de sistemas cognitivamente ergonômicos é importante lembrar que o raciocínio baseia-se nos modelos mentais referentes à imagem que as pessoas tem de mundo, o que as leva a fazerem pré-conceitos, influenciando na percepção e na forma de conduzir o raciocínio.

Assim, a interface com usuário tem a função de medir o diálogo entre dois elementos: o operador de um sistema interativo (um ser humano) e o conjunto hardware/software utilizado para implementar o sistema interativo. Cada elemento impõe requisitos ao produto final. O operador é o juiz da usabilidade e adequação da interface, o hardware e o software são as ferramentas com as quais a interface é construída.

Por conseqüência, uma interface útil e adequada ao operador deve ser construída com ferramentas de hardware e software disponíveis. A complexidade destes componentes determina a tomada de muitas decisões sobre como empregar as ferramentas disponíveis para melhor satisfazer o usuário.

3.1.4 AVALIAÇÃO ERGONÔMICA DA INTERFACE NA INFORMÁTICA

Uma interface homem-computador deve ser flexível o suficiente para adequar-se a diferentes tipos de usuários, ao mesmo tempo em que possa adaptar-se a evolução das características de um usuário específico durante o seu processo de aprendizagem com o sistema.

A escolha de interfaces é um requisito muito importante e deve permitir que o usuário troque informações com o computador. Os aspectos de hardware de uma interface são os dispositivos de entrada de dados (teclado, mesa digitadora, mouse, microfone), e também, de saída de dados (vídeo, caixa acústica). Com relação ao software, a ergonomia se preocupa com o diálogo com o usuário e com arquitetura de telas. É necessário também, que se leve em consideração à maneira como o usuário trata as informações e a sua capacidade de tratamento das informações simbólicas.

Considerando os aspectos externos do software, características relevantes devem ser lembradas no desenvolvimento da interface, tais como:

a) homogeneidade de interfaces, de forma que o diálogo com o usuário não apresente variações;
b) seja levada em consideração à lógica de utilização do usuário, a partir da tarefa a ser realizada;
c) tratamento de erros com relação à utilização do sistema, enviando mensagens que possam tirar pequenas dúvidas, auxiliando a encontrar a resposta correta.

Com relação à parte interna do software, os conceitos mais relevantes da ergonomia cognitiva a serem considerados no desenvolvimento de uma interface também são destacadas no estudo de Cybis (1994)

a) flexibilidade de utilização, permitindo que vários usuários o utilizem corretamente, desde os mais iniciantes aos mais especialistas;
b) simplicidade nas respostas fornecidas ao usuário, melhorando a eficiência do sistema;
c) distinção entre as tarefas prescritas e as realizadas pelo usuário, respeitando as características cognitivas individuais;
d) utilização das características da memória de curto termo, permitindo o encadeamento das ações;

e) desenvolvimento de automatismo para a utilização do sistema, permitindo ao usuário utilizar seus processos conscientes nas tarefas mais complexas.

A Ergonomia procura garantir produtos e sistemas adaptados às habilidades de quem os utiliza e apropriados às tarefas que as pessoas desempenham. Há um compromisso com o desempenho e a eficiência de um lado, e com a saúde e a satisfação dos usuários de outro. Para garantir este objetivo, torna-se necessária à análise da tarefa e da atividade como forma de evidenciar a lógica de utilização do sistema baseada na representação do ambiente de trabalho e dos modelos mentais utilizados pelo usuário.

A avaliação dos efeitos da interface sobre os usuários é um trabalho de elaboração relativamente rápido e pode ser realizado mesmo sem a participação destes agentes. O conhecimento ergonômico é utilizado em suas diversas formas (modelos cognitivos, critérios e recomendações ergonômicas) através de técnicas analíticas (avaliação heurística, exploração cognitiva e checklist) que permitem identificar de antemão (à priori) os problemas de facilidade de uso e intuitividade do software. Essa técnica de avaliação tem uma característica intrinsecamente qualitativa e subjetiva, pois dependem da experiência e competência de analistas em ergonomia de interfaces que verificam um conjunto de qualidades.

Os objetivos da avaliação ergonômica de um dispositivo de software interativo são: avaliar as funcionalidades, levando em conta as necessidades do usuário, e também, de avaliar o efeito da interface sobre os usuários, que se traduz na facilidade de aprendizagem do software (intuitividade) e na facilidade e eficiência de uso.

Princípios ergonômicos foram desenvolvidos a partir do exame exaustivo de uma base de recomendações ergonômicas e que são empregados nas intervenções de ergonomia, destacadas estudo de Bastien & Scapin (1993, *apud* Cybis 2003, p. 31):

a) *condução*: define-se no convite (presteza) do sistema, na legibilidade das informações e telas, no feedback imediato das ações do usuário e no agrupamento e distinção entre itens nas telas;
b) *carga de trabalho*: define-se na brevidade das apresentações e entradas (concisão), no comprimento dos diálogos (ações mínimas) e na densidade informacional das telas como um todo;
c) *controle explícito*: define-se no caráter explícito das ações do usuário (ações explícitas) e no controle que ele tem sobre os processamentos (controle do usuário);

d) *adaptabilidade*: refere-se tanto às possibilidades de personalização do sistema, que são oferecidas ao usuário (flexibilidade), como ao fato da estrutura do sistema estar adaptada a usuários de diferentes níveis de experiência (consideração da experiência do usuário);
e) *gestão de erros*: refere-se tanto aos dispositivos de prevenção, que possam ser definidos nas interfaces (proteção contra erros), como à qualidade das mensagens de erro fornecidas, e às condições oferecidas para que o usuário recupere a normalidade do sistema da tarefa (correção de erros);
f) *consistência*: refere-se à homogeneidade e à coerência das decisões de projeto quanto às apresentações de diálogos;
g) *significado dos códigos e denominações*: refere-se à relação conteúdo expressão das unidades de significado das interfaces;
h) *compatibilidade*: se define no acordo que possa existir entre as características do sistema e as expectativas, anseios e características dos usuários e de suas tarefas.

Através destes critérios, Bastien & Scapin (1993, *apud* Cybis 2003, p. 40) buscam descrever qualidades ergonômicas não quantificáveis. A atribuição de pesos aos critérios para computar notas de analistas e chegar a resultados absolutos representa o processamento de subjetivismo, que resulta em algo subjetivo. Por outro lado, para se obter uma avaliação sobre funcionalidades do dispositivo, a participação de seus usuários é fundamental. As técnicas possíveis envolvem a aplicação de questionários, realização de entrevistas e a montagem de uma simulação de uso do software. São ensaios de interação, que envolvem a participação de usuários especialistas na tarefa e especialistas no uso do software.

Segundo Cybis (2003, p. 40), os ensaios de interação envolvem um trabalho demorado e de difícil elaboração, como a montagem de cenários para as simulações e a obtenção de uma amostra de usuários-alvo do sistema, disposta a se engajar em seu projeto ou avaliação. Apesar de sua arquitetura complicada, os ensaios permitem a coleta de informações qualitativas e quantitativas importantes (tempo de execução de tarefas, quantidade de erros, os comandos utilizados, etc.). Assim, os resultados de uma avaliação qualitativa são empregados para direcionar a composição de cenários segundo as funcionalidades com interfaces problemáticas.

A realização de ensaios de interação permite trabalhar as impressões dos analistas sobre a potencialidade de problemas ergonômicos na busca de fatores quantitativos como taxa

de eficácia e eficiência no uso do ambiente (se os objetivos propostos foram alcançados e com que nível de recursos), e qualitativos, como a satisfação de seus usuários. Estas são exatamente as métricas definidas pela ISO 9241 em sua parte 11 (1993), para a avaliação ergonômica de softwares de escritório. Uma técnica alternativa de captura de dados quantitativos se vale de dispositivos informatizados que permanecem residentes nos computadores de alguns usuários, e vão com o tempo registrando todos os detalhes das interações. Deste modo, com o objetivo de diminuir o efeito da subjetividade dos analistas, o nível de utilizabilidade final resulta de uma sistemática de avaliação "mista" baseada na eficácia, eficiência (fatores quantitativos) e satisfação (fator qualitativo) de uma amostra bem definida de usuários desses sistemas.

3.1.5 ALGUMAS DEFICIÊNCIAS APONTADAS PELA ERGONOMIA

Segundo Cybis (1994), as deficiências de concepção podem extrapolar a capacidade de adaptação do usuário, conduzindo o problema no uso dos sistemas que vão deste a sua má utilização até a modificação da tarefa, causando frustrações, desinteresse e altas taxas de erros.

A ergonomia metodologicamente pode fazer frente a um outro grupo de deficiências no desenvolvimento dos sistemas, tais como:

a) concepção segundo a orientação funcional em detrimento daquela operacional;
b) concepção segundo o critério de desempenho do sistema ao invés de objetivos do usuário;
c) fornecer todas as informações e funções disponíveis e imagináveis ao invés de somente as essenciais e necessárias;
d) situação de não prever erros humanos.

3.2 INTERFACE

A interface homem-máquina é de extrema importância, pois condiciona o desempenho do operador e, na medida em que se estabelece como e quais informações estarão a ele disponíveis, influencia fortemente na realização da tarefa. A interface é um trabalho mútuo, onde a máquina e o homem devem estar em sinfonia. A construção desta interface é um desafio porque o ser humano possui particularidades que podem gerar uma grande variabilidade no sistema de produção automatizado.

Segundo Stein (1994, p. 18), o uso de interfaces interativas visa auxiliar os usuários não especialistas, via recursos de ajuda e outras características essenciais. Mas, por outro lado, o especialista não pode ser esquecido, pois é um critico em potencial e se negará a usar sistemas que tornem seu trabalho desagradável. Por isso, o desenvolvimento de interfaces se tornou um desafio para os engenheiros de software e projetistas de sistemas.

A interface com o usuário é a parte mais importante e crítica de um programa aplicativo. É a única parte que os usuários finais vêem, pois não podem percorrer o interior do pacote para admirarem a eficiência do projeto interno, tudo o que vêem é o que está na sua frente, na tela. Se o que se projeta é algo desajustado, lento, insensível ou difícil de aprendizagem, o usuário poderá classificar o aplicativo como sendo de má qualidade.

Para que os objetivos previstos sejam alcançados, deve haver uma maior participação do usuário na construção de sistemas.

3.2.1 CONCEITOS DE INTERFACE

Com o advento de novas tecnologias, em especial com a automatização de vários processos de trabalho, torna-se importante a observação de vários processos de trabalho, do relacionamento do homem com a máquina (já que o desempenho de ambos está interligado) e a saúde física e mental do operador depende muito desta interação e de como ela é desenvolvida ao longo da rotina de trabalho.

As interfaces homem-máquina interativas possuem diversos aspectos técnicos que devem ser considerados no momento do projeto. Há que se adaptar as máquinas às condições físicas, cognitivas e psíquicas do homem.

Conforme ressalta Cybis (2003, p. 05), uma interface homem-máquina engloba os aspectos dos sistemas informatizados que influenciam a participação do usuário em suas tarefas. Nela estão incluídos os dispositivos de hardware, os programas de entrada e saídas, a arquitetura do diálogo, os manuais, os cursos de treinamento e o suporte.

3.2.2 PROBLEMAS NA INTERAÇÃO HOMEM-COMPUTADOR

Cada vez menos, o usuário final precisa saber sobre linguagem de máquina, funcionamento de computadores, linguagens de programação etc. A tendência é a máquina adaptar-se a seu usuário (seja este o programador ou o cliente). Portanto, é essencial o

desenvolvimento de interfaces inteligentes, independentes do funcionamento dos computadores.

A comunicação homem-máquina é o objetivo da área de ergonomia da informática, a qual têm estimulado (através de recomendações) o desenvolvimento de interfaces amigáveis para os sistemas tradicionais de processamento de dados.

Conforme salienta Cybis (1994), os problemas na interação homem-computador podem ser identificados através dos seguintes aspectos:

a) falta de conhecimento preliminar das tarefas e dos usuários por parte dos projetistas;
b) projetista do sistema, por ser especialista em programação, tem pouco ou nenhum conhecimento da área em que o aplicativo vai funcionar;
c) utilização de jargões profissionais e termos estrangeiros ou desconhecidos na área do aplicativo;
d) falta de proteção às funções que acarretam grandes modificações no estado do sistema;
e) ausência de resposta do sistema para algumas ações dos usuários;
f) falta de guia e orientação;
g) conhecimento incompleto sobre a tarefa e os usuários do trabalho a ser informatizado;
h) ausência de métodos e ferramentas lógicas especificas para a concepção e avaliação de interfaces com o usuário;
i) considerar o computador como um fim em si mesmo.

A dificuldade no desenvolvimento de interfaces deve-se ao fato de que elas constituem, fundamentalmente, sistemas abertos, sujeitos às influências do ambiente e às interpretações dos usuários. Pode-se dizer, desta forma, que suas entradas e saídas podem significar coisas diferentes para pessoas diferentes, em função de seu conhecimento, do momento, do ambiente que as cercam.

Se, por um lado, os programas das aplicações são construídos por meio de linguagens de programação, uma interface humano-computador é construída por meio de um conjunto aberto de vários símbolos, que devem e podem ser interpretados de diferentes formas pelos usuários, em função de seu contexto dinâmico. Assim, pode-se afirmar que a experiência da interação humano-computador é individual e única, no sentido de que cada pessoa é única em sua bagagem de conhecimento e experiência. Dificilmente uma mesma interface vai ter o

mesmo significado para usuários distintos. Menor ainda é a chance dela ter um significado compartilhado entre usuários e os projetistas da interface.

4 ENGENHARIA USABILIDADE

Este capítulo tem o intuito de apresentar conceitos relacionados ao processo de desenvolvimento de interfaces ergonômicas.

Inicialmente, serão tratadas as bases teóricas advindas da psicologia cognitiva e semiótica, seguindo para uso de ferramentas e ciclo da Engenharia de Usabilidade.

4.1 CONCEITOS

Na década de 90, foram desenvolvidas diversas metodologias com o objetivo de fornecer apoio à construção de interfaces intuitivas e produtivas. A Engenharia da Usabilidade começava a ser implementada como função nas empresas desenvolvedoras de software interativo.

Assim segundo Cybis (2003, p. 01), a engenharia de usabilidade, tem como definição a utilização de princípios de engenharia de forma a obter produtos que sejam fáceis de utilizar, economicamente viáveis e que suportem trabalhar de forma real e eficaz.

Já o conceito de usabilidade segundo a *Iternational Standards Organization* (*apud* Cybis, 2003, p.02), pode ser definida como efetividade, eficiência e satisfação com que determinados usuários conseguem atingir objetivos específicos em circunstâncias particulares.

Conforme apresentados por vários autores a engenharia de usabilidade, tem como papel fundamental participar no desenvolvimento de projetos que sejam de fatores humanos e interativos com os usuários.

Desta forma Cybis (2003, p. 04) apresentou duas formas que podem ser decompostos os sistemas básicos da engenharia de usabilidade, conforme abaixo:

a) núcleo funcional: formado por programas, aplicativos, algoritmos e principalmente por bases de dados;
b) interface com o usuário: formada por apresentações, informações de dados, controles e comandos. É esta interface responsável por recepcionar as entradas, controles e comandos de dados. Finalmente, ela tem como função controlar o diálogo entre as apresentações e as entradas. Uma interface tanto pode definir as estratégias para a

realização da tarefa, como conduz, orienta, recepciona, ajuda e responde ao usuário durante as interações.

Assim foi a partir do final dos anos 80 e sobretudo nos anos 90, que foram desenvolvidos as primeiras abordagens, métodos, técnicas e ferramentas destinadas a apoiar a construção de interfaces intuitivas, fáceis de usar e produtivas. Desta forma a engenharia de usabilidade, saía dos laboratórios das universidades e institutos de pesquisa e começava a ser implementada, como função nas empresas desenvolvedoras de softwares interativos.

4.2 TEORIAS COGNITIVAS

A apresentação correta da informação na geração de softwares interativos passa pelo entendimento dos princípios da percepção e cognição, pelo conhecimento da natureza dos sinais que compõem as telas e pelas suas características físicas.

Segundo Cybis (2003, p.13), a questão principal segundo teorias cognitivas, é tornar as representações propostas para uma interface compatíveis com aquelas desenvolvidas pelo homem em seu trabalho.

A imagem operativa cognitiva varia de pessoa para pessoa em função do seu conhecimento em relação ao sistema. Portanto esta imagem é a representação que o usuário tem da realidade do trabalho, de uma forma simplificada pelo que é funcionalmente significativo. Logo uma interface deve se adequar aos diferentes tipos de usuários ao mesmo tempo em que se adapta a um usuário especifico durante o processo de aprendizagem.

Considerar o usuário a uma interface significa conhecer as informações provenientes da analise ergonômica do trabalho (idade, formação, conhecimentos, etc.), e também das informações ligadas às habilidades e capacidades humanas em termos cognitivos. Desta forma na medida que se pretende que o computador trabalhe como uma extensão do cérebro humano, é fundamental que ele conheça o papel que estas habilidades desempenham.

Para Barthet (1998, *apud* Stein 1994, p. 64), a resolução de problemas entre homem-computador coloca em funcionamento os diferentes níveis de memória e modos de raciocínio, que estão divididos em:

a) memória sensorial: aquela que recebe informações de diferentes órgãos sensitivos e a mantém por períodos de tempo muito curto, sem interpretá-las;

b) memória de curto termo: tem uma pequena capacidade de armazenamento, mas desempenha papel fundamental em atividades mentais de conversações ou de raciocínio. A informação da memória de curto termo é mantida por freqüentes "rememorizadores" que, com o tempo, podem transferi-la para a memória de longo termo;
c) memória de longo termo: está organizada em estruturas que armazenam o conhecimento adquirido no passado sobre um tipo de objeto ou evento.

Desta forma a ergonomia cognitiva mostra que é necessário conhecer o processo do pensamento humano para se fazer sistemas com interface adequada.

4.3 TEORIAS SEMIÓTICA

Os sistemas de sinais começaram a ser estudado no início do século XX, conforme Andersen (*apud* Oliveira,1995), define que semiótica computacional como "a ciência que estuda os sistemas de sinais na forma de linguagens, códigos e sinalização". Desta forma significa dizer que o computador é visto como uma máquina simbólica que realiza tratamentos de sinais produzidos pelos programadores para produzir os sinais que os usuários interpretam e manipulam em suas interfaces.

Segundo Cybis (2003), a semiótica é uma relação entre formas de expressão e de conteúdo que só ocorre quando ele é interpretado. Assim, o sistema informatizado é visto como um sistema de expressões "vazias", pois dependem do usuário para se realizarem como sinais. Os projetistas podem influenciar fortemente estas interpretações ao conceberem seus candidatos a sinais computacionais. Desta forma, sua atividade possui o caráter de criação de proposição de significados. Não se pode dizer que um projetista conceba sinais, ele propõe sinais, que em algumas circunstâncias se realizam, mas que em muitas outras nunca atingem a realização prevista.

Conforme ressalta Andersen (*apud* Oliveira,1995), programar, no sentido semiótica do termo é usar o computador para tentar dizer algo às pessoas. Deste modo, os sinais computacionais são definidos como sinais candidatos. Eles dependem do usuário para se realizarem como sinais. Entretanto o projetista, tem no seu papel, poder influenciar sua interpretação.

Desta forma, o computador é visto essencialmente, como um meio para a comunicação. Desta forma em um sistema informatizado é o projetista quem define os limites da comunicação criando os sinais que o usuário poderá manipular. Assim para o computador não possuiu todas as informações de um emissor ou de um receptor, ao contrário de pessoas, que articulam uma linguagem mesmo sem conhecer seu "programa" ou gramática. As pessoas, ao contrário de um computador, possuem a capacidade de modificar uma linguagem naturalmente, pois as linguagens humanas não foram construídas por um grupo de projetistas, mas evoluíram naturalmente com o uso.

4.4 FERRAMENTAS PARA ENGENHARIA DA USABILIDADE

4.4.1 QUALIDADES ERGONÔMICAS PARA IHC

O sucesso de qualquer atividade de concepção ou de avaliação depende do emprego de critérios bem definidos. Em pesquisa com vários autores observou-se que minimizar a ambigüidade na identificação e classificação das qualidades e problemas ergonômicos do software interativo, permite avaliar os componentes de uma IHC que futuramente irão interagir, conforme apresentado no capítulo 3 em avaliação ergonômica da interface na informática.

Conforme Cybis (2003, p.39) o uso desses critérios permitem avaliar os componentes de IHC que irão interagir. Sob outro ponto de vista, a IHC é vista como uma linguagem (sistema de sinais) cuja estrutura lexical e sintática é conhecida pelo usuário e pelo sistema informatizado.

4.4.2 COMPONENTES DE IHC

Conforme ressalta Cybis (2003, p. 39), a IHC é entendida como um subsistema do software interativo que como tal, possui estruturas e processos. A estrutura é representada por seus componentes, e os processos se estabelecem na interação entre estes componentes e usuários do sistema.

A analogia entre interfaces com o usuário e um sistema de linguagem é detalhada no modelo de camadas de abstração, conforme proposto por Nielsen (1984 *apud* Cybis, 2003, p. 39):

a) nível de objetivos: refere-se aos objetivos dos usuários independentemente do sistema informatizado;
b) nível pragmático: referem-se as funções e estruturas de dados, associados aos conceitos do mundo real, como implementados no sistema;
c) nível semântico: refere-se aos significados que o usuário desenvolve sobre a operação das funções e estruturas de dados do sistema em associação com o mundo real;
d) nível sintáxico: refere-se tanto aos diálogos como as telas, janelas e caixas de diálogo individuais. Este diz respeito às relações entre os objetos de interação apresentados numa seqüência de telas e de modo concorrente, em uma única tela;
e) nível lexical: refere-se aos nomes dos comandos e desenhos de ícones. Trata dos significados das unidades veiculando os itens de informações;
f) nível de primitivas: referem-se as fontes, linhas, texturas, cores e sons, que representam o conjunto de unidades construtivas dos itens de informação;
g) nível físico: trata dos dispositivos de entrada e saída do sistema.

Esses modelos lingüísticos proporcionam a diretriz mais utilizada para a organização dos elementos das interfaces humano-computador e do raciocínio ergonômico para sua seleção, configuração e avaliação.

O modelo abordado propõe classes de elementos organizados a partir de diálogos (sintaxe seqüência), objetos de interação (sintaxe concorrente), sistemas de significados (léxico) e primitivas.

Os diálogos são vistos como seqüências de interações entre o homem e o sistema podem ser analisados segundo perspectivas de função, forma e estrutura. As funções dos diálogos definem as classes de tarefas, e representa o nível pragmático das interações homem-sistema. Elas estão associadas às maneiras de apoiar os objetivos práticos dos usuários nas interações com o sistema. O modelo de características de interfaces humano-computador propõe alguns tipos de tarefas genéricas definidas nas relações com diversos tipos de programas aplicativos. O componente elementar de classes de tarefa é uma ação.

Os estilos de diálogo representam as sintaxes seqüenciais do modelo, que propõe as classes de; preenchimento de formulários, diálogo por menu, diálogos de manipulação direta e diálogos de questão *versus* resposta.

As estruturas dos diálogos determinam as dinâmicas possíveis de um diálogo. Entre os exemplares possíveis dessas classes constam às estruturas do diálogo, estrutura do menu, estrutura da linguagem de comando, estrutura das teclas de funções.

Assim Cybis (2003, p.41), apresenta que as classes de objetos de interação representam as relações estáticas que se estabelecem nas telas, janelas, caixas de diálogo, etc. Elas foram agrupadas segundo uma perspectiva funcional-estrutural, definindo as classes de painéis de controles, controles complexos, grupos de controles, controles simples, campos de entrada, dados complexos, dados simples e as informações. Os sistemas de significados definem as partes léxicas da interface, sendo ainda possível definir as primitivas gráficas empregadas na construção das apresentações dos objetos. O modelo de características das interfaces humano-computador poderia ainda abrigar as classes de mídias sobre os aspectos físicos dos dispositivos de entrada e saída.

O quadro 1 apresenta o modelo de componentes das interfaces humano-computador apresentado por Cybis (2003):

Organização das Componentes		Classes de Componentes
Ações	Ações de Entrada	Entrada de dados e comandos
Interações	Tarefas	Tarefa de Diagnóstico, Tarefa Corretiva Tarefa Destrutiva
	Estilos	Diálogo por Menu Diálogo por Preenchimento de Formulário Diálogo por Linguagem de Comando Diálogo por Manipulação Direta
	Estruturas	Estrutura Seqüencial (Passo à passo) Estrutura Paralela Estrutura Repetitiva
Objetos de Interação	Painéis de Controles	Tela, Janela, Caixa de Diálogo, Caixa de Ação/Tarefa, Tela de Consulta, Formulário, Caixa de Mensagem
	Controles Compostos	Barra de Menu, Painel de Menu, Página de Menu, Barra de Ferramentas, Lista de Seleção, Lista de Combinação
	Grupos de Controles	Grupo de Botões de Comando, Grupo de Botões de Rádio, Grupo de Caixas de Atribuição Grupo de campos/mostradores de dados
	Controles Simples	Botão de Comando, Caixa de Atribuição, Cursor do Dispositivo de Apontamento, Escala, Dial
	Campos de Entrada	Campo de Texto, Campo de Dado, Campo Gráfico, Linha de Comando
	Mostradores Estruturados	Lista de Dados, Tabela de Dados, Texto, Gráfico, Diagrama de figura, Diagrama de Texto, Mapa
	Mostrador simples	Mostrador de Dados
	Mostrador de Informações	Rótulo, Mensagem de Orientação, de Ajuda, de Alerta, Aviso, Mensagem de Erro, Indicador de Progressão, Efeito Sonoro, Motivo Melódico, Locução, Fala
Sistemas de Significado	Motivados Arbitrários	Denominação, Abreviatura, Ícone Código Alfanumérico, Código de Cores, Código de Textura, Código de Intermitência, Código de Vídeo-Reverso
Primitivas	Visuais Sonoras	Cor, Fonte, Linha, Arranjo Som

Fonte: Cybis (2003, p. 42)

Quadro 1 – Modelo componente de IHC

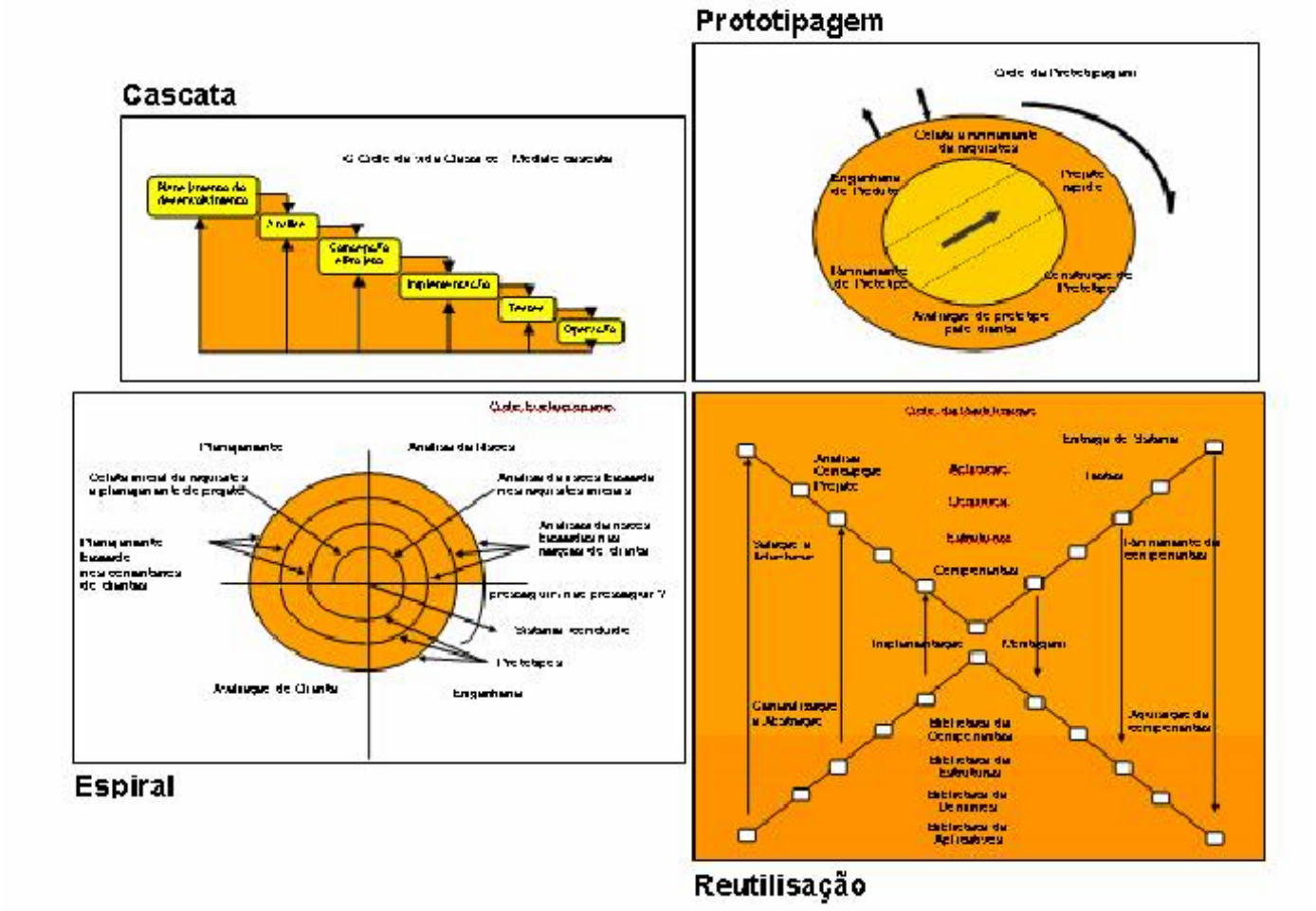
4.5 CICLO DA ENGENHARIA DE USABILIDADE

O desenvolvimento de softwares ainda inclui métodos nos quais o programador insere todas as funções e informações imagináveis, deixando para o usuário a simples tarefa de alcançar seus objetivos, como conseqüência satisfazendo suas necessidades e expectativas. Este tipo de procedimento dificulta tanto o aprendizado quanto o uso do sistema, reduzindo assim as probabilidades de que o usuário alcance seus objetivos.

A engenharia da usabilidade propõe que seja adotado um ciclo de desenvolvimento no qual o usuário tenha participação nas considerações da analise, na concepção, no projeto, no desenvolvimento, na implementação e nas revisões do sistema. O objetivo do envolvimento do usuário é trazer importantes benefícios para o projeto, ligados, principalmente, à qualidade e a sua aceitação pelo público-alvo.

Assim devemos considerar que o envolvimento do usuário deve ser uma combinação de níveis, pois através dele deve-se buscar informações, consultá-lo sobre decisões de projeto e deixar eles terem autonomia de tomar determinadas decisões.

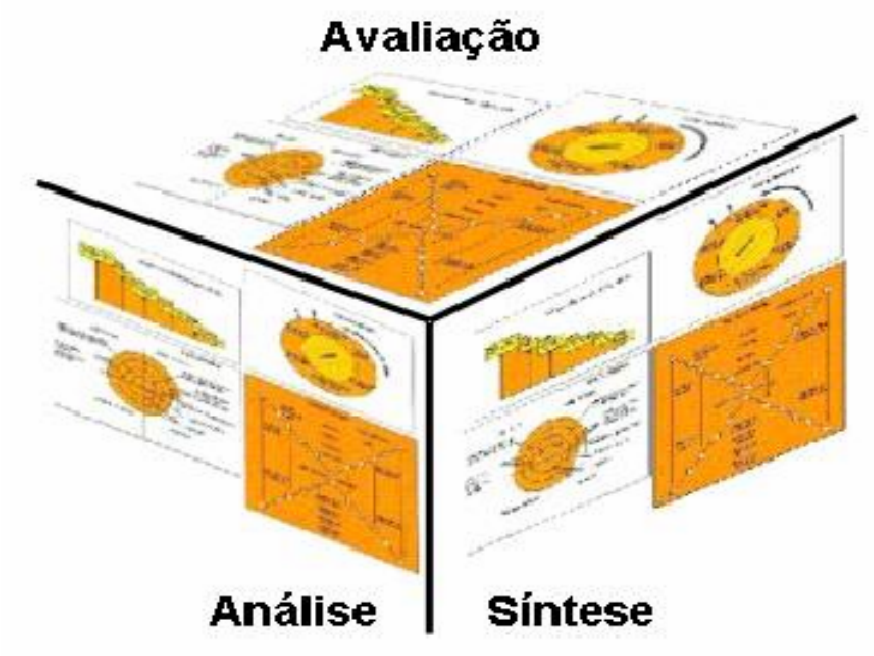
Desta forma para melhor entendimento Cybis (2003, p. 82), define usabilidade através de um conjunto de atividades, que dependendo do paradigma para o ciclo de vida do produto, podem estar encadeadas de diversas formas entre elas em cascata, em ciclos de prototipagem e/ou testes, em espirais evolucionárias ou em diagonais de reutilização, conforme figura 6.



Fonte: Cybis (2003)

Figura 6 – Tipos de ciclo de vida da engenharia da usabilidade

Segundo ressalta Cybis (2003, p.83), a pertinência de um modelo ou outro vai depender do contexto do desenvolvimento, em particular, do caráter inovador das propostas, do conhecimento do domínio do sistema, dos recursos e do tempo disponível, da experiência de equipe, etc. Mas seja qual for o modelo escolhido, as atividades necessárias para o desenvolvimento pertencerão a uma das três categorias ou perspectivas principais : Análise, Síntese e Avaliação, conforme figura 7 a seguir:



Fonte: Cybis (2003)

Figura 7 – Categorias da engenharia da usabilidade

4.5.1 PERSPECTIVA DA ANÁLISE

A perspectiva da Análise se refere ao esforço para estabelecer uma compreensão de uma realidade ou contexto, a partir da sua subdivisão em subsistemas ou componentes.

Assim para o desenvolvimento da usabilidade, a análise não só depende de uma situação existente, mas também de umas ações futuras, que assume uma importância maior uma vez que é por meio destas atividades que se estabelece um foco e um processo de comunicação e de envolvimento do usuário no desenvolvimento. Quanto mais freqüente e progressivo o processo de análise, maior será a qualidade nas decisões de projeto.

Conforme ressalta Cybis (2003, p. 85), para a usabilidade, podem ser descritas as seguintes análises:

a) usuário (categorias, perfil, habilidades, necessidades): devem ser definidos em dois tipos de usuários; as categorias de usuários que são operadores diretos do sistema, ai incluindo todos os que interagem com ele de forma mais ou menos assídua, e os indiretos como gerentes, supervisores, administradores, que são apenas responsáveis pelo sistema;
b) tarefa (objetivos, elementos, condicionantes, estrutura, atributos...): realizada por meio da análise de documentação e de entrevistas com os representantes dos usuários diretos e indiretos. Estes últimos têm uma visão gerencial, muitas vezes teórica e abstrata, dos objetivos e dos métodos que os operadores devem buscar e empregar em seu trabalho;
c) ambientes fisicos, técnicos e organizacionais: refere sobre a síntese do equipamento utilizado, do software, iluminação, ruído e estilo de chefia.

Esta visão é geralmente formalizada pela empresa incluindo dados sobre as interligações e os aspectos econômicos do sistema. Os operadores, por seu lado, possuem uma representação própria de como realizar as tarefas que lhes são solicitadas.

4.5.2 PERSPECTIVA DA SÍNTESE

O processo de síntese de uma interface com o usuário deve ser fruto de uma seqüência lógica de etapas. Desta forma deve-se estar atento às estruturas de atividades como o novo contexto de operações, incluindo a nova estrutura do trabalho a usabilidade de forma quantitativa e qualitativa.

Assim a lógica da perspectiva de síntese (especificação, projeto e implementação) parte de uma nova interface com o usuário. Conforme ressalta Cybis (2003, p.92), seguem as estruturas de atividades que devem ser analisados:

a) especificação do contexto de uso para o qual a interface estará sendo desenvolvida;
b) definição da nova estrutura do trabalho com o novo sistema, resultante da reengenharia do trabalho atual;
c) especificação do desempenho em termos de usabilidade que a interface deve alcançar (incluindo especificações quantitativas e qualitativas);
d) elaboração do projeto da interface;
e) especificação das regras para o projeto gráfico para a interface.

Desta forma o projeto da interface com o usuário se completa com a configuração da apresentação e do comportamento dos objetos de interação relacionados aos comandos, controles e mostradores que povoam a superfície de apresentação. As abordagens para o desenvolvimento de IHC se baseiam em geral, em ciclos de prototipagens e testes.

4.5.3 PERSPECTIVA DA AVALIAÇÃO

Segundo Cybis (2003, p. 106), a usabilidade de avaliação é definida como a capacidade que apresenta um sistema interativo de ser operado, de maneira eficaz, eficiente e agradável, em um determinado contexto de operação, para a realização das tarefas de seus usuários.

Assim, a avaliação de usabilidade de um sistema interativo tem como objetivos gerais validar a eficácia da interação "*humano computador*", efetivando a realização das tarefas por parte dos usuários, contudo ele também pode verificar a eficiência desta interação, utilizando recursos empregados (tempo, quantidade de incidentes, passos desnecessários, busca de ajuda, etc.) e obter indícios da satisfação ou insatisfação (efeito subjetivo) que ela possa trazer ao usuário. Desta forma estes objetivos devem ser pensados em relação aos diferentes contextos de operação previstos para o sistema.

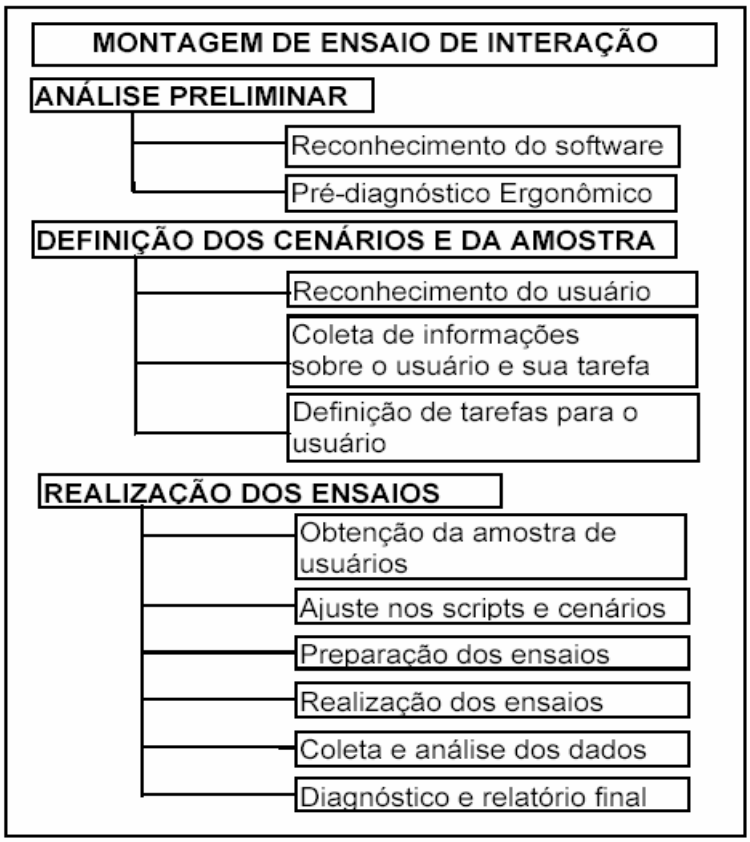
Conforme os objetivos citados, podem aparecer problemas de usabilidade, onde deve ser descrito a partir de informações sobre o contexto de operação onde o problema pode ser encontrado, incluindo a freqüência com que estes problemas se manifestam.

Conforme os estudos realizados por Cybis (2003, p. 107), foi estruturado uma forma de descrever um problema de usabilidade, e este sobre um formato de questionário para poder identificar, conforme segue:

a) identificar o problema;
b) descrevê-lo;
c) determinar o tipo de usuário considerado;
d) determinar o tipo de equipamento;
e) qual o efeito sobre o usuário que o problema está trazendo;
f) qual o efeito sobre a tarefa realizada;
g) indicar a sugestão de melhoria.

Assim nesta proposta, poderão ser identificados os problemas de usabilidade por sua origem no projeto da interface, e não por suas conseqüências durante a interação.

Conforme Cybis (2003, p. 121), a montagem de um ensaio de avaliação pressupõe inicialmente de uma etapa preliminar para conhecer o software e seus atributos ergonômicos, conforme figura 8, a seguir:

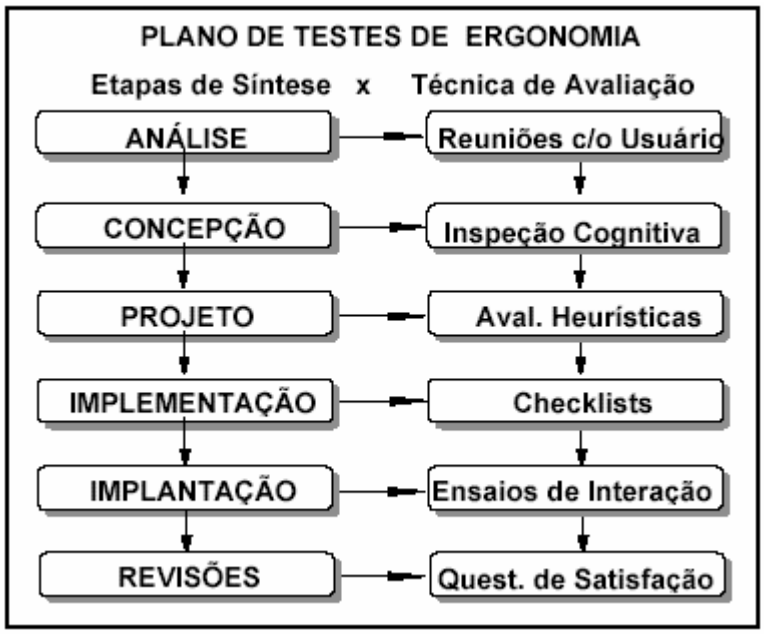


Fonte: Cybis (2003)

Figura 8 – Montagem de um ensaio de avaliação do tipo interação

Nessa etapa os analistas tomam conhecimento dos fatos a cerca do software e de seu contexto de desenvolvimento e realizam assim um pré-diagnóstico dos problemas ergonômicos de sua interface com o usuário.

Assim será criado um plano de testes refere-se a diversos projetos de avaliação, um para cada etapa do desenvolvimento ou versão intermediária do software interativo. Trata-se da combinação de técnicas para testar o projeto e a implementação do sistema, da forma mais abrangente possível, dentro das limitações de recursos disponíveis. Ele deve prever, para cada versão intermediária do sistema, mesmo que não passe de um conjunto de idéias, qual o tipo de teste apropriado, quais as condições de teste e quais os resultados admitidos. A figura 9 mostra uma estratégia de validação ergonômica possível, conforme Cybis (2003, p. 130):



Fonte: Cybis (2003)

Figura 9 – Plano de Testes de Ergonomia

5 SOFTWARE PROPOSTO

5.1 FERRAMENTAS UTILIZADAS

5.1.1 UML

Segundo Furlan (1998), a UML é a linguagem padrão para especificar, visualizar, documentar e construir artefatos de um sistema e pode ser utilizada com todos os processos ao longo do ciclo de desenvolvimento e através de diferentes tecnologias de implementação.

Conforme Fowler (2000), *Unified Modeling Language* (UML) é um método de análise e projeto orientado a objetos (OOA & D) que surgiu no início dos anos noventa. A UML unifica os métodos de Booch, Rumbaugh (OMT) e Jacobson, mas o seu alcance é bem maior. Ela passou por um processo de padronização pela *Object Management Group* (OMG) e é agora um padrão OMG.

A UML é, na verdade, uma linguagem de modelagem e não um método. A maioria dos métodos consiste, pelo menos em princípio, de uma linguagem de modelagem e de um processo. A linguagem de modelagem é a notação (principalmente gráfica) utilizada por métodos para expressar projetos. O processo é a sugestão de quais passos a serem seguidos na elaboração de um projeto, conforme ressalta Fowler (2000).

Conforme Deboni (1998), não se encontra na UML a descrição de passos que se deve seguir para se desenvolver um sistema, nem mesmo quais são as etapas para se modelar um sistema. A UML se limita, exclusivamente, a representar um sistema através de um conjunto de diagramas, onde cada diagrama se refere a uma visão parcial do sistema, que em conjunto forma um todo integrado e coerente.

5.1.2 USOS DA UML

A UML é usada no desenvolvimento dos mais diversos tipos de sistemas. Ela abrange sempre qualquer característica de um sistema em um de seus diagramas e é também aplicada em diferentes fases do desenvolvimento de um sistema, desde a especificação da análise de requisitos até a finalização com a fase de testes, conforme Barros (2000).

Segundo Barros (2000), o objetivo da UML é descrever qualquer tipo de sistema, em termos de diagramas orientado a objetos. Naturalmente, o uso mais comum é para criar modelos de sistemas de software, mas a UML também é usada para representar sistemas mecânicos sem nenhum software. Aqui estão alguns tipos diferentes de sistemas com suas características mais comuns:

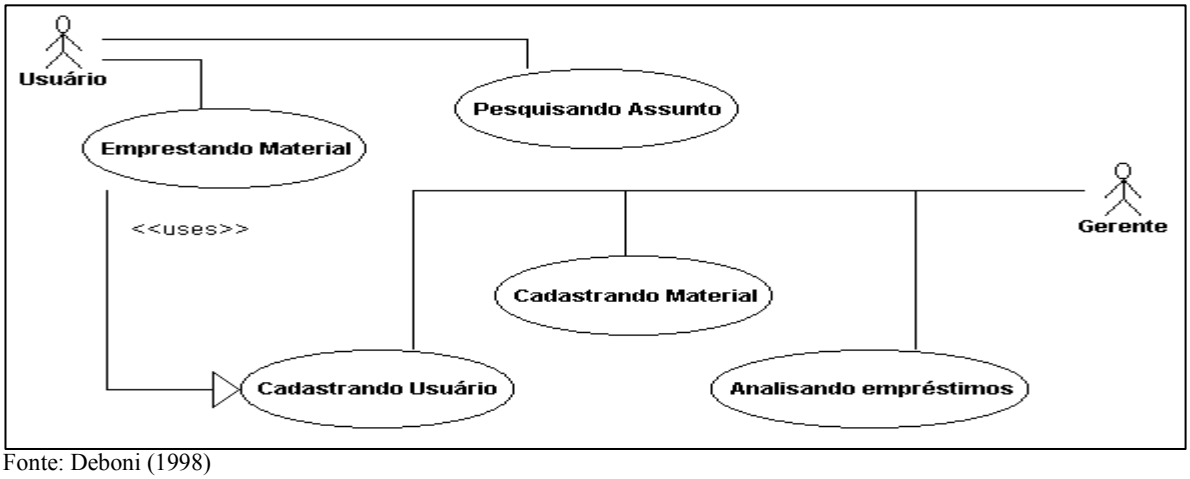
a) sistemas de informação: armazenar, pesquisar, editar e mostrar informações para os usuários. Manter grandes quantidades de dados com relacionamentos complexos, que são guardados em bancos de dados relacionais ou orientados a objetos;
b) sistemas técnicos: manter e controlar equipamentos técnicos como de telecomunicações, equipamentos militares ou processos industriais. Eles devem possuir interfaces especiais do equipamento e menos programação de software de que os sistemas de informação. Sistemas Técnicos são geralmente sistemas real-time;
c) sistemas real-time integrados: executados em simples peças de hardware integrados a telefones celulares, carros, alarmes etc. Estes sistemas implementam programação de baixo nível e requerem suporte real-time;
d) sistemas distribuídos: distribuídos em máquinas onde os dados são transferidos facilmente de uma máquina para outra. Eles requerem mecanismos de comunicação sincronizados para garantir a integridade dos dados e geralmente são construídos em mecanismos de objetos como CORBA, COM/DCOM ou Java Beans/RMI;
e) sistemas de software: definem uma infra-estrutura técnica que outros softwares utilizam. Sistemas Operacionais, bancos de dados, e ações de usuários que executam ações de baixo nível no hardware, ao mesmo tempo em que disponibilizam interfaces genéricas de uso de outros softwares;
f) sistemas de negócios: descreve os objetivos, especificações (pessoas, computadores etc.), as regras (leis, estratégias de negócios etc.), e o atual trabalho desempenhado nos processos do negócio.

É importante perceber que as maiorias dos sistemas não possuem apenas uma destas características acima relacionadas, mas várias delas ao mesmo tempo. Sistemas de informações de hoje, por exemplo, podem ter tanto características distribuídas como real-time; e a UML suporta modelagens de todos estes tipos de sistemas.

5.1.3 DIAGRAMA DE CASOS DE USO

Um caso de uso descreve um objetivo que um ator externo ao sistema tem com o sistema. Um ator pode ser um elemento humano ou não que interage com o sistema. O ator se encontra fora do escopo de atuação do sistema, enquanto o conjunto de casos de uso formam o escopo do sistema. A linha que separa os atores dos casos de uso é a fronteira do sistema, conforme ressalta Deboni (1998).

O diagrama de casos de uso, conforme figura 10, representa graficamente esta interação, e define o contexto do sistema. Os atores são representados por representações simplificadas de uma figura humana, enquanto os casos de uso são elipses contendo cada uma o nome de um caso de uso. Os atores se comunicam com os casos de uso, que é representado por uma linha unindo os dois elementos. Uma seta pode, opcionalmente, representar o fluxo principal de informação nesta interação e ajudar a leitura do caso de uso.



Fonte: Deboni (1998)

Figura 10 - Exemplo de Casos de Uso

Segundo Deboni (1998), como os casos de uso representam um objetivo do ator é comum dar como nome aos casos de uso, frases verbais curtas no infinitivo (EmprestarMaterial) ou no gerúndio (EmprestandoMaterial) onde o sujeito é normalmente o ato, conforme exemplo.:

a) o usuário empresta material;

b) o usuário pesquisa assunto.

Conforme Deboni (1998), cada caso de uso deve receber uma descrição textual que permita o entendimento do objetivo. Esta descrição pode ser detalhada em cenários. Um

cenário é uma instância de um caso de uso, isto é, é uma situação onde o ator utilizou o sistema para conseguir atingir o objetivo do caso de uso. Um cenário pode ser considerado otimista se o ator obteve sucesso no seu objetivo, pode ser pessimista se o ator não conseguiu e ocorreu uma situação de exceção, ou o cenário pode ser alternativo, quando frente a uma situação de exceção o ator optou por caminhos alternativos. Assim para cada caso de uso pode se descrever um texto com:

a) cenários otimistas;

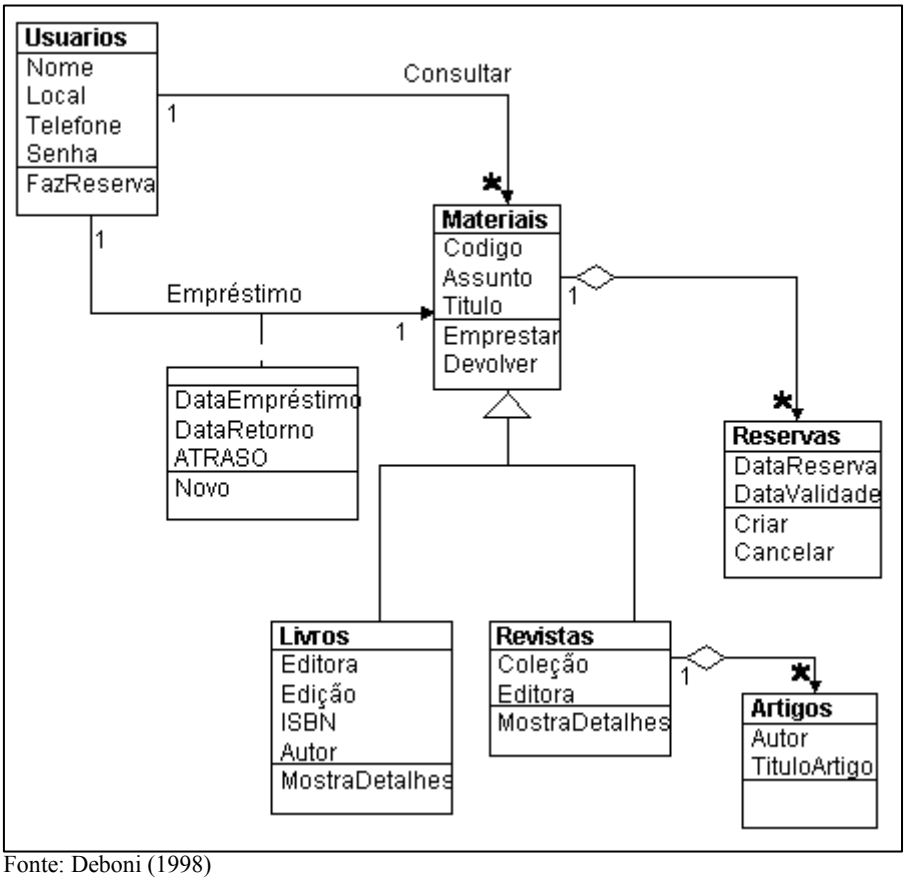
b) cenários pessimistas;

c) cenários alternativos.

5.1.4 DIAGRAMA DE CLASSES

Segundo Deboni (1998), os diagramas de classe descrevem as classes que formam a estrutura do sistema e suas relações. As relações entre as classes podem ser associações, agregações ou heranças. As classes possuem além de um nome, os atributos e as operações que desempenham para o sistema. Uma relação indica um tipo de dependência entre as classes, essa dependência pode ser forte como no caso da herança ou da agregação ou mais fraca como no caso da associação, mas indicam que as classes relacionadas cooperam de alguma forma para cumprir um objetivo para o sistema.

Sendo uma linguagem de descrição, a UML permite diferentes níveis de abstração aos diagramas, dependendo da etapa do desenvolvimento do sistema em que se encontram. Assim, os diagramas de classe podem exibir nas fases iniciais da análise apenas o nome das classes, e em uma fase seguinte os atributos e operações, destacados na figura 11. Finalmente, em uma fase avançada do projeto pode exibir os tipos dos atributos, a visibilidade, a multiplicidade das relações e diversas restrições. Existem elementos na UML para todas estas representações.



Fonte: Deboni (1998)

Figura 11 – Exemplo de Diagrama de Classes.

O diagrama de classes, ao final do processo de modelagem, pode ser traduzido em uma estrutura de código que servirá de base para a implementação dos sistemas. Observa-se, no entanto, que não existe no diagrama de classes uma informação sobre os algoritmos que serão utilizados nas operações, e também não se pode precisar a dinâmica do sistema porque não há elementos sobre o processo ou a seqüência de processamento neste modelo. Estas informações são representadas em outros diagramas, como os diagramas de seqüência de eventos ou diagramas de estado.

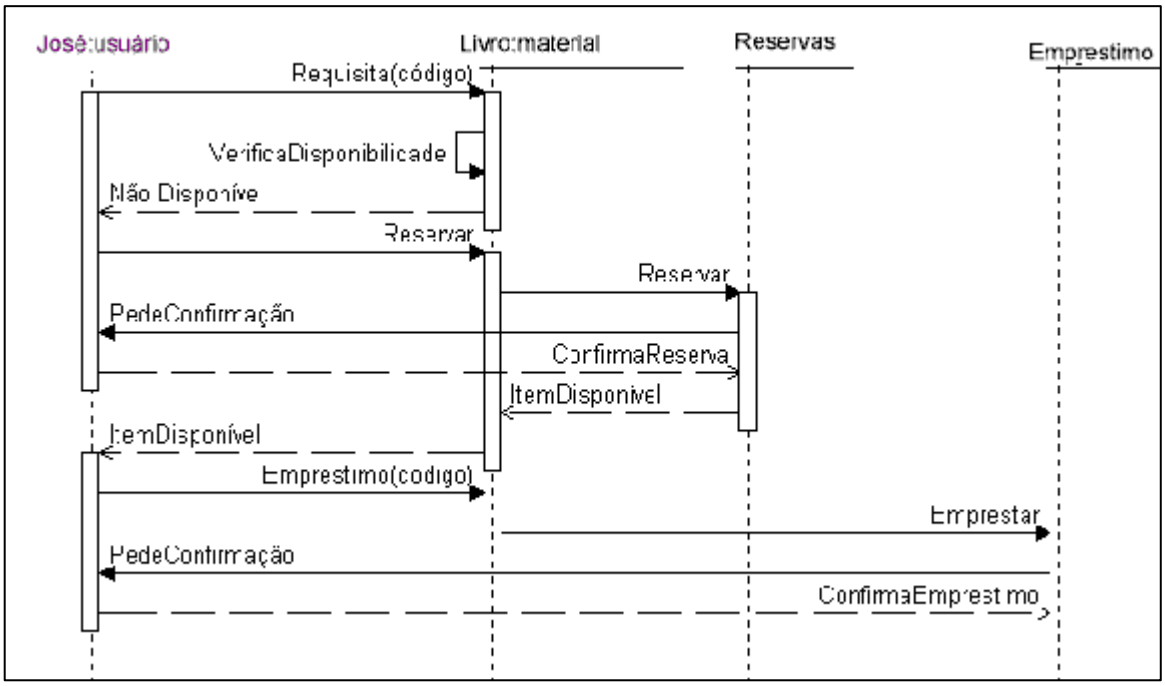
5.1.5 DIAGRAMA DE SEQUÊNCIA DE EVENTOS

Os casos de uso representam conjunto de cenários que descrevem os diferentes processos que ocorrem no sistema. O diagrama de seqüência de eventos permite modelar estes processos através da troca de mensagens (eventos) entre os objetos do sistema. Os objetos são representados por linhas verticais e as mensagens como setas que partem do objeto que invoca um outro objeto. As setas podem ser cheias para indicar uma mensagem de chamada ou tracejadas para indicar uma mensagem de retorno. A figura 12 mostra um exemplo deste diagrama, onde se pode observar que o tempo segue o eixo vertical de cima para baixo.

Devem ser desenhados tantos diagramas de seqüência quantos cenários foram levantados no diagrama de casos de uso, conforme ressalta Deboni (1998).

Fonte: Deboni (1998)

Figura 12 - Exemplo de um Diagrama de Seqüência de Eventos



Cada mensagem no diagrama de seqüência de eventos corresponde a uma operação no diagrama de classes. Como as mensagens são operações invocadas, elas devem estar presentes nos objetos de destino, que são ativadas pelas mensagens no objeto de origem.

5.1.6 RATIONAL ROSE

Segundo DBSERVER (2001), Rational Rose é uma ferramenta CASE para desenvolvimento de sistemas orientados a objetos, sendo também, um repositório para os produtos gerados por processos de análise e projetos orientados a objetos. Ela acelera esse desenvolvimento de análise e projetos utilizando metodologias de desenvolvimento muito difundidas no meio da informática, principalmente o padrão Unified Modeling Language (UML). O padrão UML é o mais utilizado hoje em dia, pois com ele há uma documentação interativa e automática dos processos de modelagem, sendo também possível à geração, pela ferramenta, de códigos a partir da mesma modelagem. As versões mais novas da ferramenta Rational Rose, contemplam até a interação com ambientes de desenvolvimento das mais variadas linguagens existentes no mercado.

Conforme DBSERVER (2001), a Rational Rose é a principal ferramenta utilizada para construção de projetos baseadas em análises orientadas a objetos.

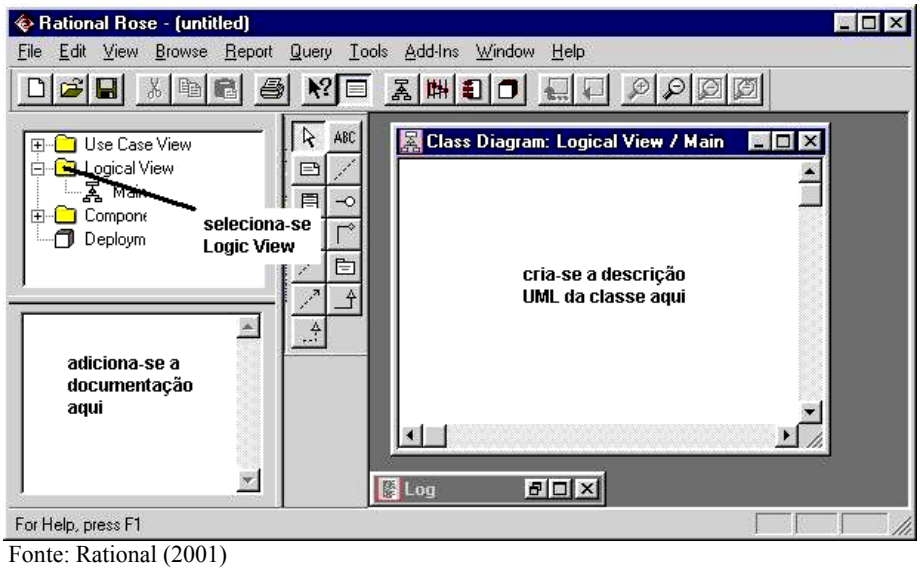
Assim dizemos que o Rational Rose é uma ferramenta CASE que utiliza a linguagem de modelagem de sistemas UML. Quando um objeto é criado no Rational Rose, este é chamado de modelo. Abaixo podemos ver uma breve descrição do ambiente:

A figura 13 mostra a aplicação Rational Rose como aparece quando é aberta pela primeira vez, assim podemos verificar três janelas independentes:

a) *ao topo à esquerda*: a Janela de Visão permite que o usuário determine que perspectiva do projeto será usada. Selecionando *Logical View* nessa janela indica que o usuário deseja olhar as classes com seus atributos e operações;

b) *abaixo à esquerda*: Janela de Documentação. É usada para adicionar textos descrevendo as classes;

c) *à direita*: Janela de Diagrama de Classes. É usada para criar a descrição de classes em UML.



Fonte: Rational (2001)

Figura 13 - Tela inicial do Rational *Rose*

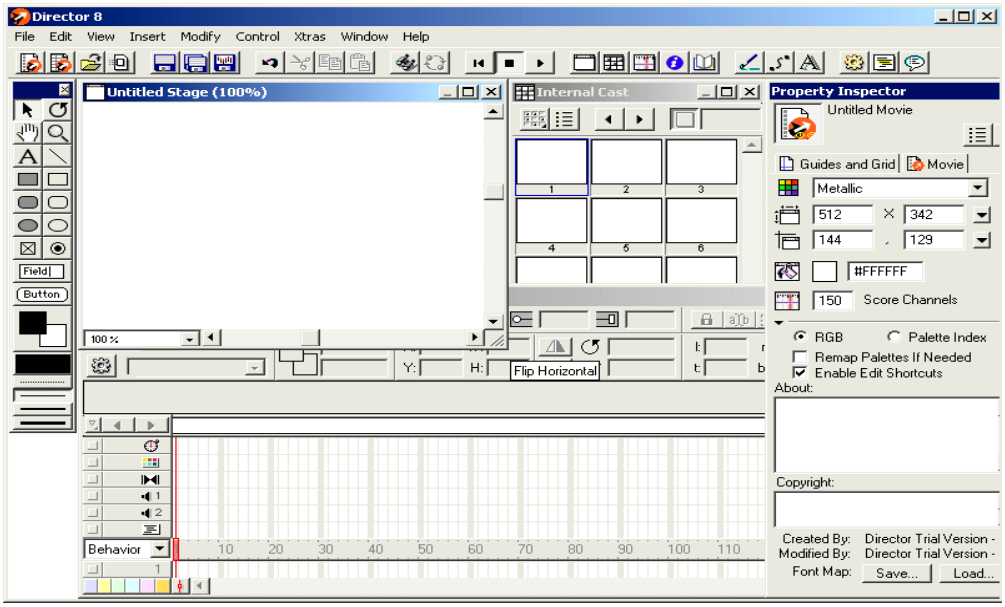
No menu *Browse* escolhe-se o tipo de janela que o usuário irá trabalhar. Esta é uma parte importante para a especificação no *Rational* Rose, é neste menu que o usuário define o tipo de análise que será feito, com as opções: *Class Diagram* (Diagrama de Classes), *Use Case Diagram* (Diagramas de Casos de Uso), *Interaction Diagram* (Diagrama de Interação ou de Seqüência de Eventos), *Component Diagram* (Diagrama de Componentes), *State Diagram* (Diagrama de Estados) e *Deployment Diagram* (Diagrama de Organização).

Para este trabalho foram utilizados os Diagramas de Classes, Casos de Uso e de Interação, onde é descrita a especificação do software proposto sendo ilustrados os diagramas modelados em Rational *Rose* com a Linguagem UML.

5.1.7 AMBIENTE DE DESENVOLVIMENTO: DIRECTOR 8

Segundo Bizzotto (2000, p. 23), o Macromedia Director 8 é uma das ferramentas mais utilizadas para o desenvolvimento de softwares hipermídia. Alem disso, esta utilização vem aumentando sensivelmente em função da grande evolução do Director nos últimos anos, pois vem incorporando novas e importantes características, principalmente com relação à integração com a Internet.

O desenvolvimento no Director é análogo ao desenvolvimento de um filme de cinema ou de uma peça de teatro. Assim, deve-se definir o tipo de filme (software) a ser desenvolvido; os atores (o elenco) que irão participar; o papel de cada ator; a seqüência de cenas, etc. A figura 14 apresenta alguns dos elementos citados como stage (cenário), CastMembers (atores), scripts (papel de cada ator), Score (seqüência de cenas), que serão posteriormente detalhados.

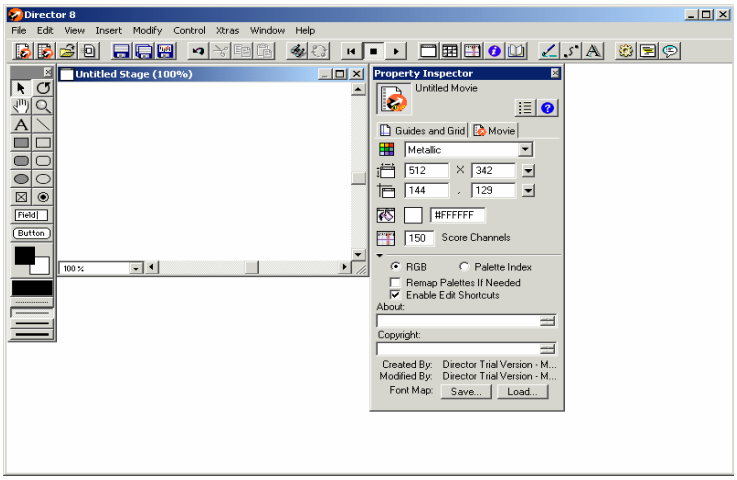


Fonte: Bizzotto (2000).

Figura 14 – Tela do Director 8

5.1.8 STAGE

Segundo Bizzotto (2000, p. 71), no Director, tudo ocorre no *stage*, ou seja, no palco (fazendo uma analogia com o teatro). No *stage*, como mostra a figura 15, são colocados todos os itens (cenários, atores etc) que irão compor a cena a ser desenvolvida. No *stage* podem ser definidas as propriedades do filme atual (*Modify – Movies Properties*), tais como, tamanho do *stage* de apresentação, a localização, a paleta *Default*, entre outros.



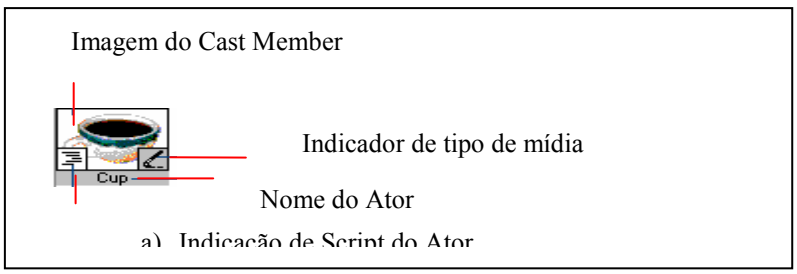
Fonte: Bizzotto (2000)

Figura 15 - A Janela *STAGE*

No *Stage*, podem ser definidas as propriedades do filme atual (*Modify – Movie-Properties*), tais como, tamanho do *Stage* de apresentação, a localização, a paleta *Default*, entre outros.

5.1.9 JANELA CAST

A janela *Cast*, contém os *cast members*, que são os atores que atuarão sobre o *Stage*. Esse elenco de atores pode ser formado por imagens, textos, sons, etc. Um membro no *Cast Member*, possui informações conforme as indicadas na figura 16, a seguir:



Fonte: Bizzotto (2000)

Figura 16 – Membro do Cast

Um mesmo membro do *Cast member*, pode ser posicionado em locais diferentes do *Stage*, sendo chamado de "Instância de *Cast Member*", ou seja, uma cópia do original. Logo, as alterações feitas em um original, afetarão suas equivalentes cópias. Por outro lado, uma

alteração feita em uma cópia do original, não afetará o *Cast* original. Embora possam existir várias cópias no *Stage* de um mesmo elemento do *Cast*, estes elementos podem assumir papéis diferentes no *Stage*.

De acordo com o tipo de mídia existente no *Cast*, temos representado na tabela 1, a origem destes tipos de mídia (criados ou importados):

Tipos	Ícones	Descrição	Criado no Director	Importado
Animated Gif		Gif Animado	NÃO	SIM
Bitmap		Imagem	SIM	SIM
Check Box		Caixa de verificação	SIM	NÃO
Digital Vídeo		Vídeo Digital	SIM	SIM
Film Loop		Seqüência de Animação	SIM	NÃO
Flash Movie		Arquivo Flash	NÃO	SIM
Behavior		Componente de Biblioteca	SIM	SIM
Button		Botão	SIM	NÃO
Custon Cursor		Cursor Personalizado	SIM	NÃO
Field		Campo	SIM	NÃO
Font		Fonte	NÃO	SIM
Linked Bitmap		Link de um arquivo bitmap	NÃO	SIM
OLE		Objeto OLE	NÃO	SIM

Tipos	Ícones	Descrição	Criado no Director	Importado
PICT		Imagem	NÃO	SIM
Radio Button		Botão de Rádio	SIM	NÃO
Shape		Imagem do tipo Shape	SIM	NÃO
Sound		Arquivo de Som	NÃO	SIM
Transition		Efeitos transição de sprites	SIM	SIM
Xtra		Plugin	SIM	SIM
Palette		Paleta	SIM	SIM
Quick Time Video		Vídeo Digital	NÃO	SIM
Script		Script Lingo	SIM	NÃO
Shockwave		Som compactado p/ Internet	NÃO	SIM
Text		Texto	SIM	SIM
Vector Shape		Imagem Vetorial	SIM	SIM

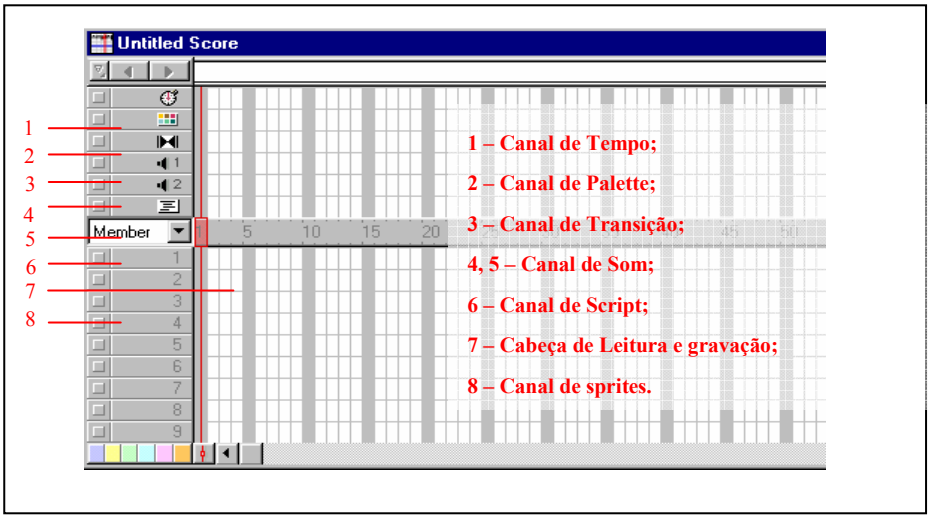
Fonte: Bizzotto (2000)

Tabela 1 – Tipos de elementos do Cast

5.1.10 SCORE

Conforme mencionado anteriormente, os *Cast Members* contém os atores do *Stage*. A janela *Score* controla quando e como as ações vão ocorrer.

O *Score* é dividido em linhas (canais) e colunas (frame). A interseção de uma coluna com uma linha, denominamos célula. Além dos canais usuais para o gerenciamento dos *Cast Members*, ele possui seis canais especiais para efeitos, conforme figura 17.



Fonte: Bizzotto (2000)

Figura 17 - Canais do Score

Os itens dispostos na figura 17, são responsáveis: (1) pelo gerenciamento do tempo da apresentação e acontecimentos no *Stage*, (2) a possibilidade de utilização de um certo grupo de cores (palette) em um dado filme, (3) gerar um efeito de transição de uma cena para outra, (4,5) utilização de sons ou narrações, (6) permitir a programação em *Lingo*, (7) acompanhar a execução dos *frames*, (8) mostrar todos os *Cast Members* que irão aparecer no *Stage*, respectivamente.

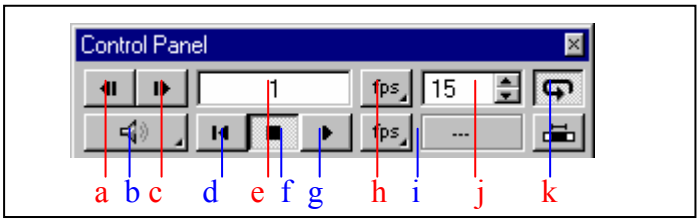
5.1.11 CONTROL PANEL

Segundo Bizzotto (2000, p. 98), a janela *Control Panel* é uma janela especial que funciona como se fosse o controle de um videocassete. Além destes controles, esta janela oferece importantes informações sobre o filme. Através desta janela é possível "rodar" um filme (ou parte dele) para avaliar se ele se comporta conforme planejado.

A janela *Control Panel*, é comumente utilizada durante o desenvolvimento, ao executar, parar, avançar ou retroceder um filme. Permite ainda o controle de repetições, no caso de optar-se por uma execução contínua do filme, e também a habilitação ou não dos sons, conforme figura 18.

Os componentes que compõe o Control Panel são:

a) botão *step backward*: retrocede um *frame*;
b) botão *sound*: controla a intensidade de som no filme;
c) botão *step forward*: avança um *frame*;
d) botão *rewind*: retorna para *o frame* 1;
e) *frame counter*: indica onde se encontra a cabeça de leitura e gravação;
f) botão *stop*: interrompe a exibição do filme;
g) botão *play*: inicia a exibição do filme;
h) botão de modo de tempo: permite o andamento do filme em *frames* por segundo;
i) botão modo de tempo real: velocidade real da apresentação do filme;
j) tempo *display*: indica o andamento do filme;
k) loop *playback*: permite que o filme fique em loop, executando.



Fonte: Bizzotto (2000)
Figura 18 – Control Panel.

5.1.12 LINGO

Lingo é o nome da linguagem utilizada pelo Director, para a produção mais efetiva junto ao usuário.

Segundo Bizzotto (2000, p. 314), através do uso do Lingo, tem-se um maior controle do que acontece no filme, podendo, inclusive, alterar o que foi feito a partir do *score*. Assim, o Lingo é o que torna o Director mais robusto e flexível.

Conforme Small (1999), o Lingo permite a programação orientada a objetos, embora não seja considerada uma linguagem orientada a objetos. Através das variáveis globais privadas, cada objeto possui características próprias.

Assim a terminologia utilizada difere no Lingo, sendo comparados na tabela 2.

Termo no Lingo	Equivalente em Orientação Objeto
Parent Script	Classe (*Class*)
Child Object	Instância de classe (*Class instance*)
Property variable	Variável de instância (*instance variable*)
Handler	Método
Ancestor script	Super classe

Fonte: Dalfovo (1999)

Tabela 2 – Terminologia utilizada

Segundo Bizzotto (2000, p. 314), como nas demais linguagens de programação, o Lingo segue algumas regras:

a) todo *script* é formado por handlers, que em outras linguagens normalmente são definidos como sub-rotinas ou funções. Um handler inicia-se com *on* e é encerrado por *end*;
b) variáveis locais: utilizadas somente dentro do *handler* onde foi criada;
c) variáveis globais: utilizadas por diferentes *frames* de um filme, ou de diferentes filmes;
d) ordem de precedência de *handlers.*

Sendo assim podemos destacar ainda algumas características que possibilitam a orientação a objetos:

a) a função "NEW" permite cópias idênticas de um determinado objeto através de um script denominado Parent Script;
b) através das propriedades, Property, os objetos mantêm suas próprias variáveis globais privadas;
d) os objetos podem dividir propriedades ou handlers através do Ancestor.

5.1.13 SCRIPTS

Os *scripts* do Lingo podem variar quanto a sua localização e tipo, conforme Bizzotto (2000, p. 325):

a) *behavior script*: são scripts associados a sprites (sprite behaviors) ou a frames (frame behaviors).
b) *cast member script*: associado a um dado *cast member*, de forma que o *script* será executado em qualquer *sprite* do qual aquele *cast member* faça parte;
c) *movie script*: são *scripts* que "valem" para todo o filme. Assim, eles são mais gerais, não se limitando a um dado *sprite* ou conjunto de *sprites* ou *frames*.

5.1.14 EVENTOS

Segundo Bizzotto (2000), no momento em que o usuário vai utilizar o software, ocorrem os seguintes eventos no instante da inicialização:

a) p*repare movie*: este é o primeiro evento que ocorre quando um filme é executado, ou seja, o filme está sendo preparado para ser rodado;
b) b*eginSprite*: indica que a cabeça de leitura e gravação entrou em um segmento de *sprite* (*sprite span*), ou seja, indica que um *sprite* foi iniciado;
c) p*repareFrame*: ocorre antes do Director apresentar um *sprite* no *stage*. Assim, neste evento podem-se incluir ações que alterem as propriedades do *sprite*, uma vez que ele ainda não foi "desenhado";
d) s*tart Movie*: este evento ocorre no primeiro *frame* do filme.

A partir do momento que o evento *StartMovie* ocorreu, o Director passa a executar os *frames* do filme, ocorrem os seguintes eventos:

a) *beginSprite*: indica que a cabeça de leitura e gravação entrou em um segmento de *sprite* (*sprite span*). Este evento ocorre sempre que existe um novo segmento de sprite;
b) *prepareFrame*: o *frame* está sendo preparado para ser "desenhado" no *stage*;
c) *enterFrame*: ocorre depois que o *sprite* é "desenhado" no *stage* e antes que a cabeça de leitura e gravação deixe o *frame* atual;
d) *exitFrame*: ocorre quando a cabeça de leitura e gravação deixa um dado *frame*;
e) *endSprite*: ocorre quando a cabeça de leitura e gravação deixa a última célula de um segmento de *sprite*.

Estes eventos são independentes da ação do usuário, ou seja, não é necessária a intervenção do usuário para que estes eventos ocorram. Mas, existem eventos que dependem da ação direta do usuário. Estes eventos são, conforme Bizzotto (2000, p.323):

a) *mouseEnter*: ocorre quando o cursor entra no espaço de um *sprite*;

b) *mouseWithin*: ocorre enquanto o ponteiro do *mouse* estiver sobre um dado *sprite*;

c) *mouseLeave*: ocorre quando o *mouse* deixa o espaço de um *sprite*;

d) *mouseDown*: ocorre quando o usuário pressiona o botão do *mouse*;

e) *mouseUp*: ocorre quando o usuário solta o botão do *mouse*;

f) *mouseUpOutside*: ocorre quando o usuário pressiona o botão do *mouse* sobre um *sprite*, mas solta o botão fora deste *sprite*;

g) *rightMouseDown*: ocorre quando o botão direito do mouse é pressionado;

h) *rightMouseUp*: ocorre quando o botão direito do mouse é liberado;

i) *keyDown*: ocorre quando o usuário pressiona uma tecla;

j) *keyUp*: ocorre quando o usuário solta uma tecla.

5.1.15 LISTAS

Conforme Small (1999) , as listas em Director, ou *arrays* em outras linguagens, são "variáveis" que podem conter diversos elementos. A alocação dos elementos na memória é feita segundo a necessidade desta, não denso, portanto, necessárias às informações quanto ao tamanho dos elementos da lista. O primeiro elemento da lista no Director é contado como "1", diferentemente das demais linguagens que consideram o primeiro elemento de lista "0". A tabela 3 mostra sintaxes para criação de listas.

Sintaxe	Criando uma lista
***Set the list to* [ ]**	Para criar uma lista linear vazia
***Set the list to* [:]**	Para criar uma propriedade de lista vazia, pode-se usar um operador de lista para especificar valores na lista.
Especifique a lista de elementos com os parâmetros da função *list*(). *List*(*value1*, *value2*, *value3*...)	Para criar uma lista linear usando a função list( ). Exemplo: *Set designers = list*("*Gee*", "*kayne*", "*Ohashi*")

Fonte: Small (1999).

Tabela 3 – Sintaxe para a criação de listas

5.2 DESENVOLVIMENTO DO SOFTWARE

5.2.1 PRINCIPIOS DA PESQUISA

Conforme Oliveira (1996), a introdução do computador na escola remete a dois objetivos:

a) aprendizagem do uso do computador pelo aluno, caracterizando o ensino da informática enquanto técnica e ciência;
b) aprendizagem pela e com a informática, caracterizando seu uso como ferramenta didático-pedagógica no desenvolvimento do processo ensino aprendizagem, possibilitando inúmeras formas de tratar o conhecimento, bem como criar ambientes de aprendizagem.

Com a evolução da tecnologia e do grau de exigência dos usuários, os ambientes educacionais passaram por muitas modificações, deixando para trás as abordagens clássicas de tutoriais, simulações e jogos. Estas modificações foram decorrentes, dentre outros aspectos, da crescente influência da psicologia sobre o desenvolvimento de softwares educacionais. Isso influenciou não só os objetivos mas também a própria concepção dos softwares.

Atualmente, conhecer a qualidade e a eficácia dos ambientes educacionais desenvolvidos remete, necessariamente, à avaliação desse produto com relação ao grau de adequação no processo de ensino e aprendizagem, bem como à qualidade ergonômica que os mesmos proporcionam.

Conforme Koslosky (2000), a escolha do meio educacional está baseada, fortemente, no conteúdo, sem uma adequada ênfase na avaliação dos atributos da qualidade do produto.

Assim, para contribuir na resolução deste problema, a área de estudos da ergonomia de softwares mostra o caminho a ser perseguido por educadores e projetistas, desde a concepção até a utilização do produto, tendo como prioridade três componentes, conforme ressaltado por Cybis (2003, p. 07):

a) projeto educacional;
b) projeto comunicacional;
c) desenho computacional.

Portanto, torna-se necessário estabelecer critérios para poder avaliar a eficácia e a eficiência desses produtos. Para isso, existem modelos e teorias tentando, da melhor maneira, abordar a questão da avaliação de ambientes educacionais.

Assim os objetivos de projetistas e educadores são complementares, na medida em que os educadores buscam adaptar os meios didáticos para obter uma aprendizagem significativa por parte dos alunos enquanto os projetistas têm como objetivo organizar a interação do aluno com o software de forma intuitiva. Para que estes objetivos complementares sejam atingidos tanto o educador quanto o projetista (em graus e enfoques diferentes) devem estar atentos às recomendações ergonômicas.

5.2.2 DESENVOLVIMENTO DA PESQUISA

Com a difusão de ambientes educacionais, alunos, professores e instituições, têm cada vez mais utilizado o computador como ferramenta de ensino-aprendizagem. Muitas vezes, no entanto, a escolha desses produtos acontece de forma indiscriminada, sem que haja uma abordagem sistemática para a avaliação da adequação do software no qual ele será utilizado. Adicionalmente, uma parcela significativa de professores e alunos é levada a navegar em ambientes educacionais sem terem conhecimentos suficientes de informática (usuários novatos) e/ou de como utilizar pedagogicamente o programa.

Este é exatamente o contexto no qual se enquadra o presente trabalho, o qual irá utilizar as propostas da Engenharia da Usabilidade para desenvolver um software educacional que permita aos alunos uma aprendizagem significativa. Com isso, será possível avaliar a adequação da abordagem proposta e sua influência sobre o aprendizado do aluno.

5.2.3 DELIMITAÇÃO DO PÚBLICO ALVO

Torna-se importante ressaltar que o público-alvo do software desenvolvido são crianças de 6 a 8 anos, que freqüentam a Associação Assistencial dos Deficientes Auditivos e Visuais (AADAV) em Jaraguá do Sul/SC.

Contudo o problema está delimitado em construir um software multimídia, que estimulasse estas crianças quanto ao uso e aplicação da informática. Devido estas crianças possuírem dificuldades no seu processo de ensino-aprendizagem, decidiu-se durante o projeto buscar auxílio e acompanhamento de um grupo de estagiárias do curso de psicologia.

Desta forma as estagiarias estariam envolvidas no desenvolvimento das crianças, intervindo com a psicologia quando fosse necessária sua aplicação.

5.2.4 INFORMAÇÕES SOBRE A INSTITUIÇÃO - AADAV

A AADAV está localizada na cidade de Jaraguá do Sul, sendo uma organização não governamental sem fins lucrativos e de personalidade jurídica de direito privado. Foi fundada em 17 de julho de 1987 por iniciativa de pais e professores.

A instituição tem por objetivo propor e executar políticas de atendimento que contribuam para o desenvolvimento integral de seus usuários (os que apenas procuram a instituição para saber sobre programas e serviços oferecidos) e associados (os que realmente se associaram a ela).

Para atender as demandas postas por seus usuários, a AADAV conta com uma equipe de profissionais capacitados para instrumentalizar suas ações. Atualmente esta equipe é composta de: três fonoaudiólogas, duas assistentes social, um médico especialista em otorrinolaringologia, dois psicólogos, um facilitador de informática para pessoas com deficiência auditiva e visual e uma alfabetizadora para pessoas que apresentam alguma deficiência auditiva.

Desta formas as ações realizadas por estes profissionais, são na maioria das vezes operacionalizadas através da execução de diversos programas e serviços, os quais apresentam:

a) *oficina ocupacional*: ênfase para o encaminhamento para o mercado de trabalho;
b) serviço de terapia ocupacional: se aplicam métodos e técnicas terapêuticas com o objetivo de desenvolver e conservar habilidades físicas, mentais e psicossociais dos usuários;
c) *serviços de fototerapia para pessoas com deficiências auditivas*: com ênfase para o desenvolvimento das linguagens, oral e escrita;
d) *oficinas de Artes*: com o objetivo de trabalhar as potencialidades artísticas de cada pessoa com deficiência sensorial, através de coral, dança, teatro e artes plásticas;
e) *serviço conhecido como Escola de Pais*: que oferece aos pais das pessoas com deficiência, uma melhor compreensão a cerca dos processos de reabilitação de seus filhos;
f) *informática educativa*: pode ser vista como um dos programas e/ou serviços oferecidos, a qual proporciona aos educandos com deficiência o acesso às novas

tecnologias, bem como ao ensino de informática através de metodologias e instrumentos específicos para atendimento a esta demanda;

g) *serviço social*: que enfatiza a implementação e a execução de políticas públicas que garantam o atendimento integral às necessidades básicas de seus usuários;
h) *serviço de Fonoterapia Especializada*: objetivo atender usuários referenciados pela rede básica de saúde, que apresentam distúrbios de ordem fonoaudiológica, como patologias de linguagem oral e escrita.

Uma das atividades desenvolvidas na AADAV é também o estágio curricular, modalidade esta que é acompanhada e supervisionada por profissionais da entidade e das unidades de ensino. Atualmente, a instituição disponibiliza dois campos de estágio, um em Serviço Social, onde contam com estagiários da FURB (Universidade Regional de Blumenau) e outro em Psicologia, onde conta com a participação de estagiários da ACE.

Neste sentido, a direção da AADAV e o curso de Psicologia da ACE formaram uma parceria, em 2003, abrindo na instituição jaraguaense um novo campo de estágio na área de Psicologia Escolar, o que possibilitou a inserção de seis acadêmicos do quinto ano de 2003; sendo que no último semestre do mesmo ano deu início em paralelo a inserção de cinco acadêmicas do quarto ano para acompanhar o estágio em andamento, e dar continuidade no ano em 2004.

A proposta destas acadêmicas é agrupar as crianças em grupos de três crianças, fornecendo um atendimento por estagiária durante a semana. Para tanto, foi verificado quanto à possibilidade das crianças de estarem indo aos dias e horários que estabelecidos, levando em conta os horários de atendimento das fonoaudiólogas, para serem realizados na mesma jornada.

5.2.5 ANALISE DO SOFTWARE PROPOSTO

A análise do novo sistema partiu das relevâncias que os usuários em potencial não teriam nenhuma habilidade na arte de manusear o computador, com isso contou-se com a participação de três das seis estagiárias de psicologia que formam a parceria com a instituição da AADAV.

Aplicando as sugestões das estagiarias durante as reuniões que aconteciam quinzenalmente, foram levantados alguns aspectos relevantes quanto à usabilidade de como

seria um software apropriado. Abaixo as principais relevâncias que o novo sistema deveria conter:

a) interatividade: gerar um sentido de participação interativa utilizando-se de técnicas de apresentação e de alguns recursos que facilitam a discussão informal entre os usuários;
b) apresentações visuais: definido como um objeto de software cujo processamento gera uma imagem que é apresentada ao usuário e com a qual ele pode interagir;
c) textura/cores: uso de cores para transmitir informações, chamar a atenção, contrastar e associar objetos para uma melhor interação;
d) interface gráfica: objetos que fornecem ao usuário um cenário adequado, em termos dos diferentes tipos de mostradores, de controles e de comandos necessários para a realização de suas ações ou tarefas.
e) menu navegação: seleção de opções de fácil manuseio que acionam os comandos do programa aplicativo;
f) objetivos claros: os participantes devem acreditar que o tempo que dedicam às interações vale a pena;
g) relatórios de respostas: resposta rápida a cada contribuição ou ação realizada pelo usuário;
h) ambientação entre colegas: encorajar os usuários novatos no uso do mouse a trabalhar com colegas mais experientes.

5.3 IMPLEMENTAÇÃO DO SOFTWARE

A implementação do software deu-se através da realização da analise, seguindo a orientação do item (5.2.5).

Desta forma procurou-se desenvolver um sistema prático que pudesse colocar em prática as habilidades cognitivas necessárias para a utilização e manuseio do *mouse*. O tipo de software escolhido foi à forma de um jogo educacional, porque possui a vantagem de desenvolver um conceito ou uma habilidade sem que o aluno se dê conta disso.

Assim os principais eventos do sistema quanto ao manuseio do *mouse* seriam as funções do clique, clique-duplo e clicar-mover.

Com estas observações partiu-se para a fase de implementação, onde foram definidos 4 módulos distintos, conforme adequação do sistema mencionado no item anterior:

a) jogo clique;
b) jogo clique-duplo;
c) jogo clicar-mover;
d) relatório de desempenho.

Desta forma, cada evento criado para manusear o *mouse*, era repassado para observação e aprovação das estagiárias, onde era verificada toda a performance do sistema, avaliando cenários, interatividade e dificuldade de jogo. Com as observações que eram apresentadas, as fases de análise, concepção e inspeção cognitiva tornaram-se freqüente durante todo o período de validação do sistema.

5.3.1 IMPLANTAÇÃO DO SOFTWARE

Para implantar o software, foi considerado que a abordagem do desenvolvimento seria realizada levando em conta características do público-alvo.

Assim podemos dizer que em um sistema, em geral, existem dois tipos de usuários: as categorias de usuários que são operadores diretos do sistema, incluindo todos os que interagem com ele de forma mais ou menos assídua, e as dos indiretos como professores e orientadores, que são apenas responsáveis pela aplicação sistema.

Torna-se importante ressaltar que neste módulo está atribuído o monitoramento do desempenho do aluno. O software acumula, enquanto o aluno joga, os erros, os acertos, o número de vezes que ficou em uma mesma fase e o tempo gasto. O monitor pode, a qualquer momento, acessar o relatório do desempenho do aluno, uma vez que o software armazena essa informação em um arquivo texto externo.

5.3.2 INTERAÇÃO DO SOFTWARE

Disposto um período de duas semanas para o uso do novo sistema , se observou como era a ação do usuário no processo de manuseio com o jogo do *mouse.* Desta forma os principais itens pertinentes encontrados foram:

a) performance do usuário perante os processos repetitivos: devido os alunos não possuírem contato com a informática, a utilização do novo sistema ficou bastante

dificultada. Mas após orientação e persistência realizada pelas educadoras, os usuários em potencial se sentiram encorajados a continuar a praticar o sistema;

b) localização dos objetos na nova interface: nesta etapa as educadoras realizaram uma análise, de modo avaliar o tempo ideal para prática do jogo em execução;
c) comandos do novo sistema: nesta etapa, foram definidos qual seria o tamanho e onde estariam localizados os botões de navegação do jogo.

5.3.3 REVISÃO DO SOFTWARE

Para a fase de revisão, quanto a implantação do novo sistema, as adequações foram sugeridas seguindo a lista de critérios ergonômicos, praticados pelo laboratório de Usabilidade (LabiUtil) da Universidade Federal de Santa Catarina, que possui um *checklist* que contribui para determinar a aplicar as devidas alterações no sistema. Em sua *homepage* encontram-se uma atualização do referido *checklist*, o qual é colocado à disposição daqueles que desejam utilizá-lo.

Este *checklist* foi adotado para avaliar as modificações do sistema, obedecendo os critérios de avaliação, com base na presteza, legibilidade, compatibilidade e ações dos usuários quanto a utilização. Desta forma a tabela 4, traz as alterações apontadas pelo *checklist,* que não estavam em conformidade com este trabalho.

CRITÉRIO DE AVALIAÇÃO	ITENS NÃO CONFORME	ALTERAÇÕES REALIZADAS
Presteza	Os gráficos não possuíam título nas suas janelas	Criação de título em todas as janelas
	O usuário não encontra disponíveis as informações necessárias para suas ações	Criação de janelas com informações de como praticar as ações
Legibilidade	Os códigos alfanuméricos do sistema não agrupam separadamente letras e números	Adaptado para realizar separação de letra e numero no momento que for cadastrado o nome do usuário
	Ícones não apresentam legibilidade	Clareza na informação dos ícones (textos)
Ações Mínimas		Atende todos os itens
Compatibilidade	Telas não compatíveis com o ambiente	Criação de telas animadas
	Não apresentam botão de anulação	Criação de um botão de anulação e validação
	As mensagens não são firmadas após realização de algum evento	Aplicado mensagem após algum evento realizado

CRITÉRIO DE AVALIAÇÃO	ITENS NÃO CONFORME	ALTERAÇÕES REALIZADAS
Significados	Título das janelas não estão compatíveis com o que realmente representam	Janelas com nomes compatíveis ao que realmente representam
	Denominações das opções de menu não são familiares ao usuário	Criação janelas com desenhos

Tabela 4 – Aplicação da lista de critérios ergonômicos

Com isso outro fator que facilitou aplicação desta etapa, foi a facilidade que a ferramenta Director, apresenta quanto as suas alterações.

5.3.4 ESPECIFICAÇÃO DO SISTEMA

Para especificação foi utilizada a ferramenta Rational Rose por conter a maioria dos elementos existentes para o desenvolvimento do projeto conceitual. As classes representam os diferentes módulos propostos para o ensino e aprendizado, em que cada atributo representa um ou mais elementos existentes no módulo. As operações atribuídas a cada classe especificada se referem às ações associados às animações desenvolvidos no Director. A figura 19, apresenta a estrutura do jogo.

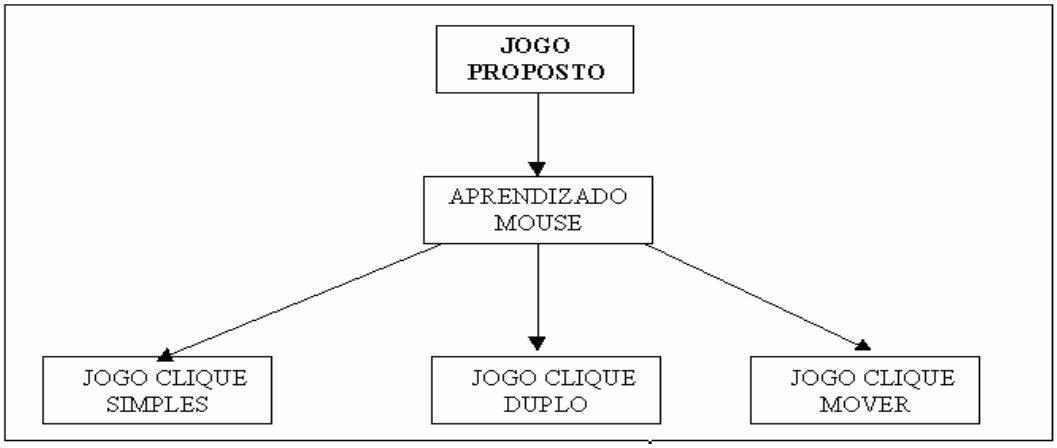


Figura 19 – Estrutura do Jogo

5.3.4.1 DIAGRAMAS DE CASOS DE USO

Nesta parte foram desenvolvidos dois diagramas de casos de uso, sendo um caso de uso na visão do aluno, e outro na visão do professor, conforme figura 20.

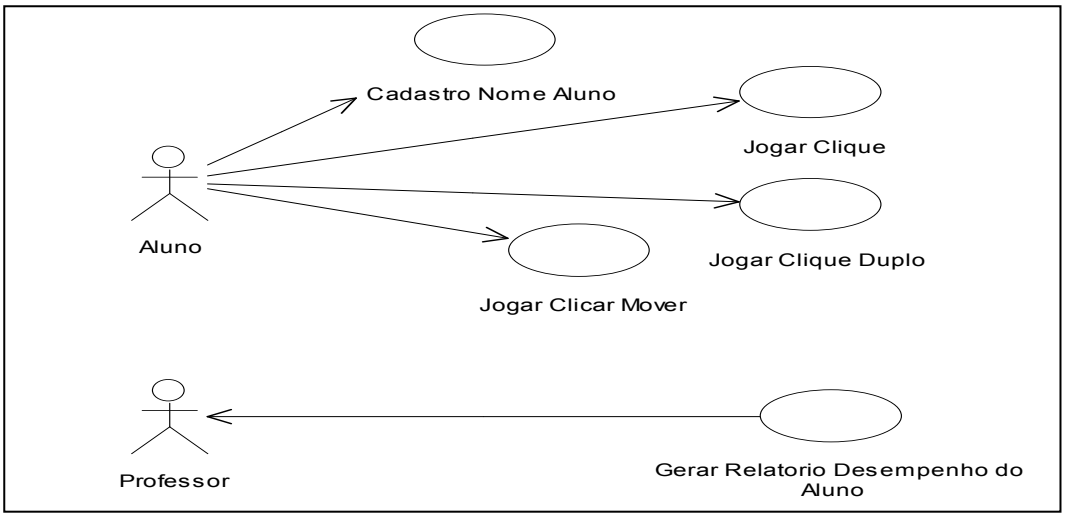


Figura 20 – Diagramas de Casos de Uso

Desta forma para um melhor entendimento, abaixo é apresentado um detalhamento de como se comporta as funções do diagrama de caso de uso conforme especificado anteriormente. No diagrama de caso de uso do aluno está presente, as seguintes funções:

a) cadastro nome aluno: o sistema apresentará uma tela onde o aluno deverá informar seu nome para poder iniciar o jogo. Depois de digitado o nome o aluno, deverá clicar no botão "iniciar", onde será direcionado ao menu do jogo que conterá as opções "clicar", "clique duplo" e "clicar mover".
b) jogar clique: neste caso de uso o sistema mostra um cenário com bolas de futebol que caem na tela durante um tempo pré-estabelecido. Assim o aluno com o *mouse* deverá usar o botão esquerdo, executando desta forma a função "clique". Quando o aluno clicar na bola, será adicionado um ponto ao placar de acertos, caso a bola não for clicada será adicionado um ponto ao placar de bolas perdidas. Desta forma as bolas irão cair durante um determinado tempo, caso o aluno não cumpra um percentual mínimo de acertos especificado pelo sistema, automaticamente, será sugerido para reiniciar a fase. Contudo se o aluno cumprir o percentual mínimo de acertos irá prosseguir para próxima fase, onde as bolas irão aumentar sua velocidade, prosseguindo desta forma até chegar à fase 3. No momento que o aluno chegar ao final

da fase 3, ele será encaminhado para o menu das opções, onde poderá escolher outro jogo.

c) jogo clique-duplo: neste caso de uso o sistema mostra um cenário com borboletas voadoras que se movem na tela durante um tempo pré-estabelecido. Assim podemos dizer que a funcionalidade desta etapa é idêntica ao jogo clique, mas seu diferencial está na utilização do *mouse*, pois este deverá usar a função "clique-duplo" para obter sucesso no jogo.
d) jogo clique-mover: neste caso de uso o sistema mostra um cenário de corrida de carrinhos que se deslocam na tela de forma aleatória durante um tempo predeterminado. Assim o aluno com o *mouse* deverá usar a função manter "pressionado" (clicar) e conseqüentemente "arrastar" (mover) para as casinhas de suas respectivas cores. No momento que o aluno pressionar sobre os carrinhos e levá-los até as casinhas determinadas, será adicionado um ponto ao placar, caso os carrinhos não forem direcionados para as casinhas corretas será adicionado um ponto ao placar de carrinhos perdidos. Mas os carrinhos irão cair durante um determinado tempo, caso o aluno não cumpra um percentual mínimo de acertos especificado pelo sistema e automaticamente será sugerido que ele reinicie a fase. Caso o aluno cumprir o percentual mínimo de acertos irá prosseguir para próxima fase. Quando o aluno chegar ao final da fase 3, ele será encaminhado para o menu das opções.

No diagrama de caso de uso do professor, somente será emitido um relatório com os dados estatísticos do desenvolvimento do aluno, nas fases que forem executadas.

5.3.4.2 DIAGRAMAS DE CLASSES

As atividades de projeto com o usuário visam definir formas de apoiar a realização da futura estrutura de trabalho no contexto de uso definido para o sistema, relacionando com suas tarefas e o seu ambiente técnico, organizacional e social. Mais especificamente, nesta etapa, é realizado um detalhamento da especificação do contexto de uso, processo no qual são definidas as diferentes características das interfaces.

Assim nesta etapa da abordagem, está presente a definição de um diagrama de trabalho a ser executado pelo usuário com o novo sistema. Este é descrito por um diagrama que apresenta classes de entidades para o início, para os processos e decisões, assim como para o

resultado esperado do trabalho. A descrição nestas classes de entidades contém nomes associados a objetos e atributos manipulados pelos usuários, além de verbos, associados às ações realizadas pelos usuários sobre estes objetos.

Na figura 21, é demonstrado o diagrama de classes do módulo treinamento com o mouse, cujas classes são descritas a seguir:

a) frame_inicia_Jogo: tem como finalidade iniciar todos os atributos do jogo, controlando as fases e o tempo de jogo;
b) frame_tipo_mouse: possui como característica gerar o controle dos eventos do mouse (clique, duplo-clique, arrastar-mover), como a de calcular a largura da tela para gerar de forma aleatória a posição dos sprites a serem lançados na tela;
c) tela_Jogo: função de incializar lista do jogo a ser praticado e controlar quantos acertos deverá ser realizado para prosseguir de fase, bem como de adicionar quantidades de acertos realizados no placar do jogo;
d) frame_clicar_mover: possui as funções de verificar encaixe do objetos, de forma a respeitar as regras do jogo;
e) frame_controlador_tempo: trabalha com as funções de tempo de duração das fases que será praticado no jogo;
f) frame_obtem_lista_desempenho: responsável por gerar a equivalência dos acertos e erros praticados pelo aluno no momento do jogo.

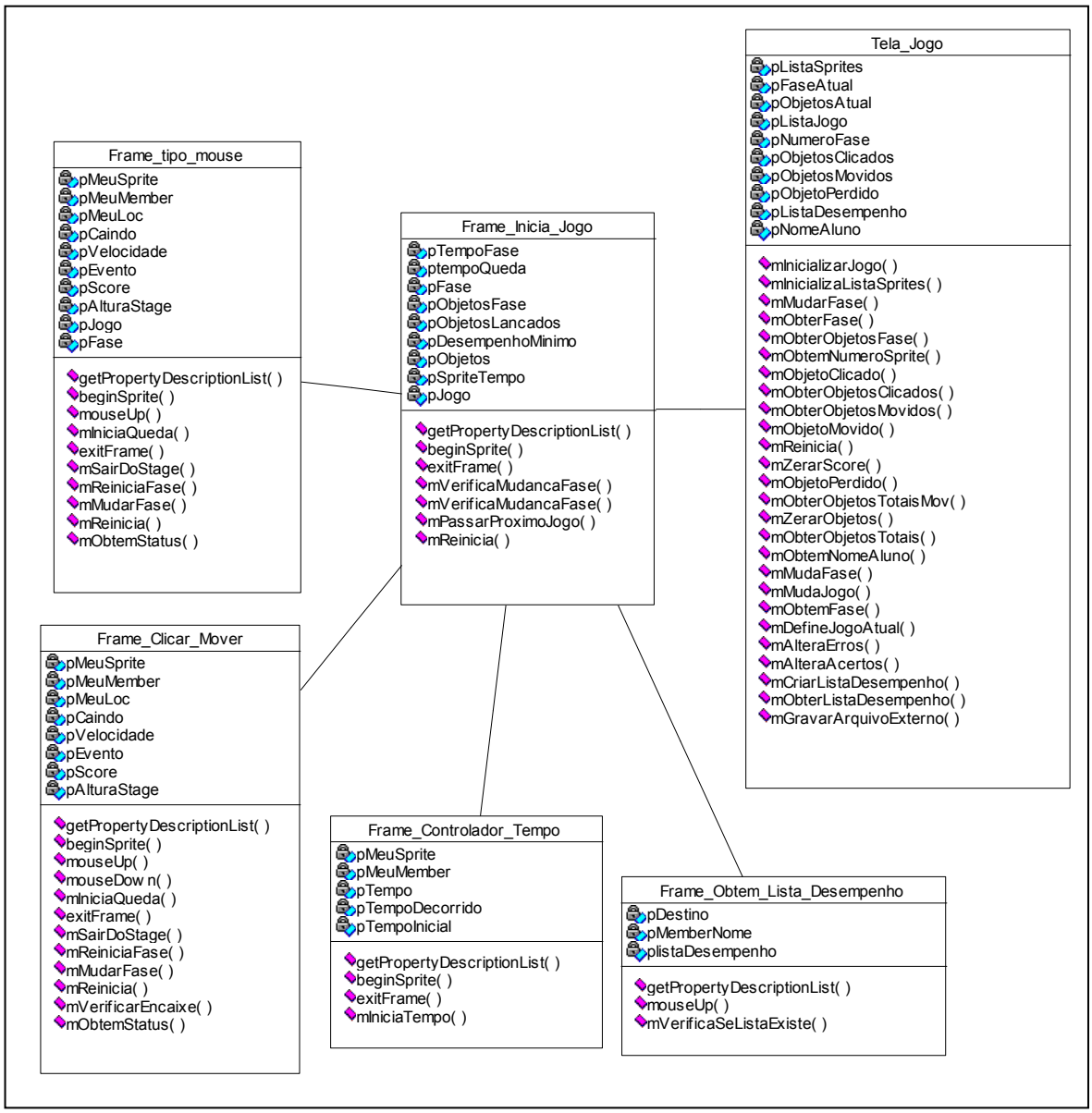


Figura 21 – Diagrama de Classes: Jogo *Mouse*

5.3.4.3 DIAGRAMA DE SEQÜÊNCIA

Continuando o processo de modelagem do *mouse* será considerado as análises das narrativas validadas de casos de uso, linha por linha, de modo a identificar quais ferramentas e materiais terão que ser fornecidos de forma a que o usuário seja capaz de realizar a interação (Confuso! Melhorar!!!). Neste processo, inicialmente são definidos os materiais necessários. Na seqüência, os casos de uso essenciais são revistos em busca das ferramentas e operações necessárias. Os materiais e as ferramentas identificados são listados no modelo de conteúdo do espaço de interação. Uma descrição de sua forma e comportamento pode ser adicionada, como forma de avançar sobre decisões do modelo de implementação. Estes já podem ser agrupados de modo a refletir como devem provavelmente ser usado.

Na figura 22, é demonstrado o diagrama de seqüência de interação: Jogo *Mouse.* Este diagrama tem a finalidade de exemplificar passo-a-passo o funcionamento do jogo, cujas funções são descritas a seguir:

a) quando iniciar o jogo, o aluno terá uma tela inicial onde será apresentados, todos os módulos que o jogo apresenta (clique, duplo-clique, arrastar-mover);
b) no momento que o aluno definir o jogo que deseja jogar, ele será direcionado a tela do campo jogo escolhido, e conseqüentemente a lista dos objetos (sprites) estará acionado, bem como o tempo de jogo e lista de desempenho do aluno;
c) depois de carregado as funções anteriores, neste momento iniciam os movimentos dos objetos na tela do jogo, como o reconhecimento da ação do *mouse* que deverá ser executado, avaliando desta forma suas regras;
d) no momento que o aluno estiver executando o jogo, as regras serão obedecidas de forma que, se o aluno não atingir um desempenho mínimo no tempo sugerido, ele será avisado que deverá refazer a fase. Se ele conseguir atingir o desempenho estabelecido pelo sistema, poderá passar para próxima fase, reiniciando os contadores e as regras de execução, ou seja, o jogo aumentará a velocidade de seus objetos a fim de dificultar a fase a ser jogado.

No diagrama de caso de uso do professor está presente as seguintes restrições:

a) na tela inicial do jogo o professor irá ao campo relatório e este irá verificar se o aluno mencionado pelo professor existe;
b) se existir o frame_obter_lista_desempenho será fornecido a lista de desempenho.

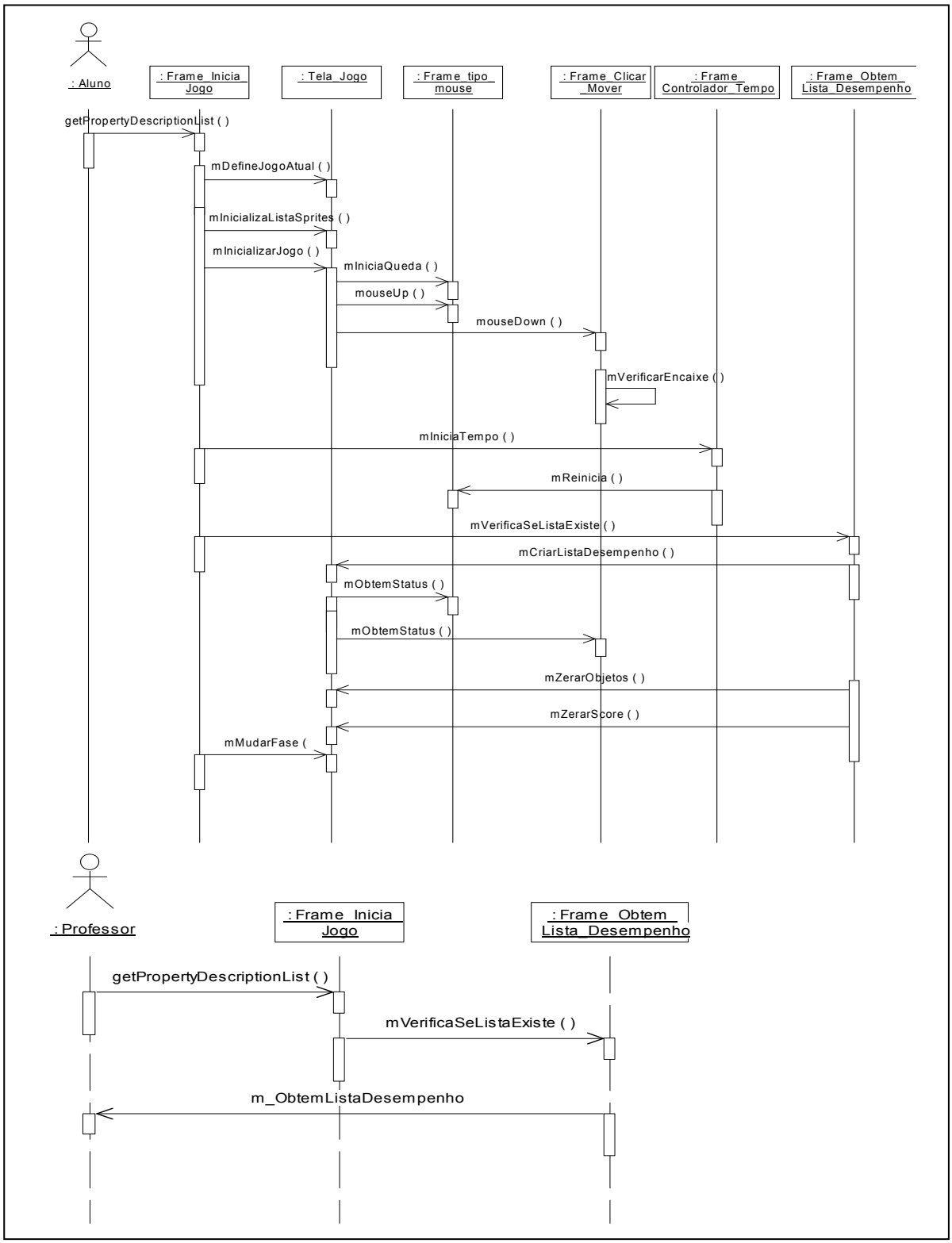


Figura 22 – Diagrama de Interação: Jogo *Mouse*

5.4 TUTORIAL DO SOFTWARE

O software aqui apresentado resultou na geração de um sistema na forma de jogo educativo para prática do uso do *mouse*..

5.4.1 FUNCIONAMENTO DO SOFTWARE

Primeiramente, quando iniciar o jogo, o usuário deverá digitar o seu nome para que o software possa fazer o monitoramento do seu desempenho, conforme figura 23.

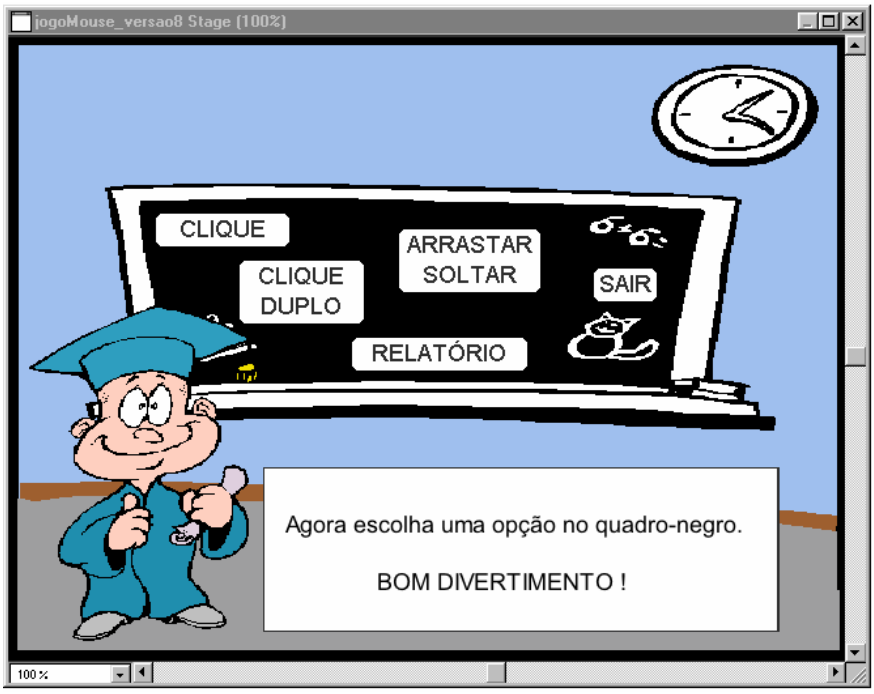


Figura 23 – Tela Abertura Jogo

Logo após ter digitado seu nome, na tela aparecerá uma nova janela, que contem o menu de opções. Assim o usuário poderá escolher qual opção ele vai ter desejo de jogar, conforme figura 24.

Figura 24 – Tela opções do Jogo

Quando o usuário escolher na tela de opções qual módulo irá praticar, abrirá uma nova janela com as instruções de como jogar. Conforme opções abaixo:

a) opção do jogo *clique*, conforme figura 25;

Figura 25 – Tela inicial da opção clique

b) opção do jogo clique-duplo, conforme expressa na figura 26;

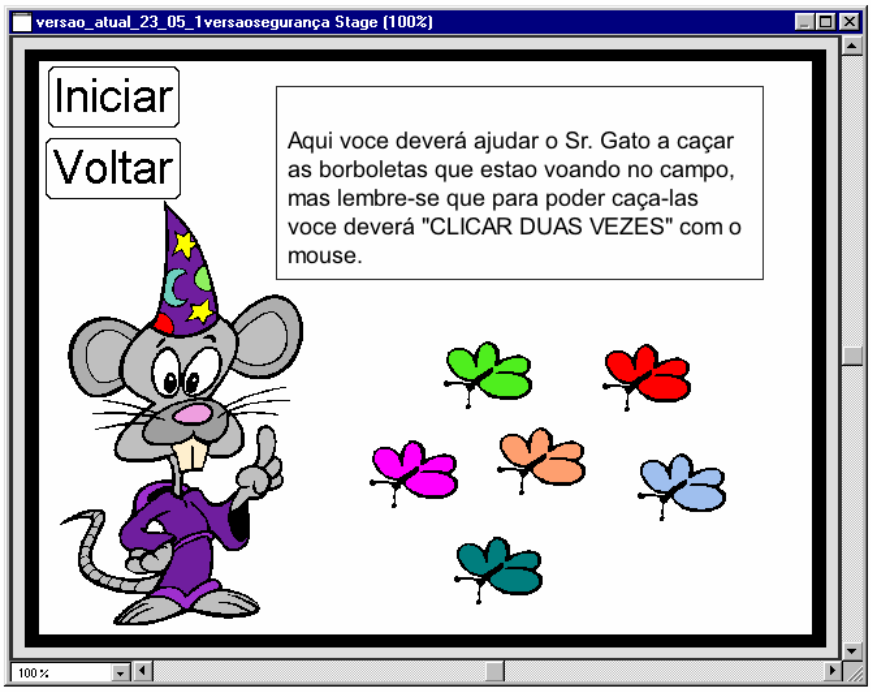


Figura 26 – Tela inicial da opção clique-duplo

c) opção do jogo clique mover, conforme expressa na figura 27;

Figura 27 – Tela inicial da opção clique mover

Todas as janelas terão na parte superior esquerda dois botões para que o usuário possa optar em iniciar ou voltar ao menu principal para escolher outro jogo.

5.4.2 ETAPAS DO JOGO

Abaixo será apresentado o funcionamento dos cenários, com os respectivos códigos de implementação.

Iniciaremos com a fase do clique, nesta fase a finalidade é acertar o maior número de bolas. Desta forma o usuário estará instruído para usar somente a função clique do *mouse*, conforme figura 28.

Figura 28 – Tela jogo clique

Para acomodar os objetos que estão na tela foi criada uma lista, conforme código de implementação no quadro 2, que tem como finalidade acomodar os números dos *sprites*, ou seja, os objetos que estão em movimento na tela do jogo.

```
on mInicializaListaSprites(me)
  -- Gera uma lista vazia para acomodar os números dos sprites
  pListaSprites = []

end mInicializaListaSprites


on mIncluiSprite(me,tSprite)

  pListaSprites.add(tSprite)
-- Adiciona sprites na lista
```

Quadro 2 – Adicionar objetos na lista

Para ter um controle do desempenho do usuário, é adicionado na tela do jogo um controle de acertos (*score*), para registrar cada clique realizado sobre os objetos que estiverem em movimento na tela, conforme mostra o código de implementação do quadro 3:

```
on mObjetoClicado(me, tValor)

  -- Responsável por contar quantos objetos foram clicados na
tela_jogo.
  pObjetosClicados = pObjetosClicados + tValor
  member("score").text = String(pObjetosClicados)
```

Quadro 3 – Score

Para que os objetos pudessem desaparecer da tela a cada ação gerada pelo mouse, foi desenvolvida a seguinte estrutura de implementação, demonstrado no quadro 4. Desta forma pode-se verificar que conforme os objetos forem passando pela tela do jogo, eles retornam na seqüência, obedecendo a estrutura de uma lista.

```
on mSairDoStage(me)

  -- Posicionando o sprite fora do stage (tela_jogo)
  sprite(pMeuSprite).loc = point(1000,1000)
  updateStage

   -- Reposicionando o sprite na lista de sprites
  mIncluiSprite(script "tela_jogo",pMeuSprite)

  pCaindo = FALSE
```

Quadro 4 – Posicionar objetos na lista

Para acompanhar o desenvolvimento de cada usuário, foi criado um índice de desempenho que calcula o aproveitamento do aluno durante a execução do jogo, conforme estrutura desenvolvida no quadro 5.

```
on mVerificaMudancaFase(me)
  tObjetosClicados = mObterObjetosClicados(script "tela_jogo")
  tPercentual = float(tObjetosClicados) /pObjetosFase*100
  if tPercentual > pDesempenhoMinimo then
    mMudarFase(script "tela_jogo")
    pFase = mObterFase(script "tela_jogo")
    if integer(pFase) then
      pObjetosFase = mObterObjetosFase(script "tela_jogo")
      sendAllSprites(#mMudarFase)
      alert " Parabéns!Prossiga para Próxima Fase!"
      sendAllSprites(#mIniciaTempo)
    else
      alert "Parabéns! Você chegou ao Final do Jogo."
      alert "Agora Voce Voltará ao Menu Inicial."
      mPassarProximoJogo(me)
    end if
 else
pObjetosFase = mObterObjetosFase(script "tela_jogo")
mZerarObjetos(script "tela_jogo")
sendAllSprites(#mReiniciaFase)
alert "Voce Deverá Repetir Novamente a Fase"
sendAllSprites(#mIniciaTempo)
pObjetos = 0
```

Quadro 5 – Mudança de fase

Quando o aluno passar por todas as fases de cada módulo do *mouse*, automaticamente irá retornar ao menu, para escolher outro módulo, caso queira praticar. No quadro 6, será demonstrado o controle de acertos que o usuário deverá fazer para passar de fase e retornar ao menu inicial.

```
on mInicializarJogo(me)

   --Inicializa lista do jogo
   --Controla quantos acertos, para passar de fase
  pListaJogo = [#fase1:5,#fase2:5,#fase3:5]

  pNumeroFase = 1

end mInicializaJogo
```

Quadro 6 – Lista fases e quantidades de objetos

Para os jogos de clique-duplo do mouse, as funções de implementação são iguais, mas seu diferencial está no cenário, como mostrada na figura 29.

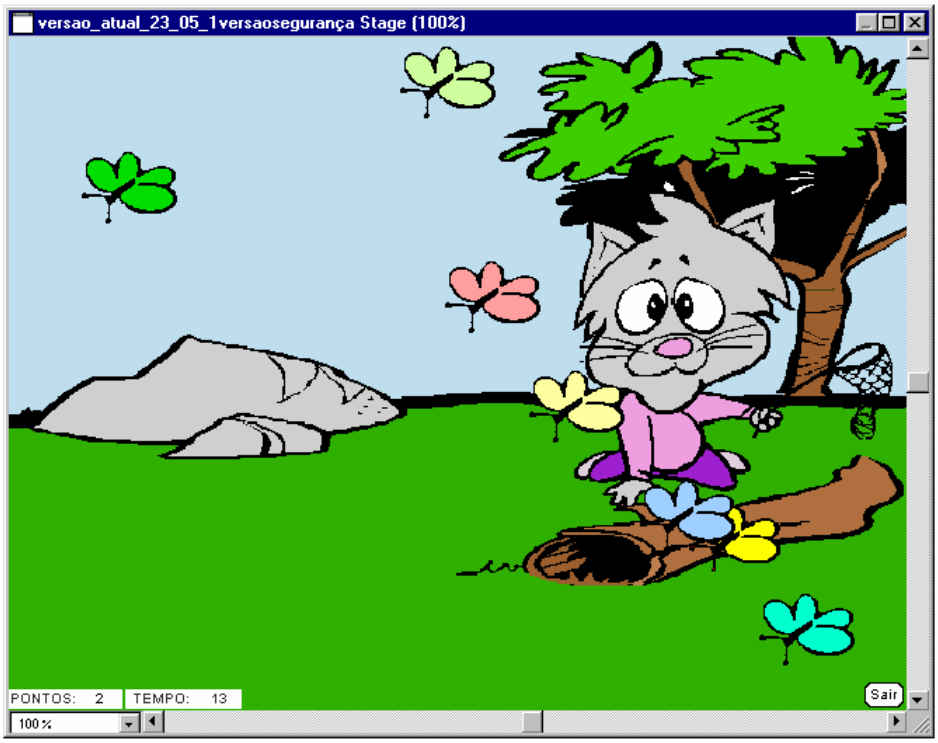


Figura 29 – Tela jogo clique-duplo

No quadro 7 é mostrado como foi realizada a criação da lista de eventos do *mouse*.

```
on getPropertyDescriptionList(me)

  pPropertyList = [:]

  pPropertyList[#pEvento] = [\
                            #comment:"Objeto Evento?",\
                            #format:#symbol,\

#range:[#mouseUp,#duploClique,#mouseDown],\
                            #default:#mouseUp\
                            ]
pPropertyList[#pVelocidade] = [\
                                #comment:"Velocidade Queda?",\
                                #format:#integer,\
                                #range:[#min:3,#max:8],\
                                #default:3\
                               ]
    RETURN pPropertyList
```

Quadro 7 – Criação lista de eventos do *mouse*

Para o jogo de clique mover, foram feitas algumas alterações na tela e nos objetos. Os objetos agora devem ser clicados e movidos para os locais determinados, conforme figura 30.

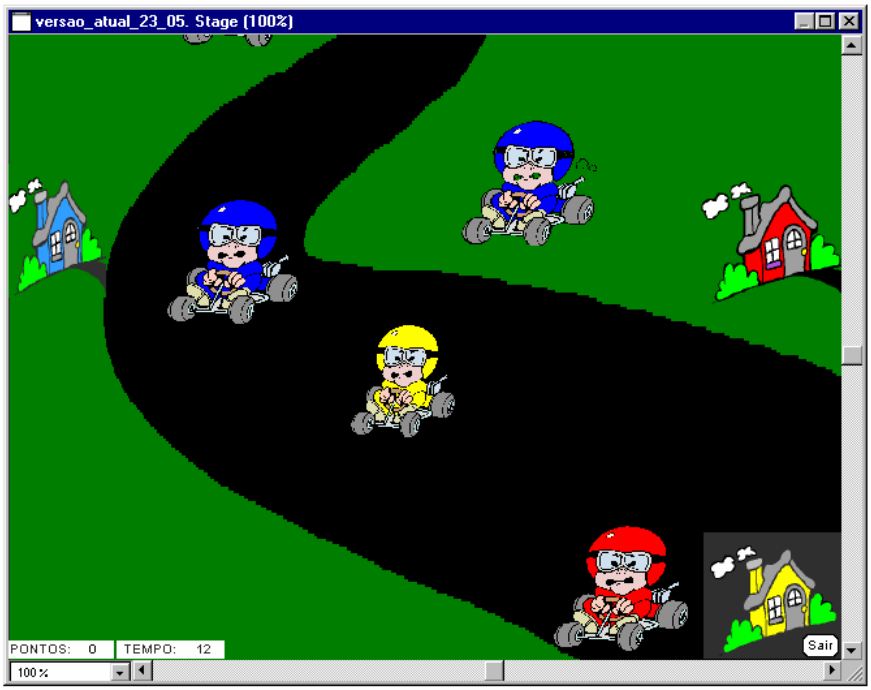


Figura 30 – Tela do jogo clique mover

Para que os objetos desaparecessem da tela ao serem colocados nos seus respectivos lugares, desenvolveu-se a seguinte implementação, conforme quadro 8.

```
on mVerificarEncaixe(me)

  -- Para encaixar os carrinhos nas determinadas casa, foi
adicionado uma lista onde a casa recebe o sprite da lista.

  tCasa = sprite(pSpriteRecipiente).rect
  tLoc = sprite(pMeuSprite).loc

  pMovendo = FALSE

  -- Se o objeto está dentro da caixa correta o objeto sai da
tela, senão o objeto continua caindo.

  if tLoc.inside(tCasa) then

    mSairDoStage(me)

    mObjetoClicado(script "tela_jogo",1)

  else

    pCaindo = TRUE

  end if
```

Quadro 8 – Verificando encaixe de objetos

6 CONCLUSÕES

O presente trabalho mostra que um sistema gerado através de conceitos de Engenharia da Usabilidade, tende a trazer benefícios aos usuários que fazem utilização do mesmo, fazendo com que os programadores evitem perdas desnecessárias com correções de programas apontados pelos usuários em relação às interfaces utilizadas.

Como objetivo, este trabalho apresentou o desenvolvimento de um software educacional, na forma de jogo multimídia, para auxiliar crianças que contenham alguma deficiência física ou mental, no uso e aplicação do *mouse*. Em seu formato atual, o software desenvolvido permite ao usuário em potencial a utilização do *mouse*, na opção clique, clique-duplo e clique-mover, além dos recursos básicos de um navegador.

Com isso, espera-se que este trabalho traga como contribuição aos leitores, técnicas e métodos que melhorem o processo de criação de interfaces, que conforme apresentado por Cybis (2002, p. 2), este método está presente como um fator importante nos requisitos de programação de software.

Observa-se que, com relação às funções previstas, o presente trabalho alcançou os objetivos traçados, apresentando um software educacional interativo, que utiliza conceitos e técnicas de Engenharia da Usabilidade para sua construção.

6.1 EXTENSÕES

Como sugestão para trabalhos futuros, indicamos a adição de trabalhos que explorem todo o potencial que a ferramenta Director proporciona.

Em relação às técnicas de Engenharia de Usabilidade, a sugestão é que futuros trabalhos formulem modelos descritivos e intuitivos, compostos por exemplos, onde os usuários possam identificar de maneira eficaz inconsistências que precisam ser reformuladas.

Outra sugestão é, baseado na seqüência da especificação desenvolvida neste trabalho, criar tutoriais que possam ser utilizados como ferramentas para outras áreas do conhecimento, abordando técnicas de Engenharia de Usabilidade.

REFERÊNCIAS BIBLIOGRÁFICAS

ABT, Clark C. **Jogos simulados:** estratégias e tomada de decisão. Tradução Alexandre Lissovsk. Rio de Janeiro, RJ: Olympio, 1974.

ANGIOLETTI, Jocir Leandro. **Protótipo de um software educacional para educação primária usando recursos de multimídia**. 1995. 69 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

BATTAIOLA, André Luiz. **Jogos por computador** - Histórico, Relevância Tecnológica e Mercadológica, Tendências e Técnicas de Implementação. In: Jornada de Atualização em Informática, 19, 2000, Curitiba. **Anais...** Curitiba: PUCPR, 2000.

BARROS, Pablo. **UML**: Linguagem de Modelagem Unificada em Português. São José, jan. 2000. Disponível em: <http://www.eng.uerj.br/~araujo/disciplinas/uml/tutorial/page02.html> . Acesso em: 07 maio 2004.

BIZZOTTO, Carlos E. Negrão. **Director 8:** rápido e fácil. São Paulo: Makron Books, 2000.

BIZZOTTO, Carlos E. Negrão. **Aprendiz:** ambiente extensível para o aprendizado distribuído. 2003. 123 f. Tese (Doutorado em Engenharia de Produção) – Departamento de Engenharia de Produção e Sistemas, Universidade Federal de Santa Catarina, Florianópolis.

CAMPOS, Fernanda C. A. **Hipermídia na Educação:** Paradigmas e Avaliação da Qualidade. Tese de Mestrado. COPPE/SISTEMAS - UFRJ. Agosto. 1994.

CPV. **Clube de Pilotos Virtuais**. O portal da aviação, fev 2000. Apresenta informações sobre aficionados por aviação em geral (real e virtual). Disponível em: <http://www.clubedepilotos.com/>. Acesso em : 25 maio 2004.

CYBIS, Walter de Abreu. **Engenharia de usabilidade:** uma abordagem ergonômica, Florianopolis, maio 2003. Disponível em: <http://www.labiutil.inf.ufsc.br/>. Acesso em: 18 maio 2004.

CYBIS, Walter de Abreu. **A identificação dos objetos das interfaces homem-máquina e de seus atributos ergonômicos.** 1994. Tese (Doutorado em Engenharia de Produção) – Departamento de Engenharia de Produção e Sistemas, Universidade Federal de Santa Catarina, Florianópolis.

DALFOVO, Regiani. **Protótipo de software para o ensino de introdução a microinformática**. 1997. 65 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

DBSERVER, Ferramentas Case. **Parceria Rational**. Assesoria em Sistemas de Informação, set. 1999. Disponível: <www.dbserver.com.br>. Acesso em: 30 maio 2004.

DEBONI, José Eduardo Zindel. **Breve Introdução aos Diagramas da UML**. Voxxel Consultoria de Sistemas, São Bernardo do Campo, ?.2000. Disponível em: < http://www.voxxel.com.br/deboni/IntroUML/#bibliografia>. Acesso em: 8 maio 2004.

FAGUNDES, Léa da Cruz. **Informática na Escola**. Tecnologia Educacional, Rio de Janeiro, v.21 (107), p. 79-84, jul./ago. 1992.

FERREIRA, Josemar Dias. **Multimídia para programadores e analistas**. Rio de Janeiro: Infobook, 1995.

FOWLER, Martin; SCOTT, Kendall. **UML essencial:** um breve guia para a linguagem-padrão de modelagem de objetos. Porto Alegre: Bookman, 2000.

Furlan, José Davi. **Modelagem de objetos através da UML:** the unified modeling language. São Paulo: Makron Books, 1998.

GAGNÉ, Robert M. **Princípios essenciais da aprendizagem para o ensino**. Porto Alegre: Globo, 1980.

GONZALEZ, Julio Francisco Planella. **Director 8.5:** criando aplicativos multimídia. São Paulo: Berkeley Brasil, 2001.

GREEN, Thomas J. **Macromedia Director 8 & Lingo** – guia rápido para desenvolvimento na Web. Rio de Janeiro: Ciência Moderna Ltda., 2001.

GROSS, Phil. **Director 8 and Lingo**. Berkeley: Macromedia, 2000.

HANNA, M. **Attention to process ups software quality**. Software Magazine Dec. 1993 vl 3 n18 43-47.

KLEIS, Margarete Lazzaris. **Validação e priorização de critérios e recomendações para projeto e avaliação de ambientes virtuais de educação a distância.** 2001. 192 f. Dissertação (Mestre em Engenharia de Produção) – Departamento de Ergonomia, Universidade Federal de Santa Catarina, Florianópolis.

KOSLOSKY, Ivana Therezinha Gogolevsky. Metodologia para criação de jogos a serem utilizados na área de educação ambiental. 2000. 132 f. Dissertação (Mestre em Engenharia de Produção) – Departamento de Ergonimia, Universidade Federal de Santa Catarina, Florianópolis.

LABIUTIL, Laboratório de Utilizabilidade. **Engenharia ergônomica de usabilidade de interfaces, humano-computador,** Florianopolis, ? 1993. Disponível em: < http://www.labiutil.inf.ufsc.br/ergolist/check.htm>. Acesso em: 1 jun. 2004.

LAVILLE, Antoine. **Ergonomia.** Tradução Márcia Maria Neves Teixeira. São Paulo: Universidade de São Paulo, 1977. 101 p.

MARTINS, Jane Gonçalves. **Realidade virtual através de jogos na educação**, Santa Catarina**,** [2002?]. Disponível em: **<**http://www.intercom.org.br/papers/xxii-ci/gt13/13m02.PDF>. Acesso em: 10 fev. 2004.

MEDEIROS, Marco Aurélio. ISO 9241: uma proposta de utilização da norma para avaliação do grau de satisfação de usuários de software. 1999. 146 f. Dissertação (Mestre em Engenharia de Produção) – Departamento de Ergonimia, Universidade Federal de Santa Catarina, Florianópolis.

MEC. Ministério da Educação. Desenvolvido pelo Departamento de Informática de educação à distância, 2001. Apresenta textos sobre o Programa nacional de informática na educação – Proinfo. Disponível em: <http://www.proinfo.gov.br>. Acesso em: 10 maio 2004.

MIELKE, Fernando Luiz. **Ensino assistido por computador:** algumas considerações teoricas da ergonomia e da inteligencia artificial num ambiente hipertexto. 1991. 112 f. Dissertação (Mestre em Engenharia de Produção) – Departamento de Engenharia de Produção e Sistemas, Universidade Federal de Santa Catarina, Florianópolis.

MONTEIRO, Eduardo Bastos; GOMES, Flávia Rezende dos Santos. **Informática e educação.** São Paulo: Tecnologia Educacional, 1993.

MORAES, Anamaria de; SOARES, Marcelo M. **Ergonomia no Brasil e no Mundo:** um quadro, uma fotografia. Rio de Janeiro: Univerta ABERGO – Associação Brasileira de Ergonomia, 1989.

MORO, Mirella Moura. **Engenharia de software**, Porto Alegre, maio 2000. Disponível em: <http://www.inf.ufrgs.br/~mirella/cmp102/>. Acesso em: 18 mar. 2004.

ODEBRECHT, Clarisse; GONTIJO, Leila Amaral. **Sistemas ergonômicos:** arte, ciência ou bom-senso?. Dynamis Revista Tecno-Cientifica. Editora FURB. Blumenau, 1993.

OLIVEIRA, Verlane Puel de. **Plano de validação para o modelo de OIAe.** 1995. Dissertação (Mestre em Engenharia de Produção) – Departamento de Engenharia de Produção e Sistemas, Universidade Federal de Santa Catarina, Florianópolis.

PEIXOTO, Ana Sofia Brito, OLIVEIRA, Rosa dos Anjos. **Terminologia do ensino por computador:** abordagem socioterminológica, Brasília, ?1995. Disponível em: <http://www.ibict.br/cionline/240395/24039514.pdf>. Acesso em: 01 abr. 2004.

PRESSMAN, Roger. **Software Engeneering: a Practioner's Approach**. Third Ediction. McGraw Hill International Editions. 1992.

RATIONAL, Software Corporation. **Rational Rose.** Assesoria em Sistemas de Informação, set. 1999. Disponivel em: <www.rational.com>. Acesso em: 25 maio 2004.

ROCHA, Ana Regina, CAMPOS, Fernanda, CAMPOS, Gilda. **Dez etapas para o desenvolvimento de software educacional do tipo hipermídia**, Rio de Janeiro, ? 1999. Disponível em: <http://www.timaster.com.br/revista/colunistas/ler_colunas_emp.asp?cod=331>. Acesso em: 10 abr. 2004.

ROSENZWEIG, Gary. **Macromedia Director 7.** Indiana: Division of Macmillan Computer Publishing, 1999.

SMALL, Peter. **Lingo sorcery**: the magic of lists, objects and intelligent agents. Chichester: John Wiley, 1999.

SILVEIRA, Sidnei Renato. **Estudo de uma Ferramenta de Autoria Multimídia para a Elaboração de Jogos Educativos**, Porto Alegre, 1998. Disponível em: <http://www.inf.ufrgs.br/pos/SemanaAcademica/Semana98/sidnei.html > Acesso em: 12 maio 2004.

STEIN, Sandra Elisa. **Ergonomia:** auxiliando na interface homem-máquina através da utilização de ferramentas. 1994. 104 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

TREIS, David Jones. **Utilização de técnicas de usabilidade no desenvolvimento de um software de controle de pedidos.** 2003. 54 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

V-CAD. **Simplifique seu AutoCAD**. Software feito para simplificar a operação do AutoCAD. Apresenta tutoriais sobre como utilizar a ferramenta autoCAD, 2000. Disponível em: < http://www.vcad.arq.br/index.html>. Acesso em: 15 maio 2004.

VALENTE, José Armando. **Computadores e conhecimento**: repensando a educação. Campinas: Gráfica Central da UNICAMP, 1993.

WENDT, Itatiana Bárbara Novak. **Software multimídia para auxiliar no processo de ensino-aprendizagem da informática a pessoas da terceira idade.** 2002. 75 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

ZACHARIAS, Vera Lúcia Câmara. **Centro de Referencia Educacional.** Apresenta conteúdos na área educação, 1998. Disponível em: <http://www.centrorefeducacional.com.br>. Acesso em: 11 mar. 2004.