Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística
Bacharelado em Sistemas de Informação

Enéias Silva
Michelly Nátali Anacleto

PROJETO E DESENVIMENTO DE SOFTWARE E HARDWARE ADAPTADO A PESSOAS COM DIFICULDADES MOTORAS E COGNITIVO PRESERVADO

Proposta completa para o trabalho de conclusão do curso de Sistemas de Informação da Universidade Federal de Santa Catarina

Edla Maria Faust Ramos (orientadora)
Lisiane Capanema Silva Bonatelli (co-orientadora)

Florianópolis, Julho de 2006.

“Acredito que o deficiente seja vítima do destino,
mas não posso admitir que seja vítima da indiferença”.
John Fitzgerald Kennedy

Índice Analítico

1. Introdução 7

1.1. Problemática 8

1.2. Objetivos 9

1.2.1. Objetivo Geral 9

1.2.2. Objetivos Específicos 9

1.3. Motivação 11

1.4. Metodologia 12

1.5. Resultados Esperados 13

2. Embasamento Teórico 14

2.1. Deficiências Motoras 14

2.1.1. Paralisia Cerebral 14

2.1.2. Ataque vascular cerebral 15

2.1.3. Lesão Medular 16

2.2. Acessibilidade 17

2.2.1. Acessibilidade na tecnologia 19

2.3. Sistema de Comunicação Alternativa 19

2.3.1. Sistema Bliss 21

2.3.2. PCS – Picture Communication Symbol – Símbolos de Comunicação Pictórica 22

3. Estudo preliminar da tecnologia na Comunicação Alternativa 24

3.1. Uso do Software na Comunicação Alternativa 24

3.1.1. LMBrain 24

3.1.2. ImagoAnaVox 25

3.1.3. Comunique 26

3.1.4. Escrita com Símbolos 27

3.1.5. GRID 29

3.1.6. InVento 31

3.1.7. BoardMarker 33

3.1.8. FALAS - Ferramenta ALternativa de Aquisição Simbólica 35

3.2. Uso do Hardware na Comunicação Alternativa 37

3.2.1. Máscaras de teclado 37

3.2.2. Acionadores de varreduras alternativos para uso em softwares especiais 38

3.2.3. Acionamento pelo microfone por meio de brinquedos de pressão 40
3.2.4. Adaptações do teclado com sobreposição de teclas 40

4. Solução proposta 42

4.1. Projeto de software 42
4.1.1. Introdução 42
4.1.2. Especificação de Requisitos 47
4.1.3. Modelo de Casos de Uso 52
4.1.4. Modelo Conceitual 54
4.1.5. Modelo de Classes 55

4.2. Adaptação de Hardware 57

4.3. Utilização do Jardim das Palavras Secretas 60
4.3.1. Cadastros e Configurações 63
4.3.2. Relatórios 67

5. Conclusão 69

6. Trabalhos Futuros 70

7. Referências Bibliográficas 71

8. Anexo 1 – Código Fonte 76

Índice de figuras

Figura 1. *Exemplo de Pranchas de comunicação* *20*

Figura 2. *Exemplo de símbolos do Sistema Bliss* *21*

Figura 3. *PCS em Livro* *23*

Figura 4. *Tela do Programa ImagAnaVox* *25*

Figura 5. *Alguns símbolos do software Escrita Com Símbolos* *28*

Figura 6. *Interface do software GRID* *30*

Figura 7. *Grid em PocketPC* *31*

Figura 8. *Imagens utilizadas no InVento* *32*

Figura 9. *Alguns símbolos do InVento* *32*

Figura 10. *Símbolos PCS* *35*

Figura 11. *Arquitetura do FALAS* *36*

Figura 12. *Figura 3.1: Máscara de teclado encaixada no mesmo* *37*

Figura 13. *Figura 3.2: Adaptação de mouse* *39*

Figura 14. *Acionamento por meio de microfone* *40*

Figura 15. *Teclado adaptado por Edson Edir Dick* *41*

Figura 16. *Diagrama de atores envolvidos* *45*

Figura 17. *Diagrama de Atividades Principal* *46*

Figura 18. *Diagrama de Requisitos Funcionais* *48*

Figura 19. *Diagrama de Requisitos Não-Funcionais* *49*

Figura 20. *Diagrama de Regras de Negócios* *50*

Figura 21. *Diagrama de Regras de Usabilidade* *52*

Figura 22. *Diagrama de Casos de Uso* *52*

Figura 23. *Diagrama de Modelo Conceitual* *55*

Figura 24. *Diagrama de Classes* *56*

Figura 25. *Teclado em fase de adaptação* *57*

Figura 26. *Tampas cegas de instalações elétricas* *58*

Figura 27. *Visão inferior das teclas do teclado adaptado* *59*

Figura 28. *Teclado adaptado* *59*

Figura 29. *Configuração do teclado* *60*

Figura 30. *Tela de escolha de facilitador, usuário e categoria* *61*

Figura 31. *Tela com alfabeto e palavra secreta para adivinhação* *62*

Figura 32. *Tela de parabenização pela palavra acertada* *62*

Figura 33. *Tela que demonstra a palavra secreta não acertada* *63*

Figura 34. *Gerenciamento de Facilitadores* *64*

Figura 35. *Cadastro de Usuários* *65*

Figura 36. *Tela do cadastro de palavras e categorias* *66*

Figura 37. *Configuração do Teclado* *67*

Figura 38. *Relatório por usuário* *68*

1. Introdução

Os computadores têm se tornado cada vez mais acessíveis a todas as classes sociais. A questão do computador para todos ainda é utópica, pois as interfaces que atualmente conhecemos estão longe de corresponder às necessidades de todos os cidadãos. O teclado, o mouse e a própria tela são alguns dos pecados ergonômicos que muitos têm que ultrapassar.

Sendo assim, este trabalho tem a preocupação e a intenção de contribuir para o rompimento de algumas dessas barreiras, trazendo a adaptação e a melhoria desses recursos tecnológicos para que possam vir a ser utilizados pelas pessoas portadoras de necessidades especiais. Por convenção sempre que for citado o termo portador de deficiência ou portador de necessidades especiais, estaremos fazendo alusão a indivíduos com limitações motoras, porém com sua capacidade cognitiva preservada, ou seja, pessoas que por algum motivo tiveram sua integridade física prejudicada por acidentes, síndromes, entre outras, porém mantiveram as regiões cerebrais responsáveis pela aprendizagem preservadas. .

Um grande número de pessoas com necessidades especiais poderia encontrar nas tecnologias da informação, importantes alternativas e oportunidades para a aprendizagem, para o acesso à informação, ao lazer e ao exercício de uma atividade ou vocação. A acessibilidade a estas tecnologias é, portanto, um fator decisivo para sua qualidade de vida.

Acreditamos que as dificuldades de utilização do computador por pessoas portadoras de deficiências são originadas também pela inadequação das interfaces - hardware e software, e não somente pelas limitações dessas pessoas.

Para conceber um software acessível para pessoas com deficiências é necessário conhecer as suas dificuldades e buscar as soluções que permitam contorná-las. Inclui-se nessas soluções o conhecimento de recursos de acessibilidade, que podem cumprir diferentes categorias de funções, sendo classificados da seguinte maneira: adaptações físicas ou órteses, adaptações de hardware e softwares especiais de acessibilidade.

Adaptações físicas ou órteses são todos os aparelhos ou adaptações fixadas e utilizadas no corpo da pessoa e que assim facilitam a sua interação com o computador (ex: suporte de cabeça, ponteira para digitação, etc.).

Adaptações de hardware são todos os aparelhos ou adaptações presentes nos componentes físicos do computador, nos periféricos, ou mesmo, quando os próprios periféricos, em suas concepções e construção, são especiais e adaptados (ex: colocação de máscaras no teclado, mouse de preensão, etc.).

Por fim, softwares especiais de acessibilidade são os componentes lógicos das TIC´s[1] quando construídos como Tecnologia Assistiva[2], ou seja, são os programas especiais de computador que possibilitam ou facilitam a interação do aluno portador de deficiência com a máquina (ex: simuladores de teclado e mouse, comandos adaptados para controle através do sopro, etc.).

1.1. Problemática

Dentre as dificuldades enfrentadas pelos portadores de deficiência na utilização das tecnologias, será abordada neste trabalho a adaptação de recursos tecnológicos computacionais que favoreçam os processos educacionais de nosso publico alvo, ou seja, as pessoas com cognitivo preservado e limitação de movimentos, servindo dessa forma, como uma ferramenta auxiliar para o educador.

Mais especificamente a proposta é adaptar uma atividade lúdica, o "Jogo da Forca", velho jogo de adivinhações de palavras, que serve como exercício de memorização e ampliação de vocabulário.

Espera-se que a adaptação desse jogo não necessariamente venha a ser utilizada apenas em escolas, mas que possa servir também como uma nova alternativa de entretenimento e fator de interação entre os familiares e os portadores de limitações motoras.

---

[1] TIC´s - Tecnologias da Informação e Comunicação

[2] A Tecnologia é considerada *Assistiva* quando é usada para auxiliar no desempenho funcional, reduzindo incapacidades para a realização de atividades da vida diária e da vida prática, nos diversos domínios do cotidiano.

1.2. Objetivos

1.2.1. Objetivo Geral

Através de pesquisas pedagógicas realizadas, observou-se a carência na área de necessidades especiais, principalmente no que diz respeito ao auxílio no desenvolvimento psicomotor das pessoas portadoras de deficiências.

Este trabalho vem ao encontro desta carência, desenvolvendo um software, fruto da adaptação e remodelagem de um jogo bastante conhecido no Brasil e em todo o mundo, o Jogo da Forca. Esse jogo constitui-se de atividade lúdica que possibilita o desenvolvimento da linguagem escrita. Além disso, a adaptação prevista deve permitir que a fixação dos símbolos da comunicação alternativa sendo eles criações próprias ou padrões já existentes no mercado como o sistema de comunicação alternativa Bliss, por exemplo, cuja definição será detalhada mais adiante, bem como, a relação direta existente entre o desenho do símbolo e a palavra escrita que o representa.

Para a utilização deste software faz-se necessária uma interface adequada ao usuário, buscando perfeita interação e amenizando as distâncias entre o real e a tecnologia, por muitas vezes inalcançável para estas pessoas. Para tal, será adaptado um teclado possivelmente com a utilização de grandes teclas que servirão de guia na utilização do programa.

Desta forma, o trabalho visa projetar e desenvolver este software e a adaptação de hardware necessária, baseando-se em estudos pedagógicos direcionados e disponibilizá-lo a sociedade, a fim de contribuir à vida dessas pessoas tão especiais.

1.2.2. Objetivos Específicos

Como objetivos específicos, que detalham os alcances deste trabalho podem-se citar:

- Elaboração de um software, envolvendo análise, projeto e desenvolvimento, baseado numa atividade lúdica, o jogo da forca, buscando

entreter e ao mesmo tempo desenvolver as aptidões intelectuais como a memória, a expansão do vocabulário e pensamento racional e lógico de pessoas com problemas psicomotores.

- Adaptação de um teclado, tornando-o o principal periférico para entrada de dados adequados às limitações das pessoas portadoras de deficiências motoras, com seis grandes teclas que serão totalmente gerenciadas pelo software de acordo com os objetivos do jogo.

- Empacotamento deste software, contendo seu código e toda documentação UML[3], a fim de disponibilizá-lo à comunidade de software livre do Brasil, com o objetivo de propagar este trabalho, ajudando pessoas de outras localizações e abrindo espaço para a adaptação e melhorias que serão realizadas pela comunidade de software.

[3] UML – Unified Modelling Language: Linguagem padronizada para documentação de software mundialmente utilizada

1.3. Motivação

Uma busca constante das sociedades humanas é a melhoria na qualidade de vida, incluindo melhores serviços de saúde, estudos e lazer. No mundo dos cidadãos com algum tipo de deficiência, estas buscas vêm acompanhadas de desafios ainda maiores.

Como o número de pessoas portadoras de necessidades especiais é pequeno em relação à população total, os recursos encaminhados são poucos e os produtos disponíveis na sua maioria devem ser importados, desta forma, aumentando ainda mais os custos.

A idéia deste projeto, não é reinventar a roda, mas sim, fazer um melhor uso dela, em outras palavras, serão utilizadas tecnologias já existentes, adaptando-as, como alternativa barata para nosso público alvo.

Como nesta sociedade de desigualdades o fator financeiro é um dos mais discrepantes e tudo é movido pelo capital, assim quanto mais caro, menos gente pode ter acesso, estaremos disponibilizando às pessoas com problemas e sem poder aquisitivo, um produto que busca aumentar sua qualidade de vida.

1.4. Metodologia

Utilizando-se de situações práticas do dia-a-dia do portador de necessidades especiais e pesquisando os recursos já disponíveis no mercado optamos pela criação de uma solução de baixo custo, visto que as poucas disponíveis são onerosas e não estão hoje acessíveis a maioria deste público.

Neste projeto serão utilizados basicamente dois tipos de adaptações.

A primeira será no nível de hardware onde um teclado deverá sofrer modificações, adaptando o modelo de teclado existente em um periférico de entrada de dados específico para os movimentos das pessoas portadoras de problemas motores.

A segunda adaptação dar-se-á no nível do software, onde será adaptado um sistema simples e bastante antigo, o jogo da forca, trazendo para a realidade de nosso público-alvo, utilizando da tecnologia existente como instrumento de integração destas pessoas.

O programa interagirá como hardware adaptado, trazendo ao usuário comodidade e facilidade, possibilitando-o usufruir da tecnologia já disponível no mercado, porém adaptado à sua realidade.

O desenvolvimento de tal software será documentado, utilizando-se da linguagem para modelagem de sistemas: UML como base para a especificação completa do mesmo, desde a descrição dos atores envolvidos, das regras de negócio e requisitos funcionais e não funcionais, até diagramas de caso de uso e de classes.

1.5. Resultados Esperados

Como resultados esperados, podem ser citados diversos fatores que influenciam diretamente a vida do portador de necessidades, seus professores, facilitadores e familiares.

Como principal resultado da utilização do hardware e software adaptados, pode-se destacar o auxílio à pessoa portadora de necessidades especiais no desenvolvimento de seu potencial motor e intelectual através de adaptações de software/hardware e, dessa forma, aumentar sua auto-estima e por conseqüência uma melhora na qualidade de vida.

Além do impacto ao próprio portador, profissionais desta área, como pedagogos e enfermeiros, terão nesta tecnologia uma fonte de recursos práticos que viabiliza a implementação de atividades lúdicas com fins didáticos.

Como ferramenta didática, o software deve possuir registro do histórico de ações do usuário, onde através dele, seja possível realizar a análise da evolução deste usuário, medindo seu desempenho nas atividades e permitindo, dessa forma, o acompanhamento do seu processo de aprendizado.

O software, juntamente ao hardware adaptado, busca dispor aos pais e familiares maior atratividade, no sentido da interação entre estes e a pessoa portadora de deficiência, onde nem sempre é tão simples haver tal contato, visto que alguns sintomas das deficiências levam os familiares a se distanciarem dos portadores.

Além de todas as funcionalidades e oportunidades oferecidas pelo uso do software, cabe destacar que este programa será distribuído à população de forma gratuita e sem burocracia, visando o máximo acesso destas pessoas.

2. Embasamento Teórico

2.1. Deficiências Motoras

Esta seção tem por objeto apontar possíveis perfis de usuários para o presente projeto, ou seja, pessoas com limitações físicas e com seu cognitivo preservado. Tais limitações podem ser oriundas de diversas patologias. Algumas delas serão abordadas abaixo:

2.1.1. Paralisia Cerebral

As palavras paralisia e cerebral são usadas para descrever uma condição de ser, um estado de saúde, uma deficiência física adquirida, uma diminuição de aptidões físicas que durante muito tempo teve o significado de "invalidez".

Atualmente, o termo "Paralisias Cerebrais" vem sendo usado como o significado do resultado de um dano cerebral, que leva à inabilidade, dificuldade ou o descontrole de músculos e de certos movimentos do corpo. O termo cerebral quer dizer que a área atingida é o cérebro (Sistema Nervoso Central - S.N.C) e a palavra paralisia refere-se ao resultado do dano ao S.N.C., com conseqüências afetando os músculos e sua coordenação motora, dos portadores desta condição especial de ser e estar no mundo.

Estas lesões possuem diversas causas, freqüentemente devido à falta de oxigenação antes, durante ou logo após o parto, não existindo dois casos semelhantes, pois algumas crianças têm perturbações sutis, quase imperceptíveis, aparentando serem "desajeitadas" ao caminhar, falar ou usar as mãos. No entanto, as que submetidas a lesões cerebrais mais graves, a exemplo de casos de anóxia neonatal podem apresentar incapacidade motora acentuada, impossibilidade de falar, andar e se tornam dependentes para as atividades cotidianas.

Desta forma, as causas são divididas em dois grupos:

Paralisia Cerebral não é doença, mas uma condição médica especial, que freqüentemente ocorre em crianças no momento do parto, e quase sempre é o resultado da falta de oxigenação ao cérebro. As crianças afetadas por Paralisias Cerebrais têm uma perturbação do controle de suas posturas e dos movimentos do corpo, como conseqüência da lesão cerebral.

A Paralisia Cerebral também não é contagiosa. A pessoa portadora desta paralisia tem inteligência normal, a não ser que a lesão tenha afetado áreas do cérebro responsáveis pelo pensamento e pela memória. Se a pessoa portadora de Paralisia Cerebral tiver sua visão ou audição prejudicada pela lesão, terá dificuldades para entender as informações como normalmente são transmitidas.

A Paralisia Cerebral atinge diversas regiões do cérebro. Dependendo de onde ocorre a lesão e da quantidade de células atingidas, diferentes partes do corpo podem ser afetadas, alterando o tônus muscular, a postura e provocando dificuldades funcionais nos movimentos.

Alguns sintomas dos portadores de deficiência podem ser: os movimentos involuntários, alterações do equilíbrio, do caminhar, da fala, da visão, da audição e da expressão facial. Em casos mais graves pode haver comprometimento mental.

Se o controle da fala for atingido, a pessoa terá dificuldades para comunicar seus pensamentos ou necessidades. Quando tais fatos não são observados, a pessoa portadora de Paralisia Cerebral pode ser erroneamente classificada como deficiente mental ou não inteligente.

Homens e mulheres portadores de Paralisia Cerebral podem ter filhos como qualquer outra pessoa. As características dos óvulos e dos espermatozóides, bem como a estrutura dos órgãos reprodutores, não são afetadas pela lesão cerebral.

2.1.2. Ataque vascular cerebral

Ataque vascular cerebral (AVC) ou derrame, como é popularmente conhecido, refere-se geralmente a súbita falta de sangue em determinada região do cérebro, sem hemorragia. O AVC pode ser classificado por dois tipos: isquêmico e hemorrágico. O AVC isquêmico ocorre devido a obstrução de uma das artérias do cérebro, que provoca a morte da região cerebral que ficou sem o fluxo sangüíneo adequado. O AVC hemorrágico ocorre por sangramento de uma das artérias do cérebro, causando mais ou menos os mesmos tipos de sintomas do AVC isquêmico.

O AVC nem sempre leva o paciente ao óbito. Ele pode deixar seqüelas em seus pacientes, dentre elas:

- Hemiplegia (paralisia de um lado do corpo),
- Deformidades nos membros (devido à paralisia),
- Dores difusas pelo corpo,
- Complicações pulmonares,
- Paralisia facial,
- Alteração da sensibilidade,

Neste contexto, podemos considerar que vítimas do AVC podem fazer parte do publico alvo deste trabalho, principalmente no que se refere a hemiplegia.

2.1.3. Lesão Medular

O cérebro é responsável por comandar todo o corpo enviando e recebendo informações através dos neurônios. Desta forma, o cérebro sabe, por exemplo, se há alguma coisa quente sobre a pele através da informação que recebeu através de nervos sensitivos e neste momento ordena que o corpo se afaste da fonte de calor. Esta ordem chega aos músculos através dos nervos motores. Todos os terminais nervosos saem do cérebro, passam por dentro da coluna vertebral e chegam ao seu destino, ou pés, mãos, algum órgão, etc. A lesão medular é a interrupção deste trajeto em algum nível da coluna vertebral. Assim, quanto mais alto for o nível da lesão, mais comprometimentos haverá, pois haverá interrupção da comunicação abaixo da lesão.

A Lesão medular é devida à quebra de comunicação entre o cérebro e a medula. Todas as partes do corpo que ficam abaixo da lesão também são interrompidas. Este tipo de lesão geralmente ocorre devido a quedas de alturas, ferimentos com armas brancas ou de fogo, mergulhos em locais rasos, entre outros. Normalmente a recuperação é inexistente ou lenta

O termo lesão medular é utilizado para se referir a qualquer tipo de lesão que ocorra nos elementos neurais do canal medular. Em uma lesão medular alta, a seqüela, muitas vezes é a tetraplegia, sendo necessário um tratamento fisioterapêutico para prevenir retrações e hipotrofias musculares; tendo também como aparato, a utilização de órteses para deixar o paciente o mais independente possível, devolvendo de forma limitada, mas tornando seu movimento funcional (Hall e Lindzey, 1984).

Assim, pessoas que foram tiveram lesão medular são potenciais usuários para a solução aqui proposta.

2.2. Acessibilidade

Acessibilidade significa não apenas permitir que pessoas com deficiências participem de atividades que incluem o uso de produtos, serviços e informação, mas a inclusão e extensão do uso destes por todas as parcelas presentes em uma determinada população, com restrições as mínimas possíveis. (Wikipédia)

Neste contexto, a acessibilidade é fator indispensável quando se fala em adaptação de software e hardware para portadores de necessidades especiais. O conceito da acessibilidade que adotaremos aqui envolve três dimensões fundamentais, "Utilizadores", "Situação" e "Ambiente":

- Na dimensão "Utilizadores" um software acessível é aquele em que nenhum obstáculo é imposto ao indivíduo face às suas capacidades sensoriais e funcionais;
- Na dimensão "Situação" determina-se que o sistema é acessível e utilizável em diversas situações ou contextos, independentemente do software, comunicações ou equipamentos;
- Na dimensão "Ambiente" entende-se que o acesso não deve estar condicionado pelo ambiente físico envolvente, exterior ou interior, no ambiente da realização das atividades.

2.2.1. Deficiências e acessibilidade

Segundo Bieler (2003) todas as pessoas deveriam ter acesso aos serviços primários de saúde e à educação básica, e menciona [...] “aproximadamente quatro quintos das pessoas com deficiência existentes no mundo vivem em países em desenvolvimento. A pobreza cria condições para a deficiência e a deficiência reforça a pobreza”.

A Organização Mundial da Saúde (OMS) estabeleceu a Classificação Internacional de Deficiências, Incapacidades e Desvantagens como:

Deficiência: toda perda ou anormalidade de uma estrutura ou função psicológica ou anatômica;

Incapacidade: toda restrição ou falta – devida a uma deficiência – da capacidade de realizar uma atividade na forma ou na medida em que se considera normal a um ser humano;

Desvantagem: uma situação prejudicial para determinado indivíduo, em conseqüência de uma deficiência ou uma incapacidade, que limita ou impede o desempenho de um papel que é normal em seu caso (em função da idade, sexo e fatores sociais e culturais).

O Censo do IBGE (Instituto Brasileiro de Geografia e Estatística) de 2000 apontou que 24,5 milhões de brasileiros apresentam algum tipo de deficiência, isto equivale a 14,5% da nossa população. E a ONU, em suas pesquisas, apontou que no mundo existem 500 milhões de pessoas com deficiência.

Podemos entender por "acessibilidade" o conjunto de esforços que se realiza em diferentes âmbitos da atividade humana para facilitar o acesso a meios e recursos sociais, culturais, educacionais, etc., com o objetivo de reduzir o efeito de uma limitação do meio ambiente e assim proporcionar uma maior igualdade às pessoas com necessidades especiais. As barreiras de acessibilidade atingem a todos, pois em muitos casos, as pessoas podem ter dificuldades em utilizar as TIC, devido às características específicas próprias ou do ambiente onde se encontram e por este motivo, há uma estreita relação entre os conceitos de "acessibilidade" e "desenho universal". O desenho universal não abrange apenas as pessoas com necessidades especiais; leva em consideração as múltiplas diferenças existentes entre o maior número de pessoas. A idéia é evitar a necessidade de ambientes e produtos especiais adaptados para pessoas com "deficiência", buscando garantir a acessibilidade a todos os componentes do ambiente e a todos os produtos concebidos no decorrer do desenvolvimento de um projeto, para que sejam utilizados por todas as pessoas com diferentes capacidades, o maior tempo possível, sem a necessidade de adaptações.

A palavra "acessibilidade" pode ser encarada não apenas como o acesso à rede de informações, mas também como o rompimento das barreiras de comunicação, equipamentos e software adequados às diferentes necessidades especiais, bem como conteúdo e apresentação da informação em formatos alternativos, ou seja, garantia de mobilidade e usabilidade de recursos computacionais.

Segundo a UNIVERSIDADE DE TRÁS-DOSMONTES E ALTO DOURO, 2002, o conceito de "acessibilidade" pode ser definido em três enfoques:

Sujeito (PNEE): significa que nenhum obstáculo é imposto ao sujeito em face das suas capacidades sensoriais, funcionais e cognitivas;

Situação: significa que o sistema é acessível e utilizável em diversas situações, independentemente do software, comunicações ou equipamento;

Ambiente: significa que o acesso não é condicionado pelo ambiente físico envolvente, exterior ou interior.

2.2.1. Acessibilidade na tecnologia

Um computador ou um software pode ser dito acessível quando possibilitar que pessoas com vários graus de capacidades físicas possam fazer uso dele com êxito no trabalho, na educação e como fonte de entretenimento.

Nos dias de hoje, com a massificação do uso do computador, a preocupação com a acessibilidade é cada vez mais necessária. O próprio sistema operacional Microsoft Windows® tem se preocupado com a acessibilidade, seja disponibilizando um teclado virtual, seja aumentando o tamanho das fontes ou até mesmo disponibilizando uma lupa de aumento. Ainda assim, muito há para ser feito para proporcionar um acesso igual e razoável ao mundo dos computadores.

Porém, no foco deste projeto, consta um perfil de usuário que os sistemas operacionais normalmente não contemplam, pessoas com limitações físicas e capacidade cognitiva preservada. Assim, este trabalho pode contribuir bastante na vida destas pessoas.

2.3. Sistema de Comunicação Alternativa

O grande problema de crianças e adultos que possuem deficiências mentais e/ou motoras advindas de doenças como paralisia cerebral, autismo, seqüelas de acidentes vasculares encefálicos etc, é a grande dificuldade de comunicação, seja com a mãe ou responsável, ou com seu monitor, médico ou enfermeiro e até mesmo com outras crianças e pessoas com quem brincam e interagem.

Diante deste problema e da necessidade de resolução do mesmo, foram criados Sistemas de Comunicação Alternativa. Estes sistemas são utilizados por inúmeros fonoaudiólogos, psicólogos e pedagogos no Brasil e pelo mundo afora, como Estados Unidos, Canadá, Inglaterra, Austrália etc, e já contam com associações e sociedades de profissionais e usuários.

A Comunicação Alternativa é definida como um conjunto de técnicas que um indivíduo utiliza quando os modos usuais, como a linguagem oral e escrita, estão comprometidos. Estas técnicas são muito variadas, reunindo materiais gráficos, como desenhos, fotos, sinais gráficos e a palavra escrita, organizados em uma prancha de comunicação, permitindo a interlocução além da fala.

A maioria dos fonoaudiólogos utiliza as *pranchas*, as quais podem ser feitas de papel, pastas de plástico ou em modo digital, utilizando o computador.



Figura 1. Exemplo de Pranchas de comunicação

Segundo a Dra. Regina Yu Shon Chun[4], que trabalhou na Associação Educacional Quero Quero, em São Paulo e fez o Curso Básico do Sistema Bliss de Comunicação em é membro da ISAAC e uma das pioneiras nessa área, todos os sujeitos com necessidades especiais que apresentam prejuízo da oralidade,

[4] A Sra. Yu Shon Chun é doutora em xxxx, detentora dos conhecimentos do Sistema Bliss, é membro do ISAAC (descrever o q eh)

escrita ou gestualidade deveriam ter acesso a algum Sistema Alternativo de Comunicação. Referencia da fala

A seguir serão demonstrados alguns dos sistemas mais utilizados no mundo e no Brasil, e suas pecularied ades.

2.3.1. Sistema Bliss

O Bliss é um sistema alternativo de comunicação, utilizado por portadores de paralisia cerebral sem comprometimentos intelectuais e com déficits de linguagem. Portadores de outras deficiências, como autismo e afasia, também o utilizam.

Segundo informa a fonoaudióloga Roseli Vasconcelos, os símbolos do sistema foram idealizados pelo austríaco Charles Kaisel Bliss, entre 1942 e 65, com o intuito de ser uma linguagem universal para unir falantes de diferentes línguas. O objetivo não teve êxito, mas o sistema foi adaptado por profissionais de uma instituição de Toronto (Canadá) para a comunicação de portadores de Paralisia Cerebral com sucesso. Logo, o Bliss passou a ser usado internacionalmente.

No Brasil, o Bliss foi trazido do Canadá no final da década de 70, e utilizado pela primeira vez na Associação Educacional Quero Quero de Reabilitação Motora e Educação Especial, em São Paulo, pioneira na utilização de Comunicações Alternativas no país.

Os símbolos partem de formas básicas e foram inspirados na pictografia chinesa, na lógica matemática e nas idéias do filósofo Leibniz (1646-1716).



Mulher **Proteção** **Mãe**

Figura 2. Exemplo de símbolos do Sistema Bliss

Estes símbolos são divididos em três grupos:

- Pictográficos: semelhantes aos objetos que representam
- Ideográficos: sugestivos dos conceitos que representam
- Arbitrários: reconhecidos por convenção internacional

Geralmente estes tipos de símbolos são recombinados para modular o significado e, assim, modificados de acordo com a singularidade de cada paciente.

Para alunos que não conseguem fazer uso de sistemas complexos e abstratos como o Bliss, uma opção mais simples é o sistema PCS, que faz uso de 3500 figuras de fácil abstração, onde cada figura torna-se um sinal ou ícone muito claro e objetivo representando o que ser quer fazer entender, seus detalhes serão especificados na próxima seção.

Ambos os sistemas são utilizados sob a forma de pranchas de madeira, podendo ser fixadas na parede ou acopladas à cadeira de rodas. Os símbolos são indicados pelo olhar ou movimentos de cabeça e dispostos da direita para a esquerda, no sentido da frase.

No Brasil, foram desenvolvidos alguns programas de computador, como o LMBrain, ImagoAnaVox e o Comunique, que utilizam principalmente o Sistema Bliss, e também outros Sistemas Alternativos de Comunicação, utilizando o computador como ferramenta de auxílio ao desenvolvimento da oralidade e escrita destas pessoas.

Os três softwares mais utilizados no Brasil serão descritos no capítulo que explana diversos programas de computador que dão suporte à comunicação alternativa.

2.3.2. PCS – Picture Communication Symbol – Símbolos de Comunicação Pictórica

Os Símbolos de Comunicação Pictórica (PCS) formam um sistema de comunicação completo e foram originalmente desenhados para criar, rápida e

economicamente, recursos de comunicação consistentes e com acabamento profissional. São utilizados extensivamente em inúmeros tipos de atividades de aprendizado. Os PCS foram criados no início dos anos 80 pela fonoaudióloga americana Roxanna Mayer Johnson e compõe atualmente o conjunto de símbolos mais difundido no mundo inteiro.

A seguir será apresentadas as características do PCS:

- Desenhos simples e claros, de fácil reconhecimento;
- Adequados para usuários de qualquer idade;
- Estão divididos em seis categorias de palavras: social, pessoas, verbos, descritivo, substantivos e miscelânea;
- Facilmente combináveis com outros sistemas de símbolos, figuras e fotos para a criação de recursos de comunicação individualizados;
- Extremamente úteis numa grande variedade de atividades e lições.
- Disponíveis em livro ou software.



Figura 3. PCS em Livro

3. Estudo preliminar da tecnologia na Comunicação Alternativa

3.1. Uso do Software na Comunicação Alternativa

Os três primeiros sistemas apresentados nesta etapa têm em seu embasamento a utilização do Sistema Bliss como Sistema de Comunicação Alternativa.

3.1.1. LMBrain

O sistema LMBrain foi desenvolvido pela empresa LM Biotecnologia, fundada em 1990, em Belo Horizonte, Minas Gerais, contando com profissionais nas áreas de Engenharia Biomédica e Engenharia Clínica, e parcerias além do Brasil, com Europa e América do Norte.

O LMBrain caracteriza-se por um dispositivo de auxílio à comunicação capaz de utilizar qualquer sistema de comunicação alternativa existente, principalmente o Sistema Bliss, em conjunto com a linguagem escrita. A primeira versão de computador foi desenvolvida para uso no Sistema Operacional Windows.

A filosofia do programa é baseada no uso de PRANCHAS de comunicação. Estas pranchas são divididas em uma matriz de células contendo símbolos, letras, palavras, frases, imagens ou comandos para o computador. As pranchas podem estar interligadas uma as outras através de instruções específicas que são acessadas pelo controle do usuário, permitindo sua navegação por todo o sistema.

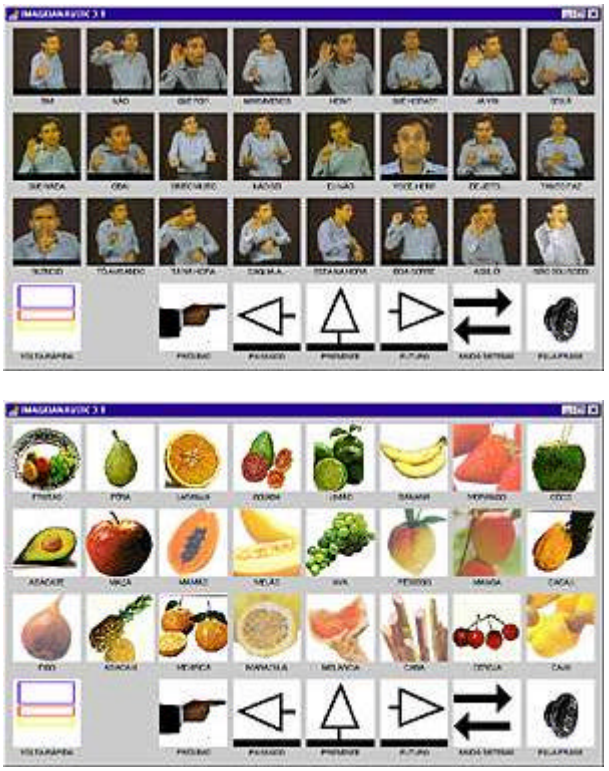
O programa dispõe ainda de vozes digitalizadas e sintetizadas, permitindo a decodificação de letras combinadas escritas pelo usuário.

3.1.2. ImagoAnaVox

O software ImagoAnaVox foi desenvolvido no Departamento de Psicologia Experimental da Universidade de São Paulo, USP, e ao longo dos últimos sete anos, a equipe do psicólogo Elizeu Macedo, de Fernando Capovilla, professor psicólogo da USP e do analista de sistemas Marcelo Duduchi, vem trabalhando em busca de aprimoramentos deste programa que não só cumpriu, como superou as expectativas iniciais.

O ImagoAnaVox é composto por 37 categorias semânticas, como pessoas, objetos, locais, ações, cores, entre outras. Para representação, são utilizados no total 1233 ícones, incluindo também os símbolos do Sistema Bliss, além de fotografias e figuras.

Na figura abaixo se pode visualizar duas imagens da utilização do ImagoAnaVox, demonstrando as figuras, símbolos e imagens.



No programa, há uma série de expressões prontas (no alto), além de imagens de frutas, verbos, objetos, lugares favoritos e fotos das pessoas da família.

Figura 4. Tela do Programa ImagAnaVox

Este programa permite a sonorização de sentenças elaboradas com pictogramas ou palavras escritas, montadas pelo usuário.

Possui várias opções de acesso, várias interfaces entre o programa e o usuário, como mouse adaptado, tela sensível ao toque, varredura automática, etc.

Como exemplo, cita-se Marisa Ribeiro, hoje com 49 anos de idade, que sofreu convulsões quando bebê, que comprometeram o funcionamento de algumas áreas de seu cérebro.

Marisa utilizou, em seu tratamento, ImagoAnaVox, como relata Solange Barreira em reportagem “Software é novo elo de comunicação de quem sofreu lesões cerebrais”, no site da Revista Galileu, da editora Globo.

Solange expõe a experiência de Marisa, que mal conhecia as letras do alfabeto. Primeiramente utilizou as figuras para manifestar seus desejos. Em menos de três anos, passou, com o auxílio do sistema, dominar a linguagem escrita.

3.1.3. Comunique

O software Comunique é distribuído atualmente pelo Centro de Terapia Ocupacional do Rio de Janeiro gratuitamente. Foi desenvolvido por uma terapeuta ocupacional, e demonstrado por Miryam Bonadiu Pelosi, no IV Congresso RIBIE, em Brasília, 1998.

O Comunique tem como objetivo desenvolver a comunicação alternativa oral e escrita de pessoas com deficiência motora, principalmente crianças.

Este programa tem o diferencial da possibilidade de ser personalizado de acordo com o tipo de dificuldade do usuário, a idade, condição motora, possibilidade cognitiva, interesses e necessidades e etc.

O software possibilita a interface entre o usuário e o microcomputador de várias formas, como mouse, joystick, tela sensível ao toque, o próprio teclado e outros acionadores externos de pressão, tração, sopro, voz, entre outros, facilitando assim a interação do usuário com o software.

O Comunique é composto por pictogramas e/ou palavras, utilizando o Sistema Bliss, podendo ser ajustando individualmente, configurando o número de informações mostradas na tela. A disposição dos símbolos pode ser realizada em uma mesma tela ou em telas encadeadas, personalizando cada usuário.

3.1.4. Escrita com Símbolos

Escrita com símbolos é um processador integrado de textos e de símbolos, destinado a pessoas que apresentam dificuldade na comunicação escrita. Além de poder facilitar o processo de aquisição da leitura e escrita, contribui para dar autonomia ao seu usuário. Estes sistemas dispõem de diversas ferramentas que podem facilitar tanto o processo de alfabetização de crianças e adultos com dificuldades de comunicação, quanto o trabalho do educador ou professor na organização de atividades.



Incorporado ao sistema, há um sintetizador de voz em português de Portugal de excelente definição e que permite ler tudo o que está escrito.

Na compra do software estão inclusos:

- Manual de utilização.
- Dicionário de significados com imagens alternativas;
- 4000 símbolos PCS da Mayer & Johnson e 5500 símbolos REBUS;
- Tutorial interativo;
- Sintetizador de voz em Português-Portugal;
- Verificador de ortografia pictográfico para aprendizagem e compreensão da linguagem.

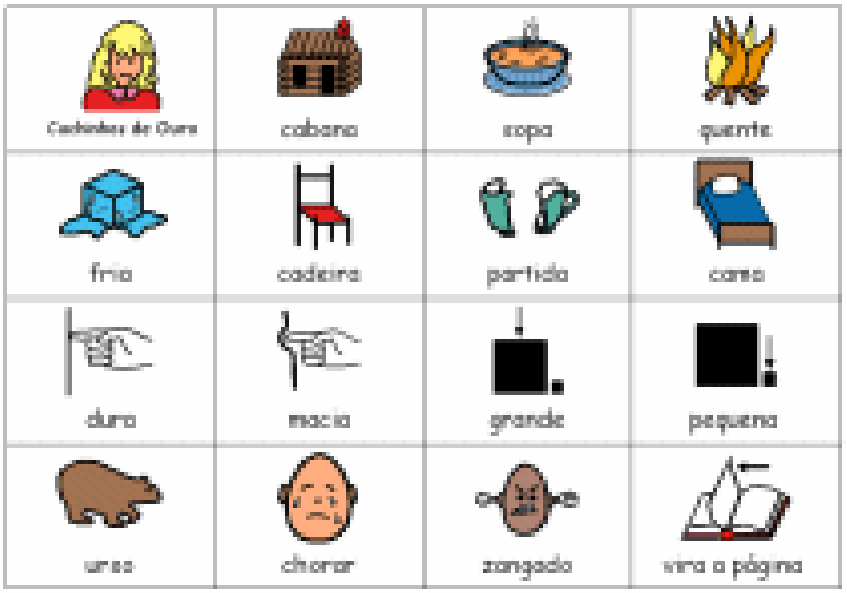


Figura 5. Alguns símbolos do software Escrita Com Símbolos

O software apresenta quatro formas diferentes de se trabalhar:

- Processador de símbolos;
- Processador de palavras;
- Grades para impressão;
- Grades de escrita

Escrita Com Símbolos possui ainda um outro programa, o Gestor de Recursos, onde são agrupadas as funcionalidades, e a partir dele é possível configurar e personalizar o software de acordo com as limitações do usuário.

Algumas funcionalidades do Escrita Com Símbolos:

- Possibilidade de ouvir tudo o que está escrito, com um sintetizador de voz em Português de Portugal;
- Utilização de modificadores, que permitem uma extensão do vocabulário disponível;
- Possibilidade de adicionar som aos símbolos e imagens;
- Possibilidade de diferentes palavras possuírem o mesmo símbolo;
- Possibilidade de acrescentar as suas próprias imagens e símbolos;
- Existência de símbolos alternativos para a mesma palavra;

Este é um produto recomendado e avaliado pelo Ministério da Educação e é recomendado a diversos tipos de pessoas:

- Pessoas que não utilizam a linguagem escrita podem começar a reconhecer e ordenar imagens para comunicar suas idéias.
- Pessoas com dificuldades de soletração, reconhecimento de palavras, ou compreensão ou que, simplesmente, necessitem de motivação para escrever, podem ser estimuladas e auxiliadas através das imagens, dos símbolos e do som (por exemplo, crianças com dislexia).
- Pessoas com dificuldades de aprendizagem podem utilizar símbolos como forma de acesso à leitura e à escrita, adquirindo assim independência e autonomia.
- Pessoas que utilizam sistemas de comunicação aumentativa e alternativa como recursos normais à comunicação.

3.1.5. GRID

O GRID consiste num sistema de teclados em tela, que substitui todas as funções do teclado e mouse, bem como os comandos do Windows. Esses teclados virtuais podem ser acedidos através do mouse ou de qualquer outro dispositivo apontador, como trackball, joystick, canetas para telas sensíveis ao toque, etc.

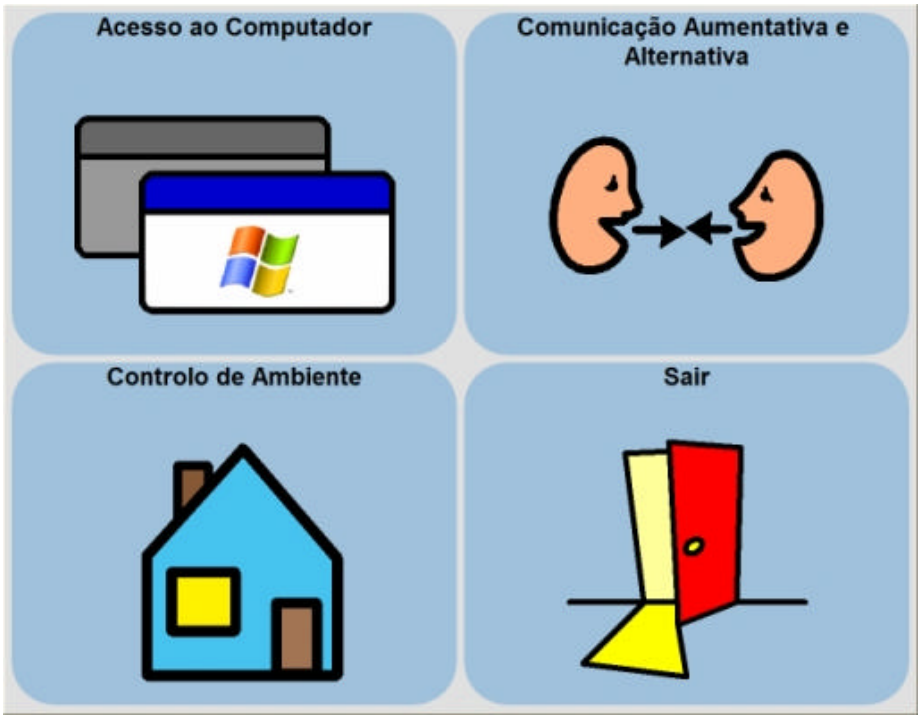


Figura 6. Interface do software GRID

Este programa contém ainda um sintetizador em português de Portugal de elevada qualidade, que possibilita aos seus usuários expressarem-se através deste sistema de conversão texto-fala.

Como visto na figura que demonstra a interface do software, o GRID inclui três funcionalidades distintas:

- Teclados para Acesso ao Computador
- Teclados para Comunicação Aumentativa
- Teclados para Controle de Ambiente

Esta solução possibilita ao seu utilizador expressar-se e comunicar de forma autônoma, utilizando o computador e os programas nele contidos, navegar na Internet e também controlar totalmente o seu ambiente físico (abertura e fechamento de portas e janelas, acesso ao telefone, TV, equipamento de áudio e vídeo, etc.).

O GRID é um programa da empresa Sensory Software, traduzido e adaptado para o português pela ANDITEC, fazendo parte de um conjunto de programas para a Comunicação Alternativa, desenvolvidos para o projeto "PT

Minha Voz" da Fundação Portugal Telecom (Gabinete para Clientes com Necessidades Especiais). O projeto contou ainda com a colaboração do INOV para a inclusão do sintetizador de fala em Português nos programas.

Este software está em constante evolução e desenvolvimento.

Em fevereiro de 2005, foi lançado o programa GRID para Windows CE, a ser utilizado em PocketPCs, como a figura abaixo:



Figura 7. Grid em PocketPC

O GRID para computadores de mão pode ser aliado ao programa GEWA, o qual utiliza a tecnologia de infra-vermelho para o usuário controlar televisões, aparelhos de som, e outros dispositivos adaptados como portas, lâmpadas e outros, trazendo poder ao paciente, muitas vezes até então incapaz.

3.1.6. InVento



O InVento é um programa de publicação que permite construir e imprimir cartazes, folhetos, livros, materiais pedagógicos, quadros de comunicação de uma forma muito simples e rápida.

Tem a vantagem de suportar imagens e/ou símbolos como suporte à alfabetização. Vem com o conjunto, de cerca de 7.500

símbolos Widgit Rebus coloridos e preto e branco. Possui também uma biblioteca com mais de 1500 imagens e fotografias coloridas. Estas podem ser utilizadas individualmente ou em conjunto, permitindo construir cenários para ilustrar a escrita. Também é possível utilizar as suas próprias imagens ou fotografias.



Figura 8. Imagens utilizadas no InVento

A figura acima demonstra a grande parametrização que o software disponibiliza, incluindo diversas culturas, dando ampla adaptabilidade ao usuário.

O software disponibiliza um sintetizador de voz em Português Europeu (Madalena) de alta definição que permite ler tudo o que está escrito.



Figura 9. Alguns símbolos do InVento

Serão listadas a seguir algumas funcionalidades existentes no InVento:

- Ler tudo o que está escrito com um sintetizador de voz em Português Europeu (Madalena) de excelente definição;
- Escrever texto ilustrado por símbolos, imagens e/ou fotografias;
- Utilizar balões de fala e molduras para caixas de texto e imagens;
- Aceder a uma galeria com mais de 1.500 imagens para ilustrar os trabalhos realizados;
- Usar cerca de 7.500 símbolos da colecção Widgit Rebus coloridos e a preto e branco;
- Inserir e utilizar inserir as suas próprias fotografias digitais ou imagens digitalizadas;

- Imprimir os trabalhos realizados em qualquer formato: A4, A4, A3; livro pronto a dobrar; ou folhas soltas;
- Construir facilmente grades para horários, fichas de trabalho ou quadros de comunicação, com tenham texto e/ou símbolos.

Esta é uma ferramenta inclusiva que permite adaptar os textos a qualquer público.

3.1.7. BoardMarker

O BoardMaker é uma ferramenta desenvolvida pela equipe de Erinoak, Mississauga, Ontario, Canadá, a ser utilizado em plataforma Windows e Macintosh.



O Objetivo deste software é confeccionar pranchas com imagens e palavras altamente configuráveis.

O BoardMaker encontra-se totalmente em português brasileiro, além de outros 15 idiomas e pode ser localizado e impresso em qualquer uma das linguagens oferecidas.

Os símbolos podem figurar de três modos: sem texto, com uma linha de texto acima do símbolo, ou com duas linhas acima do símbolo, com o texto em dois idiomas diferentes.

Estes símbolos são classificados por categorias, podendo ser filtrados com o uso de miniaturas, visualizando até 49 símbolos de uma só vez, como por exemplo “higiene pessoal” ou “limpeza”.

Utilidades do BoardMaker:

- Confeccionar pranchas com qualidade profissional em minutos;
- Localizar e aplicar símbolos e imagens com um clique do mouse;
- Trabalhar as imagens em qualquer tamanho e espaçamento;
- Imprimir e/ou salvar a sua prancha de comunicação para uso posterior;
- Imprimir pranchas em cores ou preto-e-branco (dependendo do tipo de impressora);

- Armazenar, nomear, organizar, redimensionar e aplicar imagens escaneadas de sua preferência;
- Criar folhas de tema ou trabalho, lista de instruções pictóricas, livros de leitura, jornais e pôsteres. Podem ser feitos no Boardmaker ou através da colagem eletrônica dos símbolos em seu programa de editoração.
- Acompanha várias grades prontas de calendários e agendas para você confeccionar materiais rapidamente.

O programa inclui:

- 3.500 Símbolos de Comunicação Pictórica - PCS;
- Dois conjuntos de símbolos: um conjunto em preto-e-branco e outro com os símbolos em cores;
- Dezenas grades pré-fabricadas para construção de pranchas de comunicação;
- Diversos calendários prontos para usar;
- Manual do usuário e índice de símbolos.
- Mais de 15 idiomas.



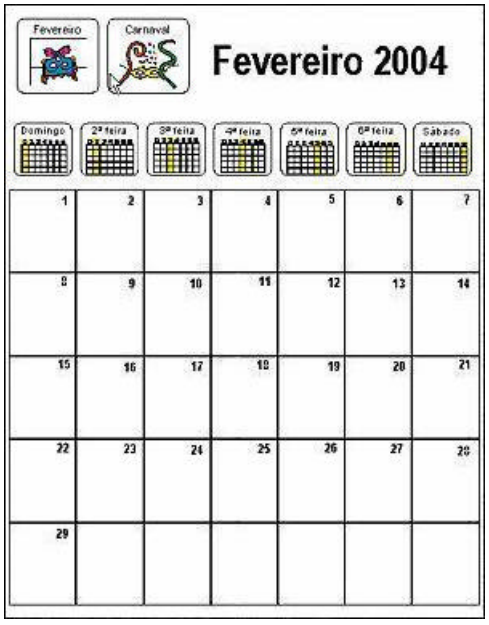




Figura 10. Símbolos PCS

3.1.8. FALAS - Ferramenta ALternativa de Aquisição Simbólica

Esta ferramenta é abordada por Silveira, em sua Dissertação de Mestrado, na Universidade Federal do Rio Grande do Sul, em 1996, como uma aplicação da técnica de Inteligência Artificial à comunicação alternativa.

O software tem como objetivo empregar recursos de Multimídia e técnicas de Inteligência Artificial para aprimorar os sistemas alternativos de comunicação, principalmente, nos seguintes aspectos, a fim de proporcionar uma maior autonomia a seus usuários:

- Adaptação do sistema às preferências do usuário (como sua velocidade de varredura, símbolos mais utilizados) e seu nível de conhecimento do sistema simbólico em questão e do próprio sistema computadorizado sendo utilizado;
- Prestação de auxílio individualizado no aprendizado dos símbolos utilizados, através do acesso ao significado viso-auditivo real ou aproximado dos mesmos, possibilitando o uso de tabuleiros de comunicação "normais" (não computadorizados) de forma mais eficaz.

A arquitetura da ferramenta é composta pelos seguintes elementos: Interface, Controle, Conselheiro, Banco de Símbolos, Banco de AAC e Modelo do Usuário, que serão descritos a seguir.

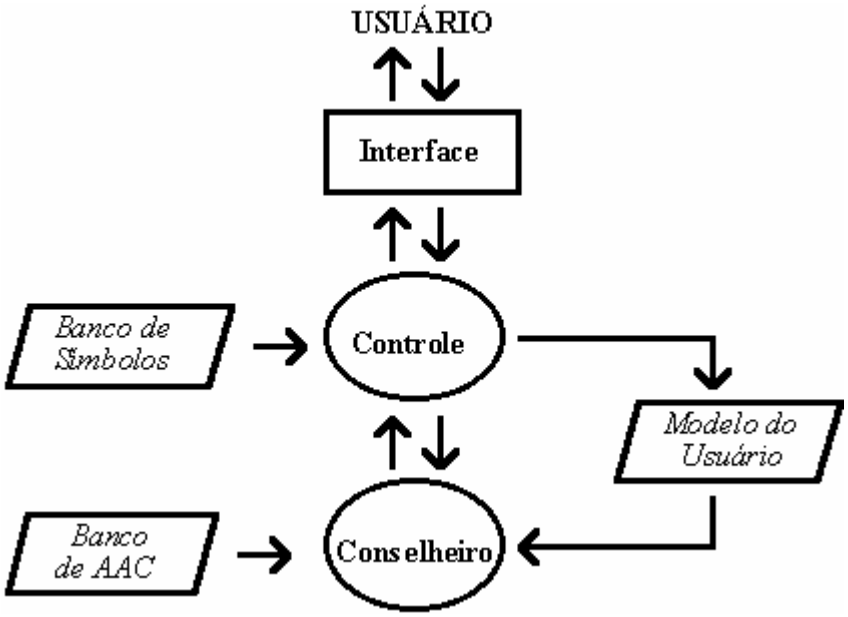


Figura 11. Arquitetura do FALAS

Este componente é dividido em três módulos distintos: Histórico Pessoal, Histórico de Símbolos e Histórico de Exemplos. As informações contidas nestes módulos servirão de base para a verificação do comportamento do usuário, possibilitando, assim, o provimento, pelo sistema, de ajudas e conselhos, inferência de suas intenções e interpretação de seus atos:

- Histórico Pessoal - este módulo é composto pelos dados pessoais do usuário e informações relativas ao uso do sistema (velocidade atual de varredura e número de interações já efetuadas com o sistema);
- Histórico de Símbolos - neste módulo estão armazenados os símbolos utilizados pelo usuário e o comportamento deste em relação a eles;
- Histórico de Exemplos - cada nova frase, criada pelo usuário, é aprendida pelo sistema e armazenada neste módulo, para posterior utilização como exemplo.

3.2. Uso do Hardware na Comunicação Alternativa

Com o objetivo de propor um hardware de baixo custo, foi realizado uma pesquisa por adaptações do teclado e mouse tradicionais de computador. Nas próximas seções será relatada a pesquisa realizada:

3.2.1. Máscaras de teclado

Um dos recursos mais simples e eficientes como adaptação de hardware é a máscara de teclado (ou colméia). Trata-se de uma placa de plástico ou acrílico com um furo correspondente a cada tecla do teclado. Ela é fixada sobre o teclado, a uma pequena distância do mesmo, com a finalidade de evitar que o aluno com dificuldades de coordenação motora pressione, involuntariamente, mais de uma tecla ao mesmo tempo. Esse aluno deverá procurar o furo correspondente à tecla que deseja pressionar. Veja a figura abaixo.



Figura 12. Figura 3.1: Máscara de teclado encaixada no mesmo

Alunos com dificuldades de coordenação motora associada à deficiência mental também podem utilizar a máscara de teclado junto com "tampões" de papelão ou cartolina, que deixam à mostra somente as teclas que serão necessárias para o trabalho, em função do software que será utilizado. Desta forma, será diminuído o número de estímulos visuais (muitas teclas), que podem tornar o trabalho muito difícil e confuso para alguns alunos, por causa das suas dificuldades de abstração ou concentração. Vários tampões podem ser construídos, disponibilizando diferentes conjuntos de teclas, dependendo do software que será utilizado.

3.2.2. Acionadores de varreduras alternativos para uso em softwares especiais

Os softwares desenvolvidos no intuito de facilitar a acessibilidade por parte do usuário, que funcionam com varredura automática, normalmente aceitam o teclado, o mouse e/ou o microfone como acionadores (controladores) dessa varredura. Como exemplo, temo-se os softwares simuladores de teclado, os simuladores de mouse, e os softwares para a construção de pranchas de comunicação alternativa.

Vários desses softwares podem ser baixados gratuitamente pela Internet, como, por exemplo, o simulador de teclado “Teclado Amigo” disponibilizado pela Rede Saci, no endereço: http://www.saci.org.br ou o software para construção de pranchas de comunicação “Plaphoons” do Projeto Fressa, onde são apresentados diversos outros projetos para Comunicação Alternativa.

O maior problema encontrado na utilização destes softwares, é que diversos alunos não conseguem utilizar o mouse, o teclado e nem o microfone, se estes não forem de alguma forma modificados ou adaptados, de acordo com a necessidade de cada usuário.

Dar um clique no botão esquerdo do mouse, por exemplo, pode ser uma tarefa muito difícil ou mesmo impossível para alguns alunos, em função ou das suas dificuldades de coordenação motora fina, ou por causa de alterações anatômicas em seus membros superiores que impeçam a execução dessa tarefa.

Esta adaptação possibilita ampliar a área de acionamento do botão esquerdo do mouse para uma superfície bem maior, com o mesmo efeito de um simples clique no botão.

Trata-se de uma caixa comum de fita de vídeo VHS, dessas onde se guardam as fitas, dentro e no centro da qual é introduzido e fixado, com tira de velcro, um mouse qualquer, como é ilustrado na figura a seguir.

Na capa dessa caixa é colada uma borracha, dessas de apagar lápis, na altura exata onde se encontra o botão esquerdo do mouse. Essa capa da caixa deve fica semi-fechada podendo ser utilizadas pequenas faixas de velcro para mantê-la nessa posição.

Colocando esse dispositivo na frente do aluno, quando ele pressionar qualquer lugar na capa da caixa, a borracha em relevo no interior da mesma entrará em contato com o botão esquerdo do mouse, e o efeito será o acionamento do clique nesse botão, conforme ilustra a figura:



Figura 13. Figura 3.2: Adaptação de mouse

3.2.3. Acionamento pelo microfone por meio de brinquedos de pressão

Quando os softwares especiais permitem (os softwares do Projeto Fressa, por exemplo), a varredura pode ser controlada através de sons ou ruídos no microfone. Mas alguns usuários têm dificuldades ou na articulação ou na sincronicidade exigida, na emissão desses sons ou ruídos.

A solução que utilizamos é acoplar ao microfone, por meio de fitas adesivas, um daqueles pequenos brinquedos infantis de borracha, que produzem sons quando são pressionados.

Dessa forma, o aluno pode comandar a varredura pressionando o brinquedo com a parte do corpo na qual exerça melhor controle (ou mão, ou pé, ou joelho, ou cabeça, etc). Com a pressão, o brinquedo emitirá o som no microfone, que acionará a varredura, conforme ilustra a figura abaixo.



Figura 14. Acionamento por meio de microfone

3.2.4. Adaptações do teclado com sobreposição de teclas

Esta adaptação foi feita por Edson Edir Dick, em sua dissertação de mestrado. Nesta dissertação, ele propunha uma adaptação de software e

hardware, de baixo custo, que possibilite auxiliar as crianças com lesão no cérebro no processo de ensino e aprendizagem e melhoria no desenvolvimento da coordenação motora. Seu projeto foi uma homenagem para seu filho Mateus Henrique, que é portador de Paralisia cerebral.

Em sua adaptação de hardware, Edson faz uso de um teclado comum, sobre o qual fixou a parte superior de outro teclado, idêntico ao primeiro com o objetivo de criar um espaço maior entre a tecla normal do teclado e a grande tecla, bem como servir de apoio para as grandes teclas.

Para fazer as grandes teclas, foram utilizadas tampas cegas de instalações elétricas na qual foi colada algumas teclas retiradas do teclado original, possibilitando assim que apenas uma tecla seja pressionada, ou seja criou-se apenas um ponto de contato da tecla grande com as teclas. Desta forma, as grandes teclas serão encaixadas de forma que as teclas normais fossem ocultadas, dando a impressão da inexistência de um teclado normal, conforme ilustra a figura que segue:



Figura 15. Teclado adaptado por Edson Edir Dick.

Seu próximo passo foi colar imagens que as imagens nas teclas, de forma que seu software "interprete" as interações do usuário com o teclado. Seu software faz a interpretação da tecla grande acionada e verifica se a imagem da tela do software é igual a da tecla precionada. Em caso afirmativo providencia a mudança de imagem, de forma aleatória. Em caso negativo, aguarda que seja pressionada outra tecla.

4. Solução proposta

Após os estudos realizados sobre como a informática auxilia no desenvolvimento de pessoas portadoras de problemáticas advindas de deficiências motoras, pode-se observar que já existem diversos softwares e adaptações de hardware que auxiliam a alfabetização dessas pessoas.

Entretanto, não foi encontrado muito auxílio da informática no estágio de finalização da alfabetização e no desenvolvimento cognitivo posterior a esta etapa, onde se desenvolve o raciocínio lógico e o uso natural da língua mãe escrita.

Portanto é proposto neste trabalho, o desenvolvimento de um software que estimule as crianças a associar figuras aos seus respectivos nomes, e à escrita correta dessas palavras.

Além do software, foi também adaptado um teclado de computador, buscando melhor facilidade de uso do público-alvo, que será detalhadamente mostrado no item posterior.

4.1. Projeto de software

4.1.1. Introdução

Nesta etapa será vislumbrado todo o projeto de modelagem, desenvolvimento e demonstração do software para compreensão de forma macro ou em detalhamento com aprofundamento técnico.

Para a documentação de modelagem, utilizar-se-á a UML – Unified Modelling Language, que é uma linguagem padrão, não proprietária, utilizada para modelar sistemas através de diagramas padrão, diversas regras e descrições textuais.

A documentação formal e a clara especificação do desenvolvimento do sistema visam além de estruturar e embasar a codificação do software, a distribuição do mesmo no meio computacional existente, com a finalidade de ser

reutilizado, ser reavaliado e readaptado para diversos fins no âmbito da pedagogia, seja ela para crianças ou adultos, especiais ou não.

Como base para a busca deste objetivo principal foi realizado grande estudo na área pedagógica e da comunicação alternativa, pois o resultado da análise de um sistema é o enunciado do problema e o projeto do software será sua resolução. Problemas mal enunciados podem até ser resolvidos, mas a solução não corresponderá às expectativas, portanto teremos a seguir diversas etapas que definem a análise para o embasamento ao projeto de software e o encaminhamento para codificação.

Contextualização

Um grande número de pessoas com necessidades especiais encontra nas tecnologias da informação, importantes alternativas e oportunidades para a aprendizagem, para o acesso à informação, ao lazer e ao exercício de uma atividade ou vocação. A acessibilidade a estas tecnologias é, portanto, um fator decisivo para sua qualidade de vida.

Visando bom desempenho no que se diz respeito à acessibilidade, este sistema vem de encontro com as necessidades de comunicação de pessoas que possuem problemas motores, porém de capacidade cognitiva preservada, com o objetivo de proporcionar acesso às alternativas dadas pela tecnologia.

Estrategicamente, utilizou-se de jogos simples, porém desenvolvedores de raciocínio, para buscar uma alternativa a esta classe de pessoas, uma maneira de desenvolver a linguagem escrita e a memorização de símbolos, através de exercícios lúdicos.

Objetivos

Objetivo Geral do Sistema

O sistema proposto - Jardim das Palavras Secretas - tem como objetivo geral o auxílio pedagógico no desenvolvimento da alfabetização dos portadores de necessidades especiais, de forma descontraída e construtiva, auxiliando a

memorização de vários sistemas de comunicação simbólica, e a correlação destes símbolos com as palavras da nossa língua portuguesa, de acordo com o objetivo de cada um dos facilitadores que o utilizarão, visto que o sistema é praticamente todo parametrizado, possibilitando o cadastro de diversos conjuntos de símbolos.

Objetivos Específicos do Sistema

- Servir como ferramenta que auxilie na formação da comunicação, mais especificamente na alfabetização, de pessoas portadoras de deficiências que comprometam o sistema motor;
- Sistema de interação que busca o entretenimento do portador de necessidades, visando integrá-lo à família e amigos, através de um jogo simples, divertido e competitivo;
- Servir como ferramenta para a memorização de sistemas de comunicação alternativos, correlacionando a imagens e palavras escritas na língua portuguesa;
- Através da utilização deste programa, observar e acompanhar a evolução dos usuários através de diversos relatórios que mostram o histórico e desenvolvimento do usuário.

Atores Envolvidos

A seguir serão apresentados todos os atores envolvidos na execução do sistema, com a devida descrição e demonstração de sua interação, seja com outros atores ou com o sistema em si.

Facilitador

O ator Facilitador é o indivíduo que dará quando necessário o suporte ao usuário, explicando as regras ou simplesmente disponibilizando os periféricos adaptados para a realização da atividade.

Ele se identificará no sistema e cadastrará seus próprios usuários. Para cada usuário, o facilitador é o responsável de dar início a atividade e realizar

monitoramentos, que serão gravados em logs do sistema. Para detalhar as atividades, o sistema gera diversos relatórios por usuário ou por facilitador.

Este ator deve conseguir usar o mouse e o teclado comuns (sem adaptações) na sua forma peculiar. Em outras palavras, não pode ter distonias de movimentos, visto que este é quem dará suporte ao usuário portador de problemas motores.

Usuário

O usuário é o individuo que participará efetivamente do jogo e através do teclado adaptado poderá dar as respectivas respostas exigidas pela atividade.

Este ator pode (ou não) ter distonias de movimentos.

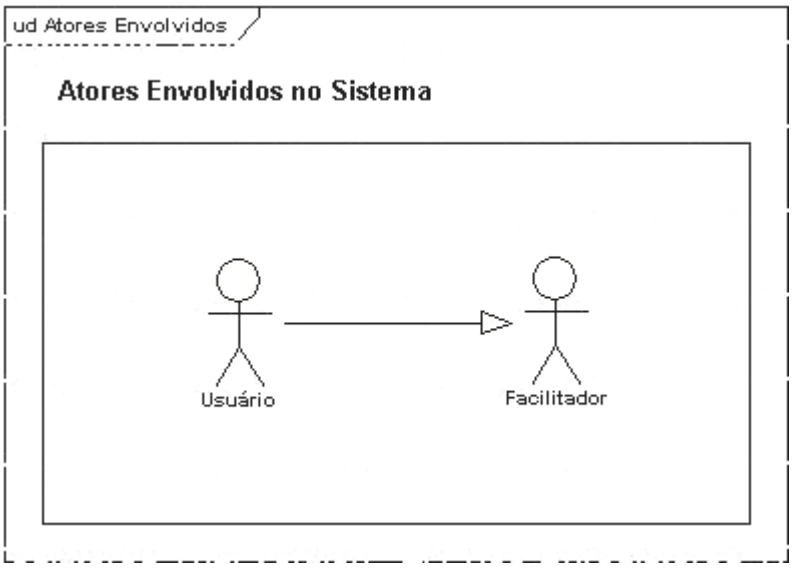


Figura 16. Diagrama de atores envolvidos

Diagramas de Atividades Principal

O diagrama de atividades principal descreve, de uma maneira macro, a atividade central do sistema demonstrando seu principal objetivo.

É possível observar neste diagrama também, os atores envolvidos e sua participação em cada uma das etapas da atividade.

A partir deste fluxo, é descrito, posteriormente, os desmembramentos em diversas outras atividades com maior descrição e detalhamento.

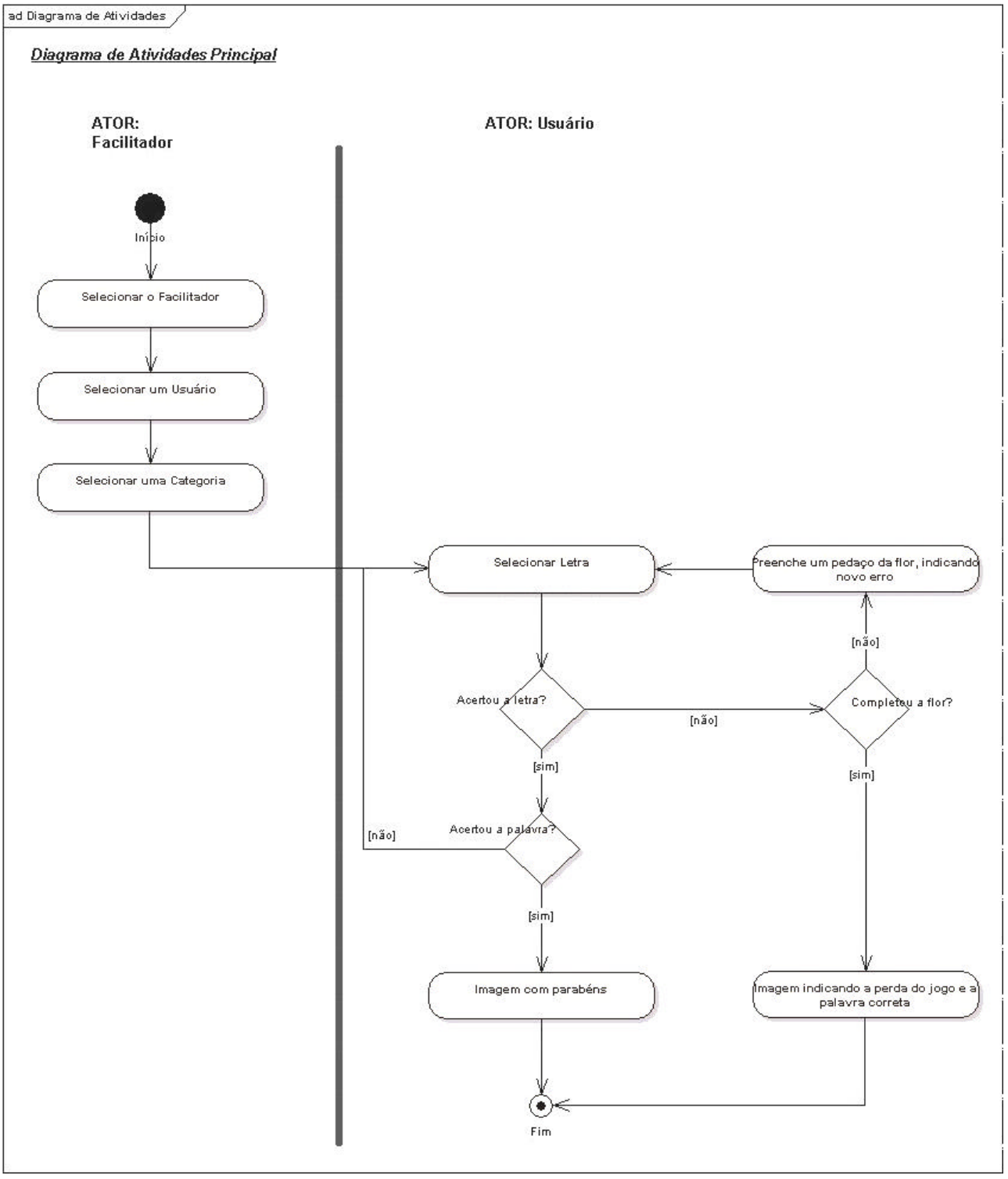


Figura 17. Diagrama de Atividades Principal

4.1.2. Especificação de Requisitos

Nesta etapa serão apresentados os Requisitos e Regras do Sistema, que definirá tanto as funcionalidades que devem ser atendidas pelo software de acordo com a análise pré-realizada, quanto as especificidades que devem ser obedecidas.

Esta é uma das etapas mais importantes e trabalhosas do projeto, pois é neste momento que são definidas as atividades que serão obrigatoriamente desenvolvidas no software, resultando no conjunto de itens que devem ser implementados e atendidos pelo programa, delimitando o escopo de trabalho. Além disso, serão estabelecidas todas as restrições, características e propriedades de cada uma das funcionalidades a serem desenvolvidas.

Os Requisitos são divididos em duas categorias, Requisitos Funcionais e Requisitos Não-Funcionais, os quais serão definidos e apontados a seguir.

As Regras também são divididas em duas visões, que são as Regras de Negócio e as Regras de Usabilidade, definidas e detalhadas adiante.

Requisitos Funcionais

Os Requisitos Funcionais de um Sistema constituem em uma especificação de maneira formal, clara e objetiva de todas as funcionalidades e atividades que os usuários deverão ser capazes de executar com a utilização do sistema.

A seguir será apresentado um diagrama onde são deslumbrados todos os requisitos, enumerados daqui por diante por apenas RF – Requisitos Funcionais.

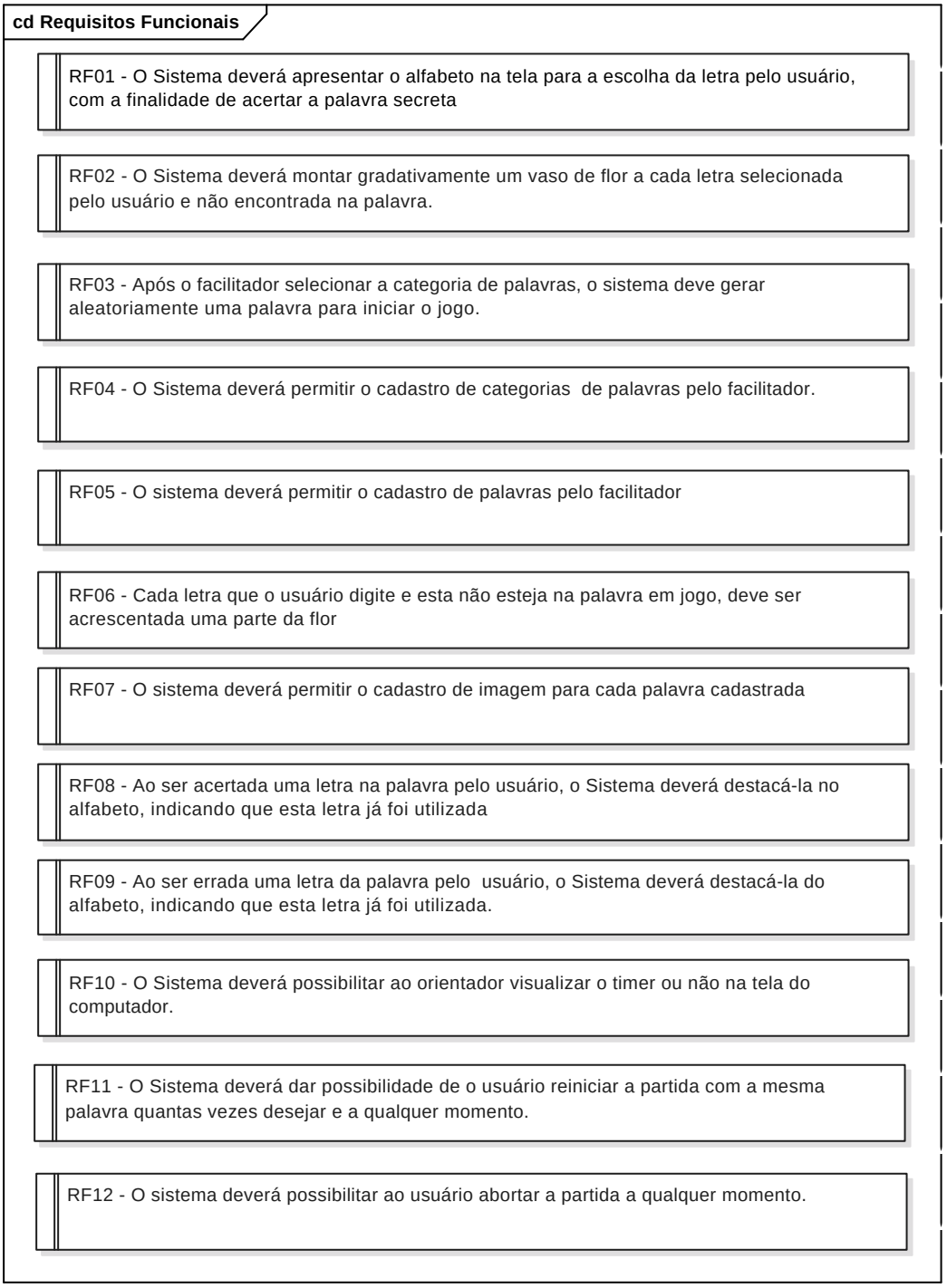


Figura 18. Diagrama de Requisitos Funcionais

Requisitos Não Funcionais

Os Requisitos não-funcionais especificam restrições oferecidas pelo sistema. Incluem restrições de tempo, restrições no processo de desenvolvimento, padrões e qualidades globais de um software, como manutenibilidade, usabilidade, desempenho, custos.

Os Requisitos Não-Funcionais oferecidos por este programa, serão os enumerados a seguir, e definidos por RNF:

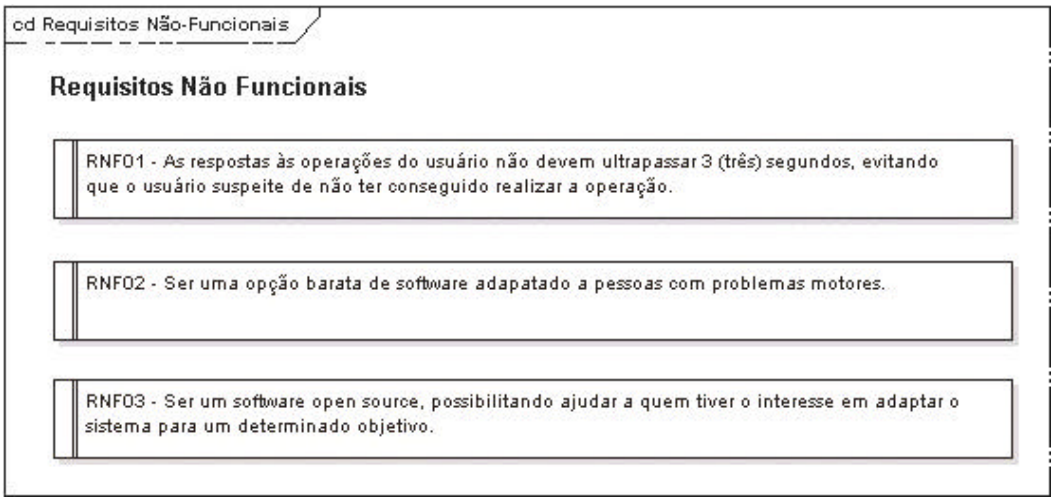


Figura 19. Diagrama de Requisitos Não-Funcionais

Regras de Negócio

As Regras de Negócios são declarações que restringem, derivam e fornecem condições de existência, representando o conhecimento do negócio.

Estas regras estão intimamente relacionadas com os Requisitos Funcionais. A enumeração destas lapida e especifica os requisitos do sistema, sob ângulo do negócio; no caso, de acordo com as regras colocadas pela análise da problemática trabalhada anteriormente, conforme a necessidade dos usuários e dos facilitadores e suas atividades no software.

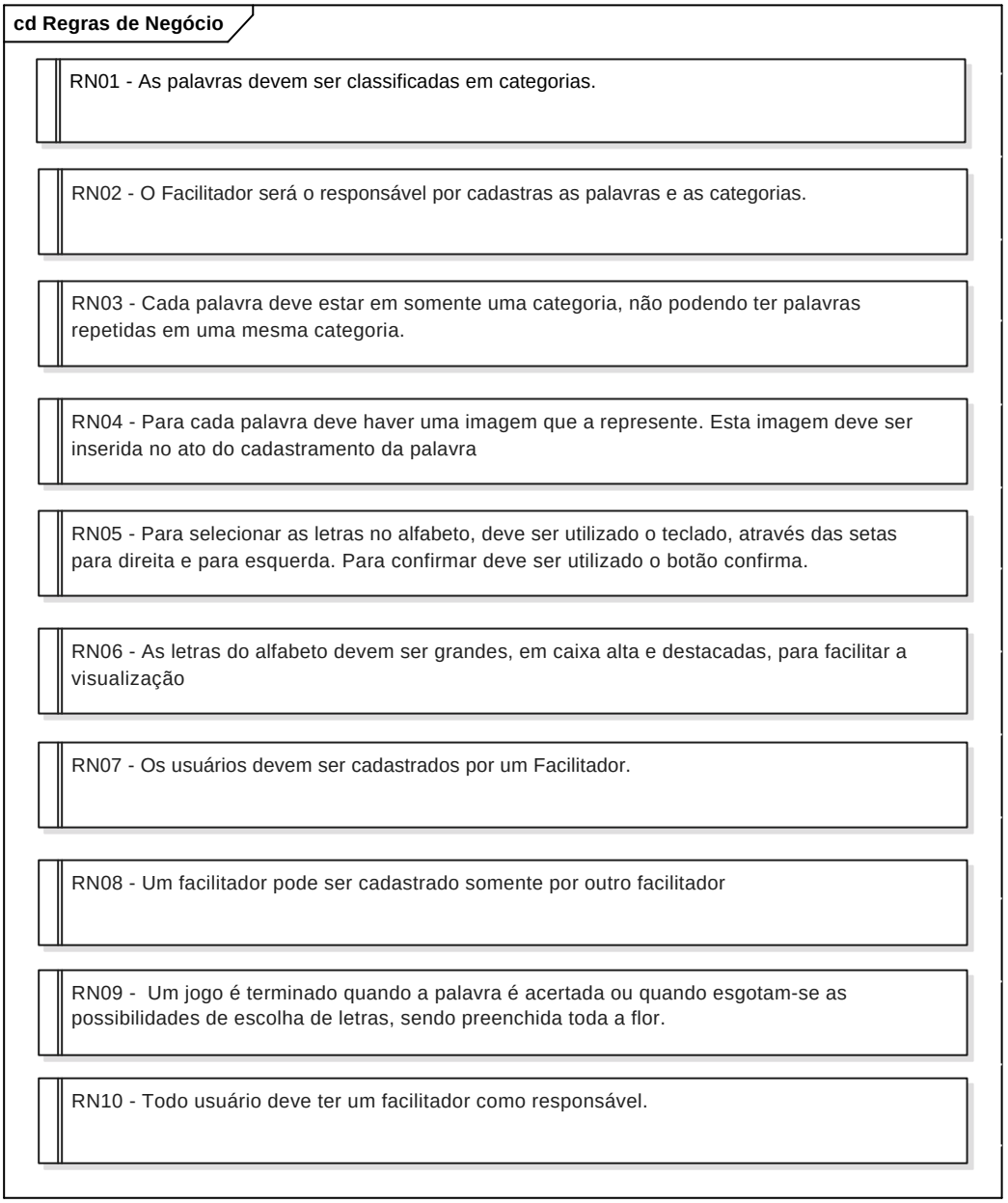


Figura 20. Diagrama de Regras de Negócios

Regras de Usabilidade

As regras de Usabilidade definem as características de interface que serão aplicadas no software de acordo com diversos fatores, buscando a melhor forma de interação com seu usuário.

Conforme Shneiderman, problemas como frustação e ansiedade, no que se refere à utilização de uma interface fazem parte da vida diária de muitos usuários de sistemas de computador, que tentam aprender a linguagem de comandos ou sistema de seleção de menus.

Para algumas pessoas, isso é tão sério que evitam usar a tecnologia no seu dia-a-dia, por mais benefícios que esta possa trazer. No caso abordado por este projeto, as regras de usabilidade são ainda mais essenciais, visto que apenas a utilização de um micro computador já é uma barreira para as pessoas com problemas motores.

Como o desenvolvimento do sistema é voltado para um grupo restrito de pessoas, e estas possuem características que provêm de problemas cognitivos, as regras de usabilidade são bastante exploradas.

Estas regras foram divididas em quatro grupos:

- Regras de Usabilidade – Condução;
- Regras de Usabilidade – Carga de Trabalho;
- Regras de Usabilidade – Controle Explícito e
- Regras de Usabilidade – Gestão de Erros.

A seguir pode ser observado o diagrama com o detalhamento das regras separadas em seus grupos:

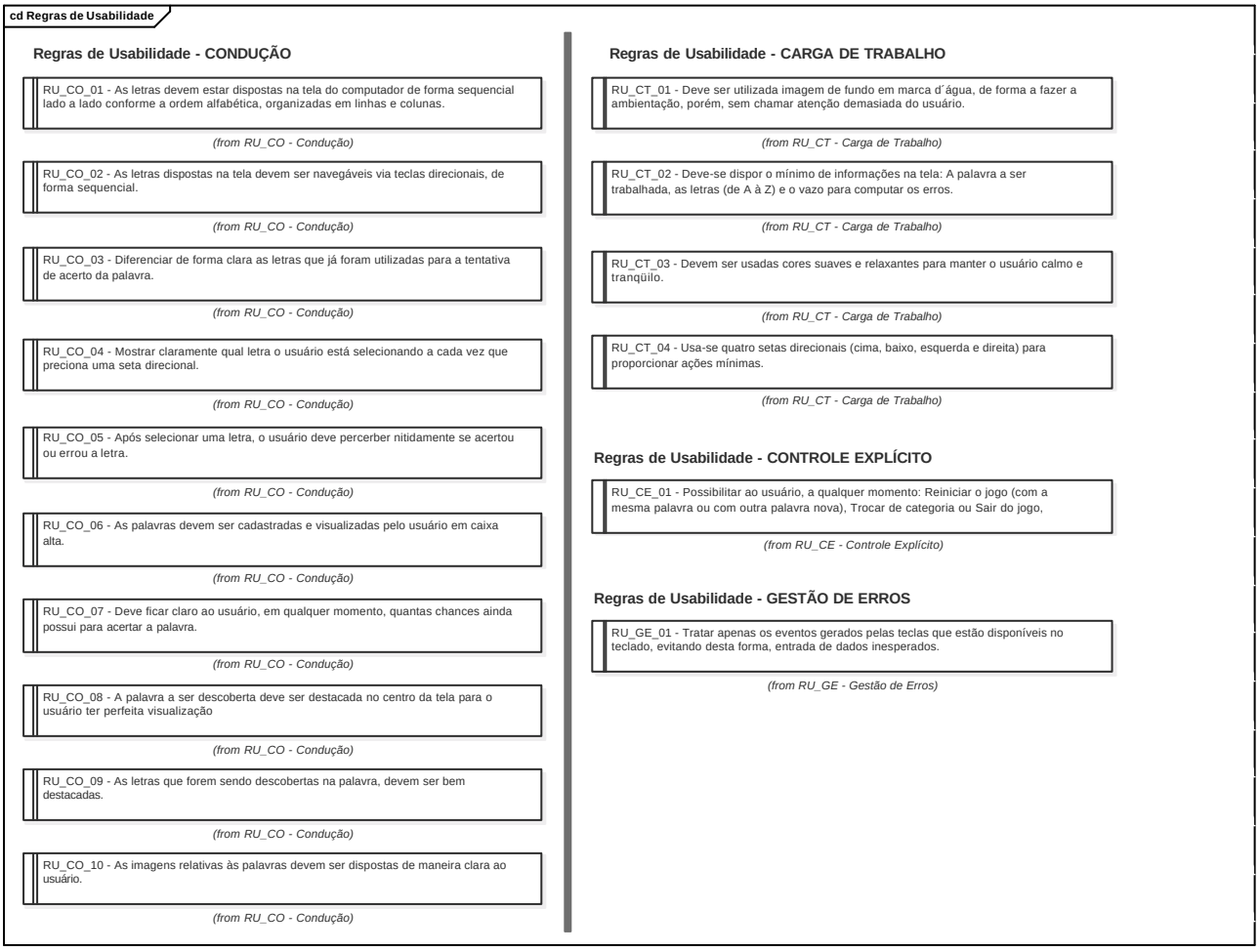


Figura 21. Diagrama de Regras de Usabilidade

4.1.3. Modelo de Casos de Uso

Baseando-se nos requisitos e regras já definidos, são apresentados os casos de uso, que nada mais são que as atividades que serão desenvolvidas e esperadas pelo sistema na utilização de cada ator envolvido.

No diagrama a seguir serão apresentados os casos de uso do sistema e a associação entre esses e os atores do sistema, mostrando desta forma quais casos de uso podem ser executados por cada ator, e logo após, a descrição textual, explicitando cada um dos casos de uso.

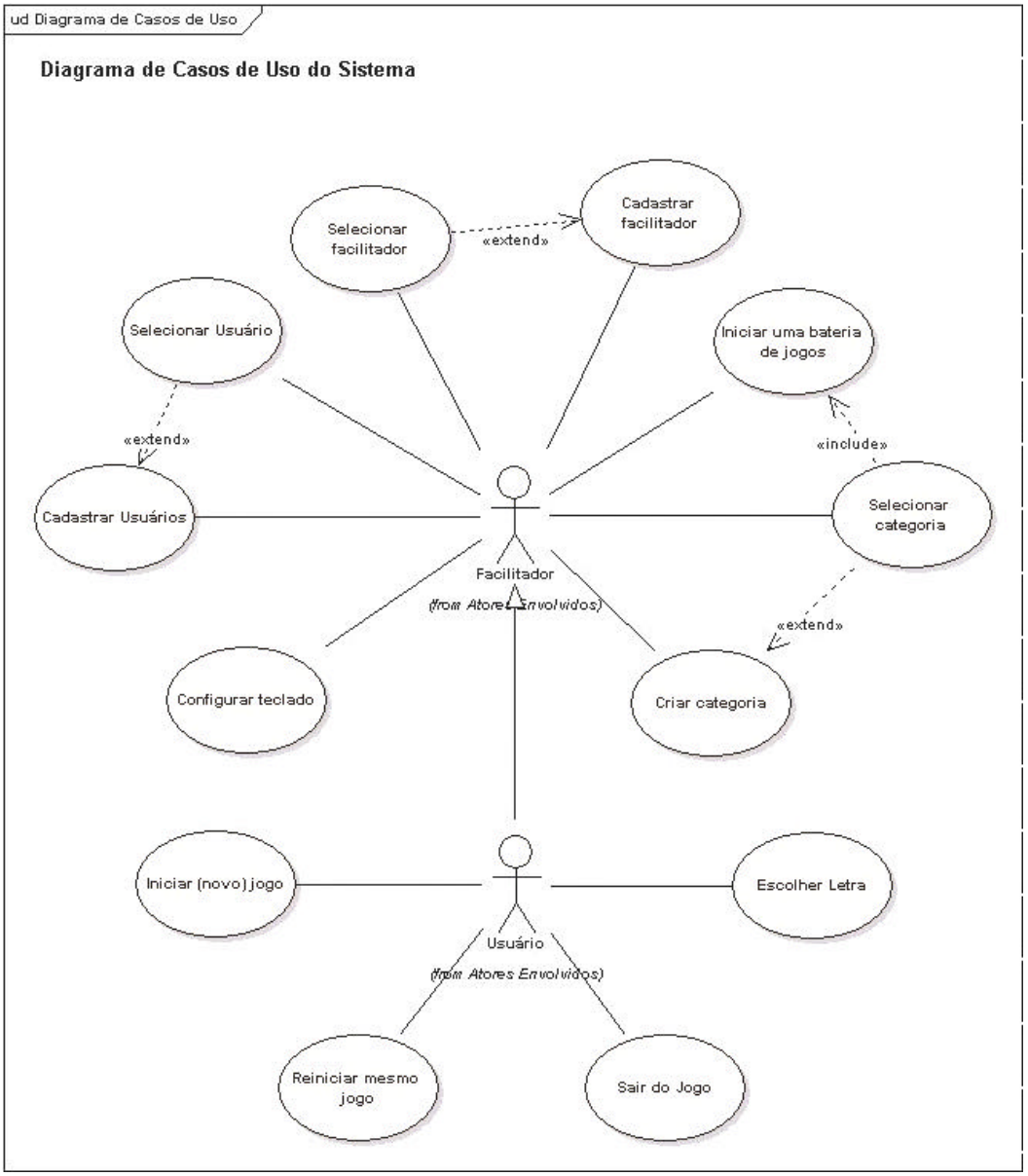


Figura 22. Diagrama de Casos de Uso

CU01_Cadastrar Usuários

Um facilitador pode cadastrar um ou mais usuários, com as seguintes informações: Nome do usuário e Facilitador responsável

CU02_Cadastrar Facilitador

Um facilitador pode cadastrar um ou mais facilitadores, inserindo o seu Nome e os respectivos usuários.

CU03_Configurar teclado

Este caso de uso define a configuração das seis grandes teclas no teclado de computador comum, indicando em quais teclas se basear.

CU04_Criar categoria

Uma categoria ou mais devem ser criadas para o cadastro das palavras a serem descobertas no jogo.

CU05_Ecolher letra

Este caso de uso descreve o ato do usuário selecionar uma letra através do teclado ergonomicamente alterado e o sistema observar se esta letra pertence a palavra e desabilitá-la do vocabulário

CU06_Iniciar (novo) jogo

Um jogo pode ser iniciado ou reiniciado em qualquer etapa deste. Como um jogo entende-se a descoberta de uma palavra por um usuário e seu facilitador.

CU07_Iniciar uma bateria de jogos

Este caso de uso deslumbra o ato de iniciar uma nova bateria de jogos, onde deve-se objetivar a seguida de descoberta de palavras escondidas.

CU08_Reiniciar mesmo jogo

Em qualquer momento o jogo em execução pode ser reiniciado, reabilitando as letras antes escolhidas, e limpando as letras já descobertas na palavra do jogo. Se for o caso, o timer é também, reiniciado.

CU09_Sair do jogo

Em qualquer momento do jogo o usuário tem a opção de desistir do jogo de forma fácil e rápida.

CU10_Selecionar usuário

Um usuário deve ser selecionado após a seleção do facilitador, para se dar o início do jogo.

CU11_Selecionar categoria

Uma categoria de palavras deve ser selecionada para o início do jogo, de onde será realizado o filtro das palavras de acordo com o optado pelo facilitador

CU12_Selecionar facilitador

Um facilitador pode escolher selecionar um facilitador que irá iniciar uma bateria de exercícios com determinado usuário.

É possível, a qualquer momento, alterar o facilitador do jogo, alterando juntamente o respectivo usuário.

4.1.4. Modelo Conceitual

Nesta etapa é demonstrado o Modelo Conceitual, que nasce a partir da abstração do domínio, utilizando como referência os estudos realizados anteriormente, além das regras e requisitos e os casos de uso já citados.

Esta abstração, com suas entidades, ações e relacionamentos é descrita formalmente conforme o diagrama mostrado a seguir.

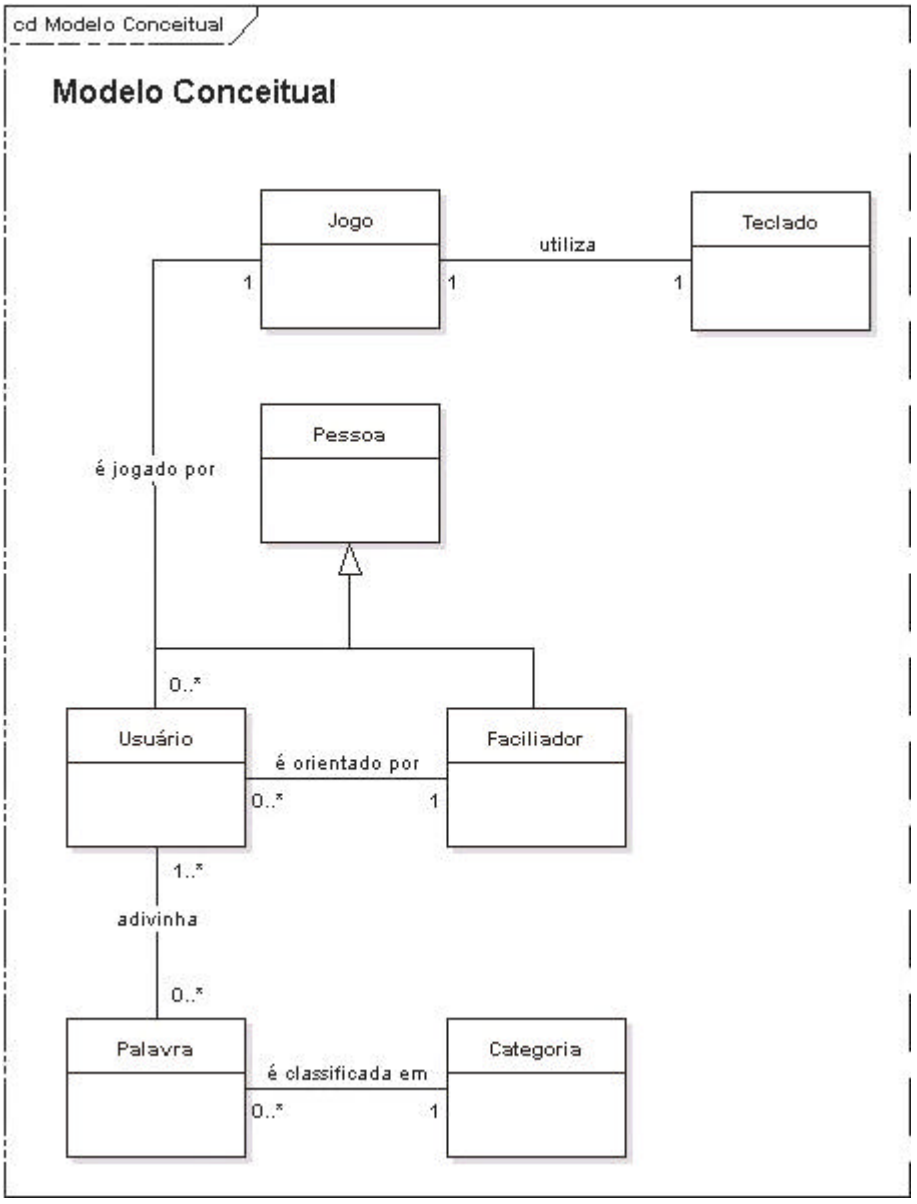


Figura 23. Diagrama de Modelo Conceitual

4.1.5. Modelo de Classes

O Diagrama de Classes apresentado a seguir demonstra todas as classes utilizadas pelo Sistema, constando uma descrição de suas funcionalidades e sua importância no contexto do software, indicando seus respectivos atributos e operações.

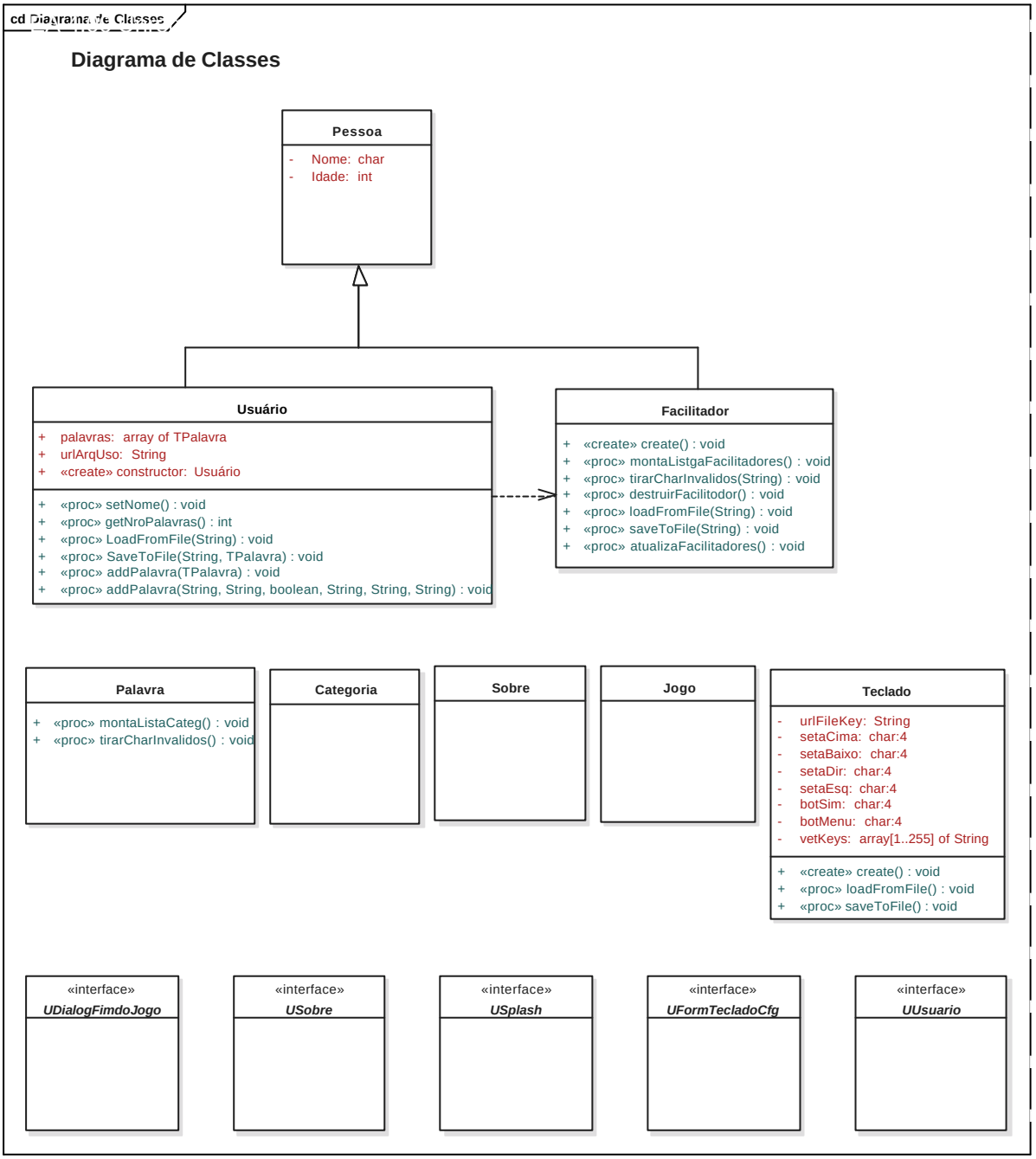


Figura 24. Diagrama de Classes

4.2. Adaptação de Hardware

No momento da concepção da idéia do software, no item anterior descrito, vislumbrou-se a utilização do mesmo com as pessoas portadoras de problemas motores e foi observada muita dificuldade de interface entre o usuário e a utilização do software.

A princípio seria utilizado um teclado e mouse comuns para a execução do programa, porém, diante da problemática motora dos usuários, sentiu-se a necessidade de adaptar um hardware dentro das necessidades destas pessoas para ser utilizado em conjunto do software também adaptado.

Para a adaptação deste hardware, respeitou-se um dos requisitos fundamentais deste projeto: o baixo custo. Desta forma, partimos da idéia já utilizada na dissertação de mestrado de Jonas Adão Pereira Soares e Edson Edir Dick.

Os autores citados criaram uma máscara que foi utilizada sobre o teclado comum, conforme especificado mais detalhadamente nos estudos preliminares, em adaptações de hardware.

Na adaptação de hardware desta proposta, retirou-se todas as teclas de um teclado comum, conforme ilustra a foto abaixo:



Figura 25. Teclado em fase de adaptação

O teclado adaptado terá seis grandes teclas, que serão teclas direcionais (que movem o foco para esquerda, direita, cima ou para baixo), tecla de MENU e tecla SIM.

As teclas direcionais servem para selecionar uma letra no alfabeto que é mostrado na tela. A tecla SIM serve para confirmar uma letra procurada pelo usuário a fim de acertar a palavra secreta e a tecla MENU chama o menu de opções, que tem a seguintes opções: Reiniciar palavra atual, reiniciar outra palavra e sair.

As teclas serão tampas cegas utilizadas em instalações elétricas medindo oito por treze centímetros, na cor gelo. Veja a figura abaixo.



Figura 26. Tampas cegas de instalações elétricas

Estas tampas cegas serão afixadas nas teclas que foram retiradas do teclado original, utilizando massa do tipo epóxi, conforme ilustra a figura abaixo:

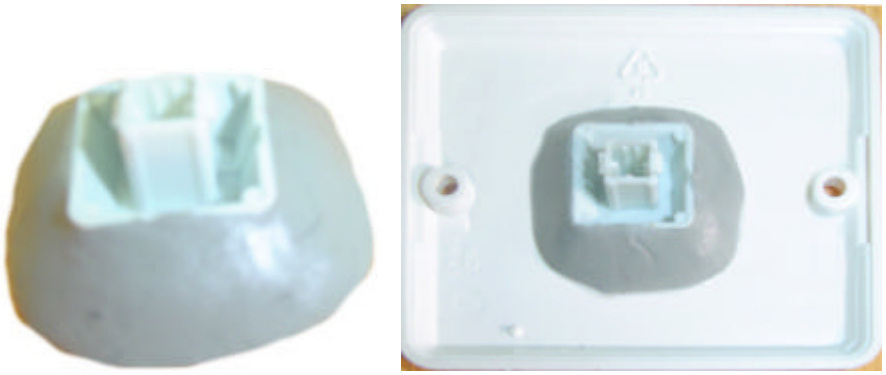


Figura 27. Visão inferior das teclas do teclado adaptado

O próximo passo é a impressão e posterior colagem das imagens que cada uma das teclas terão, que são as mesmas utilizadas no programa, para que seja melhor facilitado ao usuário.

A imagem abaixo demonstra as seis grandes teclas adaptadas no teclado que será utilizado como interface entre o usuário e o sistema adaptado.

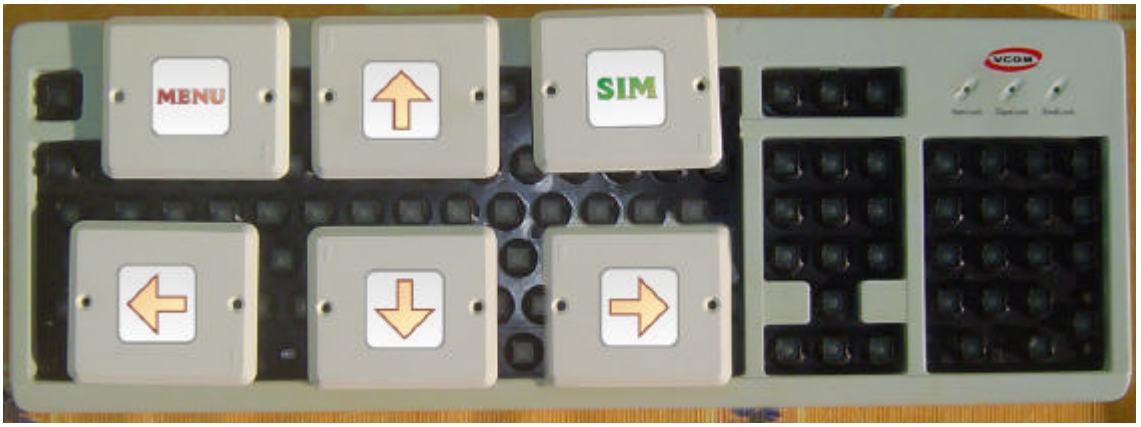


Figura 28. Teclado adaptado

As teclas acima serão configuradas pelo facilitador no software. Se o facilitador julgar melhor outra distribuição de teclas, basta que ele destaque as teclas, as reposicione e reconfigure o software de forma fácil e rápida, como mostra a figura abaixo.

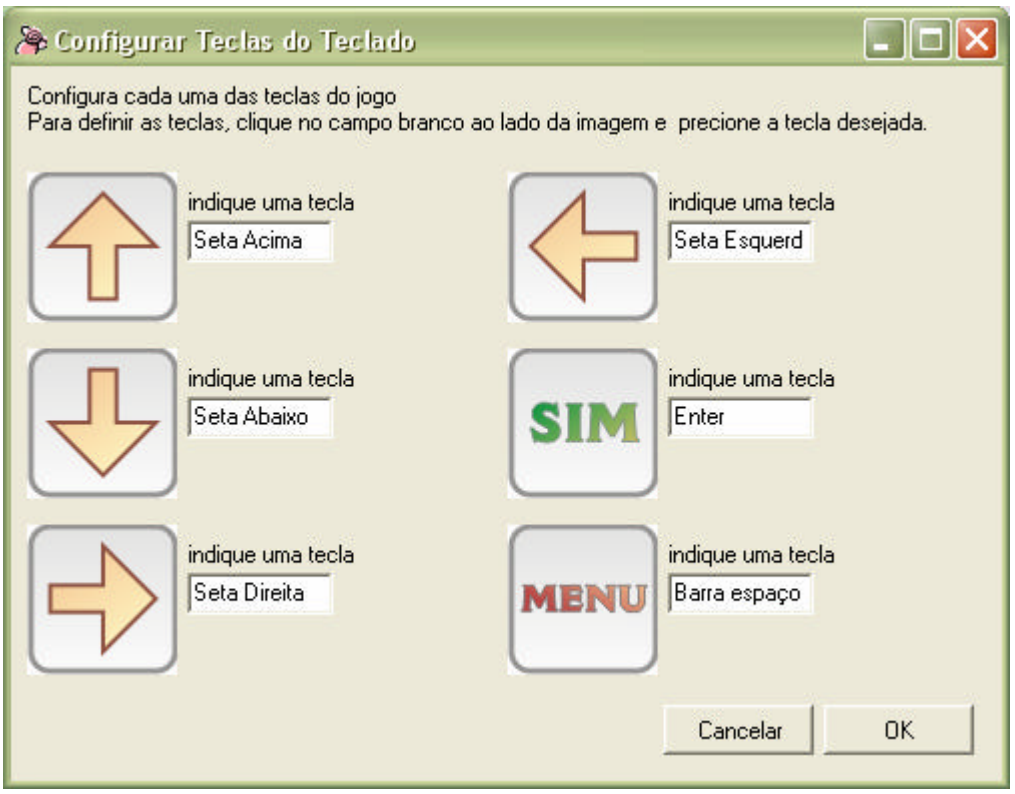


Figura 29. Configuração do teclado

A cada configuração realizada das teclas direcionais, o sistema memoriza tal configuração, de modo que a partir desta, todas as utilizações futuras tenham estas seis teclas como teclas reconhecidas pelo software, sem a necessidade de reconfiguração.

4.3. Utilização do Jardim das Palavras Secretas

Nesta etapa poderá ser vislumbrada a utilização do sistema conforme pode ser acompanhado pelo Diagrama de Atividades Principal, mostrado etapa da modelagem do software apontado como a principal atividade na utilização do Jardim das Palavras Secretas.

Logo que iniciada a utilização do sistema, é necessário que o facilitador se identifique. Caso seja um novo facilitador, este pode ser cadastrado através do botão “novo facilitador”, conforme tela de cadastros demonstrada mais adiante.

Após a identificação do facilitador, este deve selecionar o usuário que utilizará o jogo, e caso também não esteja cadastrado, será demonstrado cadastro futuramente.

Além do nome facilitador e do usuário, o facilitador deve selecionar, no último combobox da tela, a categoria de palavras a ser utilizada durante o jogo, conforme o andamento de aprendizagem do usuário, que a partir desta etapa poderá utilizar o programa sozinho, apenas com o auxílio quando necessário, do facilitador.

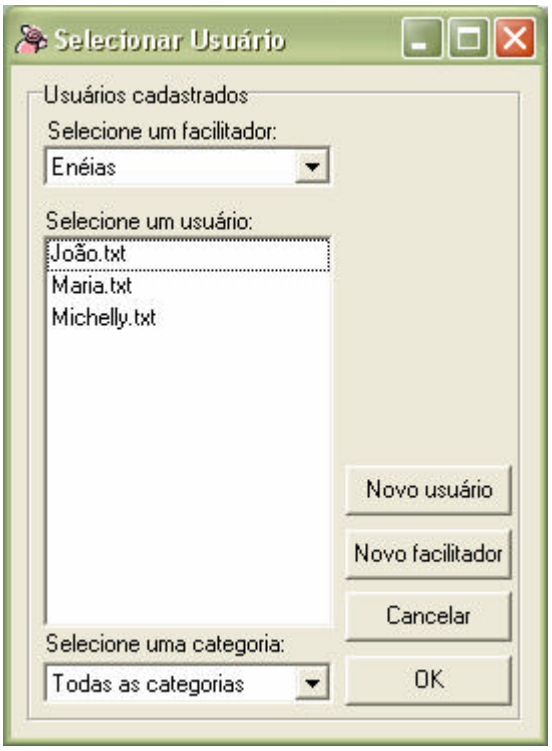


Figura 30. Tela de escolha de facilitador, usuário e categoria

Ao selecionar os três registros, e clicado em OK (até aqui com a utilização do mouse pelo facilitador), o sistema apresenta a primeira palavra a ser descoberta pelo usuário.

A partir desta etapa, o usuário passa a dominar a utilização do sistema, com o interfaceamento do teclado adaptado, onde a pessoa deve usar as teclas de navegação para selecionar a letra desejada.

Para confirmar a letra que deseja verificar se esta consta na palavra secreta, o usuário deve apertar a tecla SIM, na tela mostrada a seguir.

Figura 31. Tela com alfabeto e palavra secreta para adivinhação

O usuário segue indicando as letras até que o jogo acabe. Isto ocorre quando ele acerta a palavra ou quando ele erra dez letras.

Ao acertar a palavra antes de obter dez erros, o sistema mostra uma tela de parabéns e reafirma a palavra secreta, como pode ser visto na figura abaixo.



Figura 32. Tela de parabenização pela palavra acertada

A cada letra errada, o usuário perde uma flor do vaso e após a nona letra errada, o vaso também desaparece, aparecendo então mensagem de erro, como pode ser observado a seguir.

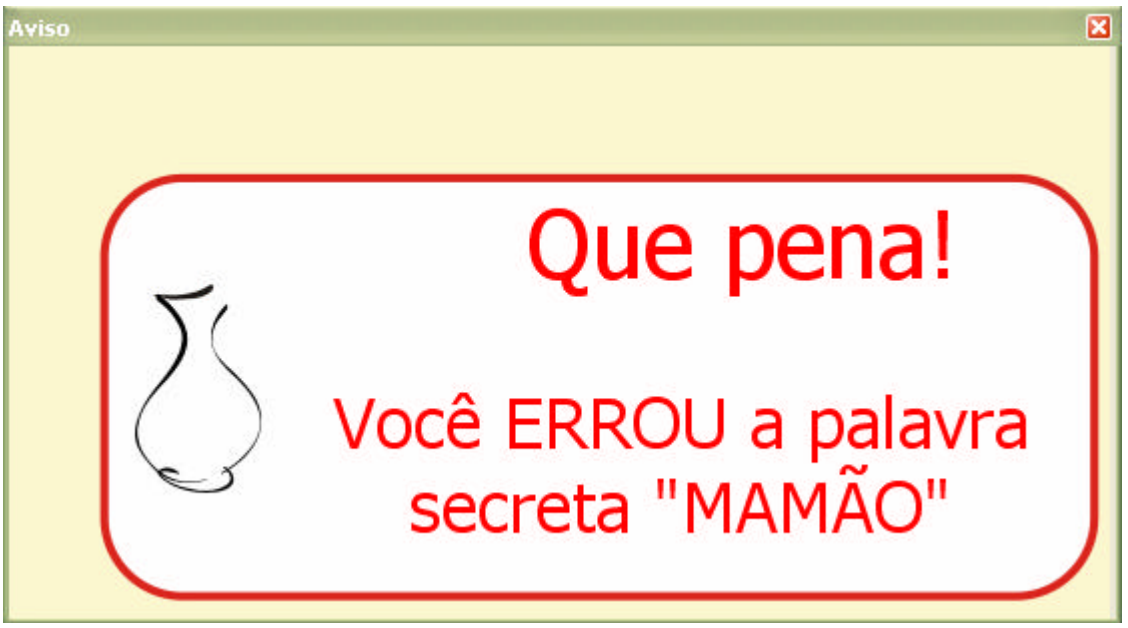


Figura 33. Tela que demonstra a palavra secreta não acertada

4.3.1. Cadastros e Configurações

O facilitador é o utilizador do sistema responsável por realizar os cadastros de facilitador, usuários, palavras e também de configurar o teclado.

Cadastros de facilitador

O facilitador deve cadastrar um (si mesmo) ou mais facilitadores para interagir com o sistema. Na tela de cadastro de facilitadores é possível, além de cadastrar um novo facilitador, também gerenciar os usuários dos facilitadores já cadastrados, como mostra a próxima tela.

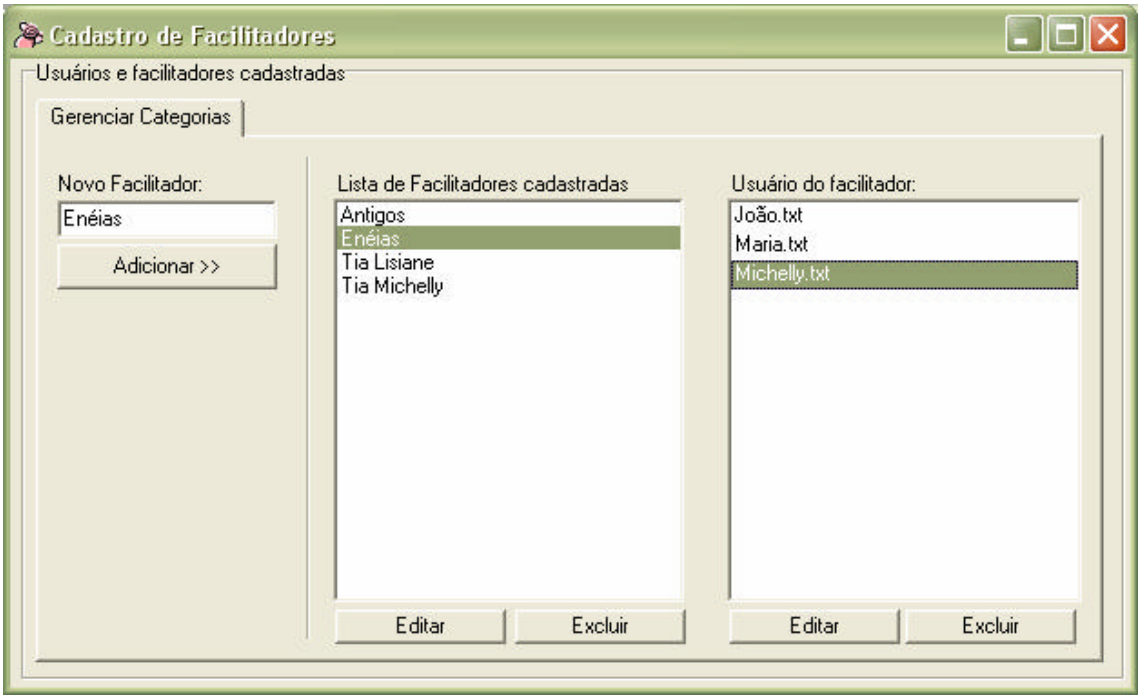


Figura 34. Gerenciamento de Facilitadores

Cadastros de usuário

O sistema prevê o cadastro de usuário, por intermédio do facilitador. Neste cadastro, será informado o nome do novo usuário, bem como qual o facilitador que será responsável por ele, pois como regra pré-definida, cada usuário será obrigatoriamente gerenciado por um facilitador.

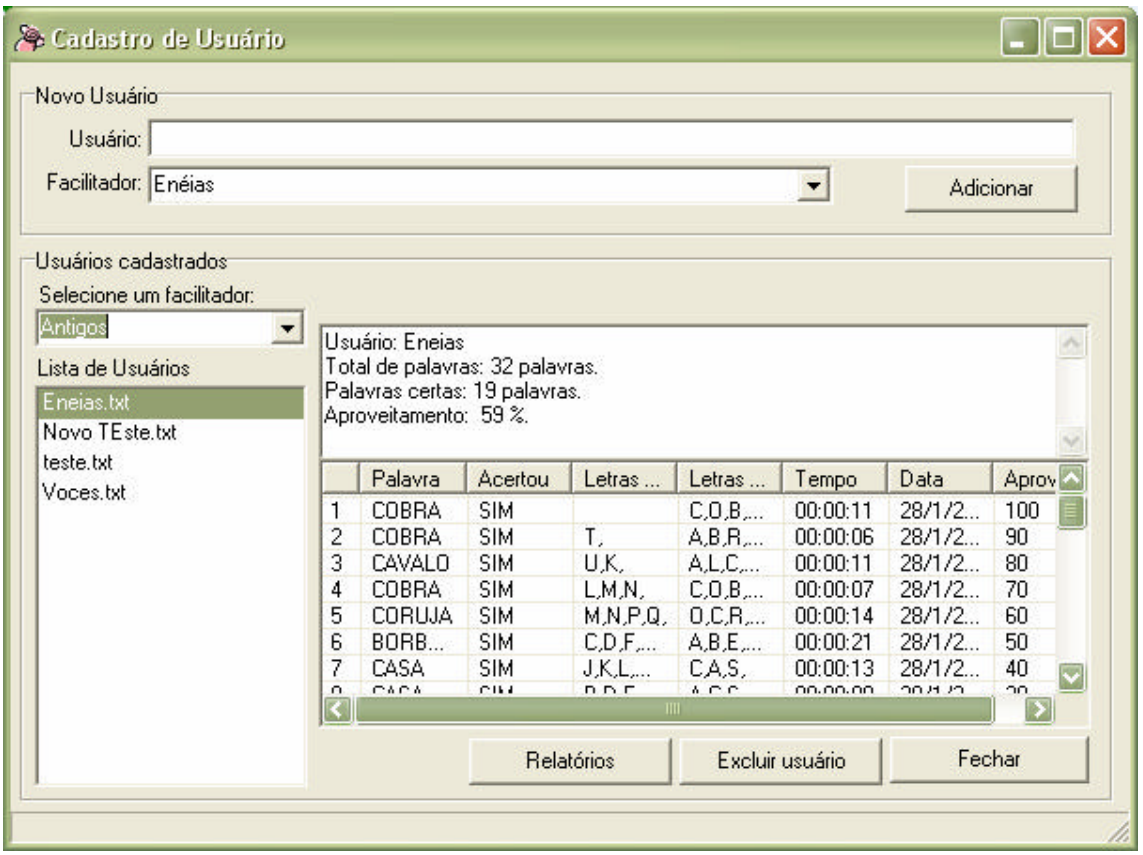


Figura 35. Cadastro de Usuários

Na tela de cadastro do usuário é possível, além de cadastrar um novo usuário, ver o histórico de palavras de cada um dos usuários já cadastrados e todas as características de cada jogo realizado, que pode ser visualizado também no menu de relatórios.

Cadastros de palavras e categorias

Uma palavra pode ser cadastrada através da tela mostrada logo a seguir, declarando todas as letras da palavra, a imagem relacionada e a categoria que a mesma pertence.

Caso seja necessário o cadastramento da palavra em uma nova categoria, esta pode ser criada facilmente e na mesma tela.

Como pode ser visualizado, o gerenciamento das categorias é realizado neste momento, podendo ser visível todas as palavras de cada uma das categorias já cadastradas no sistema.

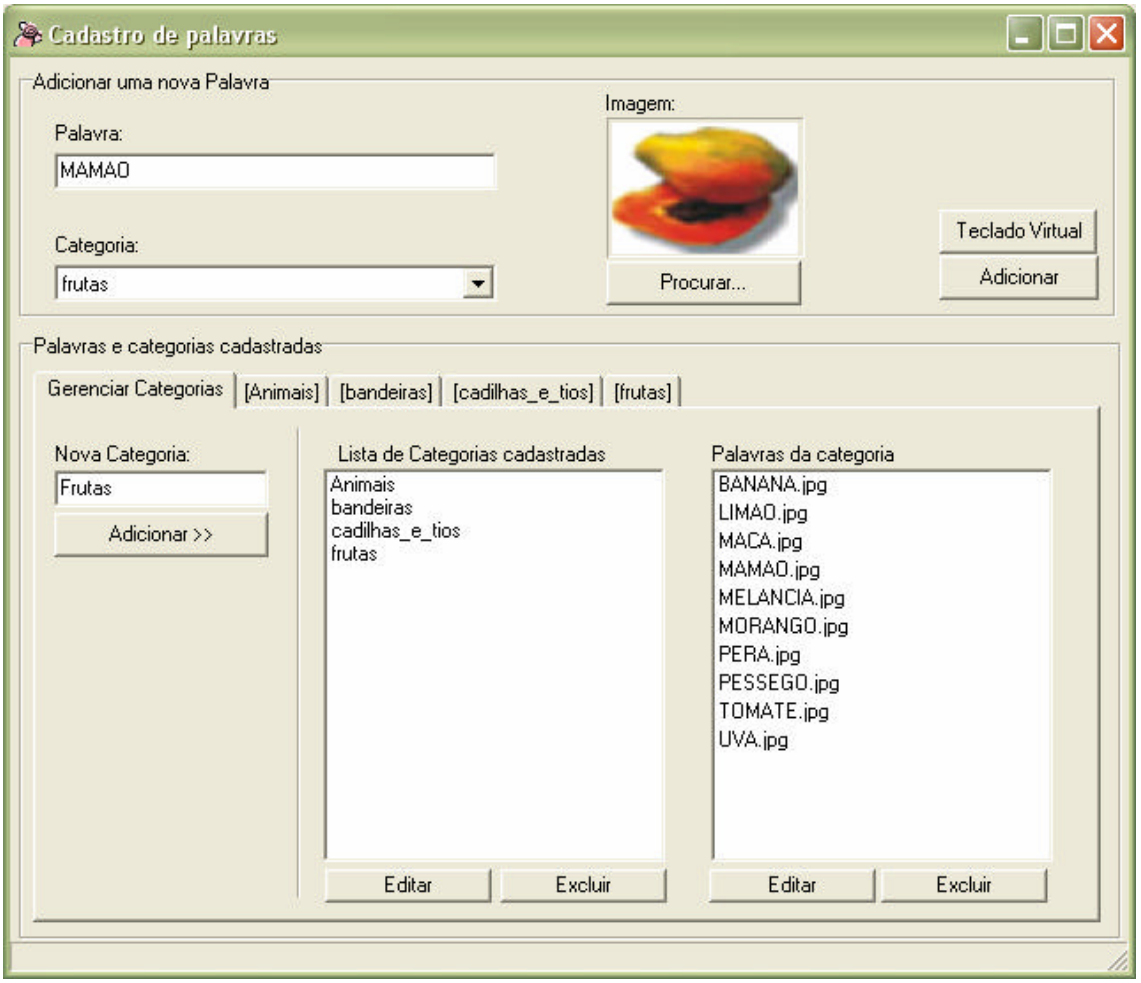


Figura 36. Tela do cadastro de palavras e categorias

Configuração do teclado

Antes de iniciar o uso do programa, o facilitador deve configurar o teclado, que foi previamente adaptado, na tela exposta a seguir. Este procedimento é necessário apenas uma vez, pois as configurações realizadas são salvas no computador.

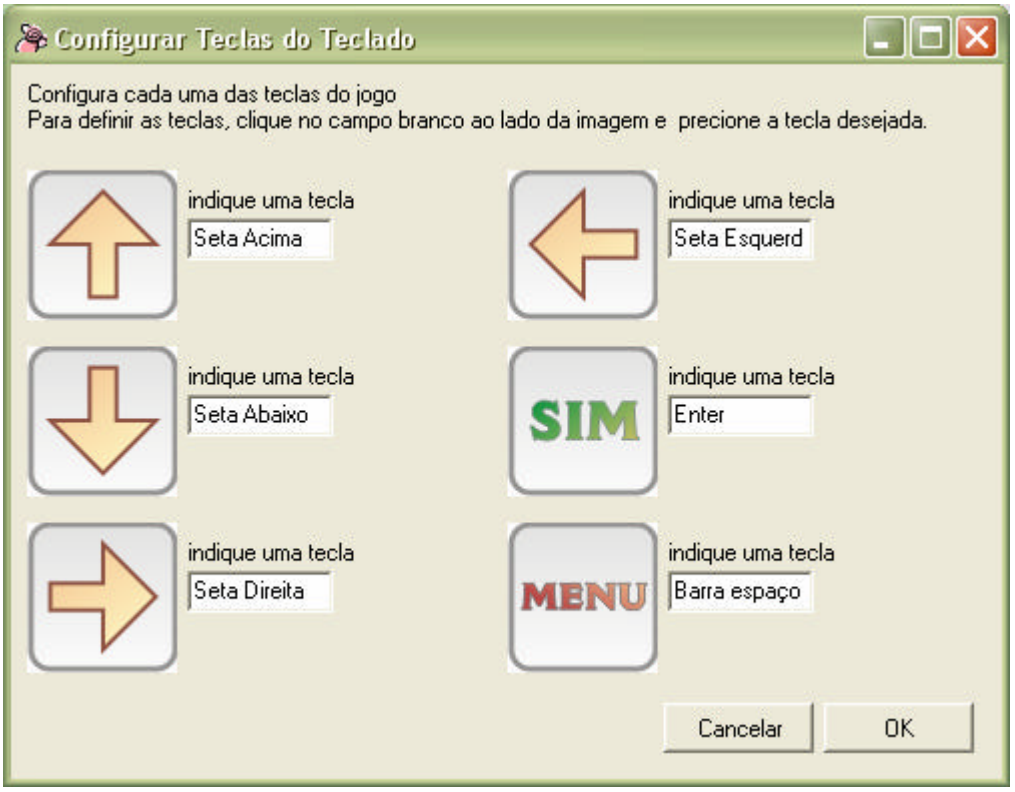


Figura 37. Configuração do Teclado

4.3.2. Relatórios

O facilitador pode ver e imprimir relatório de cada um dos seus usuários e dos usuários dos demais facilitadores, como é possível observar a seguir.

O relatório contém as seguintes informações:

- palavras secretas jogadas pelo usuário;
- acerto ou não da palavra secreta jogada pelo usuário;
- todas as letras acertadas de cada palavra secreta jogada;
- todas as letras erradas de cada palavra secreta jogada;
- data dos jogos;
- tempo decorrido em cada jogo

falar de ver as letras q o usuario toca sem querer ao lado e não na certa



Figura 38. Relatório por usuário

5. Conclusão

Baseado nos testes aplicados ao software, na experiência em programa-lo e na adaptação do teclado, percebemos que cumprimos os objetivos propostos no escopo do projeto: adaptar hardware e projetar um software adequado às pessoas com limitações motoras e capacidade cognitiva preservada.

Em um primeiro momento procurou-se, através da internet principalmente, soluções já existentes que atendessem ao perfil do usuário que já descrito anteriormente, para podermos propor uma solução inédita e de baixo custo. Foi exatamente o que ocorreu. Desenvolvemos uma proposta que ainda não havia no mercado: uma atividade lúdica inspirada no jogo forca com o intuito de poder ajudar as pessoas que desta solução necessitassem.

Constatamos, de fato, que podemos contribuir para a vida de algumas pessoas, que puderam ter, além do entretenimento, uma opção de aprimorar sua capacidade de raciocínio e memória. Além disto, o convívio familiar foi aumentado, pois o usuário não tem condições de iniciar o aplicativo, ele necessita de um facilitador, geralmente algum membro familiar.

Muitos esforços já foram feitos neste tema, mas percebemos que ainda há muito que fazer.

6. Trabalhos Futuros

No projeto original, exposto neste trabalho, o objetivo da solução proposta era auxiliar o desenvolvimento intelectual de pessoas com limitações físicas nos membros superiores e capacidade cognitiva preservada, através de expansão de vocabulário e aprendizado de novas palavras.

A primeira proposta de trabalho futuro é aprimorar a parte de relatórios, criando heurísticas para predizer se o usuário está errando letras por não saber qual a palavra secreta ou se por dificuldade de uso do teclado adaptado. Esta análise já pode ser feita, mas de forma manual. O objetivo é tornar este processo automático.

Uma outra opção de uso do software é o aprendizado de outros idiomas, mostrando a imagem palavra que a ser adivinhada e o usuário tendo que tentar descobrir as letras que formam esta palavra.

Além dos aprimoramentos das funcionalidades já desenvolvidas, é possível adaptar outras atividades lúdicas ao teclado, como por exemplo, o jogo da memória (acha par). Esta atividade estimula a memória e o pensamento lógico. É possível propor níveis de dificuldade, como por exemplo, os pares serem imagem-imagem ou imagem-palavra escrita. Assim, todo o cadastro de palavras, usuários e facilitadores seriam reutilizados.

7. Referências Bibliográficas

http://www.itaucultural.org.br/programacao/pop_revista/pop_uniban3.htm

http://www.click.com.br

http://www.msd-brazil.com/msd43/m_manual/mm_sec23_270.htm

http://www.stlouischildrens.org/default.aspx?tabid=96&acn=view&aid=1571

http://www.drgilberto.com/pc.html

http://lsm.dei.uc.pt/ribie/docfiles/txt200342415851240.PDF

http://www.acessibilidade.net/at/kit/index.htm

http://educacao.cnotinfor.pt/index.php?lng=pt&pag=153&id=0

http://anditec.pt/produtos/product_info.php/cPath/27_35/products_id/42

http://educacao.cnotinfor.pt/index.php?lng=pt&pag=453&id=0

http://www.clik.com.br/mj_01.html

http://www.niee.ufrgs.br/ribie98/CONG_1996/CONGRESSO_HTML/145/FALAS.HTML

http://infoesp.vilabol.uol.com.br/recursos/recurso2.htm

http://infoesp.vilabol.uol.com.br/recursos/adapta2.htm

http://www.saci.org.br/?modulo=akemi&parametro=3847

http://www.lagares.org

http://www.defnet.org.br

http://galileu.globo.com/edic/98/saude3.htm

http://lsm.dei.uc.pt/ribie/docfiles/txt20034241541111.pdf

http://www.unilinhares.net/Noticias/Banco_Noticias_Visualizar.asp?c=562

http://www.proex.pucminas.br/sociedadeinclusiva/arquivos/Sociedade%20Inclusiva%202%20art%2004.pdf

http://galileu.globo.com/edic/98/saude2.htm

SILVEIRA, M. Selbach. **Aplicações de técnicas de Inteligência Artificial à comunicação alternativa e aumentativa**. (Dissertação de mestrado). Porto Alegre: CPGCC/ UFRGS, 1996. 113p.

HALL, Calvin Springer; LINDZEY, Gardner. Teorias da Personalidade. Traducao de Maria Cristina Machado Kupfer. 18.ed. Sao Paulo: EPU, 1984. v.1. 159 p. il. ISBN 85-12-63310-7

SOARES, Jonas Adão Pereira. **Desenvolvimento de uma ajuda técnica para o processo de comunicação aumentativa**. Dissertação (Mestrado em ciências da computação) UFSC, Florianópolis, 2002.

DICK, Edson Edir. **Avaliação de uma ajuda técnica informática para o processo de comunicação alternativa de uma criança com lesão no cérebro**. Dissertação (Mestrado em ciências da computação) UFSC, Florianópolis, 2002.

OMG - Unified Modeling Language: Superstructure. Versão 2.0, OMG, agosto de 2005.

http://info.abril.com.br/corporate/edicoes/33/entrevista/conteudo_138501.shtml

Shneiderman, B. Designing the User Interface, Addison- Wesley, 1987.

http://netuno.icmc.usp.br/cdrom2002/Artigos/WTDD_AIC08.pdf

http://www.lsc.ufsc.br/~edla/ine5624/materiais/iso9241-11F2.pdf

http://www.acessibilidade.net/

http://www.unifacs.br/ihc/downloads/doc/luzzardi/Tese_de_Doutorado_Cap3.doc

http://www.docsystems.med.br/puericlt/anoxneol.htm

http://www.labiutil.inf.ufsc.br/ergolist/

Requisitos Ergonômicos para Trabalho de Escritórios com Computadores Parte 11 – Orientações sobre Usabilidade - 08/2002 NBR 9241-11

SEVERINO, Antônio Joaquim. Metodologia do trabalho científico. 21. ed. rev. ampl. São Paulo: Cortez, 2000.

BRASIL. Constituição da República Federativa do Brasil. 8. ed. São Paulo: Revista dos Tribunais, 2003. RT Legislação

Pasquarelli, Maria Luiza Rigo Normas para a apresentação de trabalhos acadêmicos [ABNT/NBR-14724] / Maria Luiza Rigo Pasquarelli - - 2a ed. Osasco: [EDIFIEO], 2004.

http://www.ietec.com.br/ietec/techoje/techo je/tecnologiadainformacao/
Artigo na revista tec hoje: Qualidade na gestão de projeto de desenvolvimento de software - Patrícia Baraúna Magno (Graduada em ciência da computação, pós-graduada em Gestão de Projetos pelo IETEC)

http://pt.wikipedia.org/wiki/Processo_de_Engenharia_de_Requisitos"

http://pt.wikipedia.org/wiki/ISO_12207

MELISSA MARI KATO A Especificação de Requisitos no Processo de Desenvolvimento de Software (Trabalho de conclusão de curso obrigatório do Curso de Graduação em Ciência da Computação da Universidade Estadual de Londrina) 2004

Paper: Gerenciamento de Projetos na Engenharia de Software. Mauro Sotille, PMP, ITIL

Artigo: Gerencia de requisitos: O principal problema dos projetos de software. José Roberto Blaschek. Na revista developer´s Magazine (sem data – aprox 2000)

http://pt.wikipedia.org/wiki/Diagrama_de_classes

http://www1.serpro.gov.br/PQGF2004/3.htm - relatorio de gestao serpro 2004: cidadãos e sociedades

Edna Pacheco Zanlorenci e Robert Carlisle Burnett - A Abordagem da Qualidade e a Engenharia de Requisitos. Abril 2003

BOEHM, B. Software model conflicts and how to avoid them. In: SBES´98 XII SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE, 12., 1998, Maringá. Tutorial... Maringá: SBC, 1998. 80 p. Disponível em: <http://www.sunset.usc.edu/>.

Alcides Calsavara - PUC-PR, Cristina Ângela Filipack Machado – GPT, Sheila dos Santos Reinehr – BANESTADO, Robert Carlisle Burnett - PUC-PR - Aderência do RUP à norma NBR ISO/IEC 12207

http://www.ucb.br/ucbtic/mgcti/paginapessoalprof/Nicolas/Disciplinas/UML/uml.pdf

OMG - Unified Modeling Language: Superstructure. Versão 2.0, OMG, agosto de 2005.

http://info.abril.com.br/corporate/edicoes/33/entrevista/conteudo_138501.shtml

Shneiderman, B. Designing the User Interface, Addison- Wesley, 1987.

http://netuno.icmc.usp.br/cdrom2002/Artigos/WTDD_AIC08.pdf

http://www.docsystems.med.br/puericlt/anoxneol.htm

Anexo 1 – Código Fonte

```
unit UDialogsFimJogo;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 ExtCtrls, StdCtrls, UTeclado;

type
 TFormDialog = class(TForm)
   ImageAcertou: TImage;
   LabelTitulo: TLabel;
   LabelDesc: TLabel;
   ImageErrou: TImage;
   procedure FormKeyDown(Sender: TObject; var Key: Word;
     Shift: TShiftState);
   procedure ImageAcertouClick(Sender: TObject);
 private
   { Private declarations }
 public
   palavra: string;
   procedure dialogAcertou(pal:string);
   procedure dialogErrou(pal:string);
 end;

var
 FormDialog: TFormDialog;

implementation

{$R *.DFM}

procedure TFormDialog.dialogAcertou(pal: string);
begin
  ImageAcertou.Visible:= true;
  ImageErrou.Visible:= false;
  LabelTitulo.Caption:= 'Parabens!';
  LabelTitulo.Font.Color:= clGreen;
  LabelDesc.Caption:= 'Você ACERTOU a palavra secreta "'+pal+'"';
  LabelDesc.Font.Color:= clGreen;
end;

procedure TFormDialog.dialogErrou(pal: string);
begin
  ImageAcertou.Visible:= false;
  ImageErrou.Visible:= true;
  LabelTitulo.Caption:= 'Que pena!';
  LabelTitulo.Font.Color:= clRed;
  LabelDesc.Caption:= 'Você ERROU a palavra secreta "'+pal+'"';
  LabelDesc.Font.Color:= clRed;
end;
```

```
procedure TFormDialog.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  self.close;
end;

procedure TFormDialog.ImageAcertouClick(Sender: TObject);
begin
  self.close;
end;

end.
```

```
unit UFormBackground;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 UPrincipal, ExtCtrls, UTeclado, StdCtrls;

type
 TForm1 = class(TForm)
   PHelp: TPanel;
   setaCima: TImage;
   SetaEsq: TImage;
   setaBaixo: TImage;
   botSim: TImage;
   SetaDir: TImage;
   botNao: TImage;
   Label1: TLabel;
   Label2: TLabel;
   Label3: TLabel;
   Label4: TLabel;
   Panel1: TPanel;
   Label5: TLabel;
   procedure FormActivate(Sender: TObject);
   procedure FormPaint(Sender: TObject);
   procedure FormKeyDown(Sender: TObject; var Key: Word;
     Shift: TShiftState);
   procedure FormClick(Sender: TObject);
 private
   teclado: TTeclado;
 public
   procedure ChamaFormPrincipal;
 end;

var
 Form1: TForm1;

implementation
```

```
{$R *.DFM}

procedure TForm1.FormActivate(Sender: TObject);
begin
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas\acima.bmp') then
     setaCima.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas\acima.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas\abaixo.bmp') then
     setaBaixo.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas\abaixo.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas\direita.bmp') then
     setaDir.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas\direita.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas\esquerda.bmp') then
     setaEsq.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas\esquerda.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas\SIM.bmp') then
     botSim.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas\SIM.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas\NAO.bmp') then
     botNao.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas\NAO.bmp');

  teclado:= TTeclado.create(ExtractFileDir(Application.ExeName)+'\config\teclas.ini');


end;

procedure TForm1.FormPaint(Sender: TObject);
begin
  PHelp.Left:= round(self.Width/2 -PHelp.Width/2);
  PHelp.Top:=  round(self.Height/2 -PHelp.Height/2);
end;

procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  ChamaFormPrincipal;
end;

procedure TForm1.ChamaFormPrincipal;
begin
  FormPrincipal.ShowModal;
  Self.Close;
end;

procedure TForm1.FormClick(Sender: TObject);
begin
  ChamaFormPrincipal;
end;

end.
```

```
unit UFormFacilitadores;

interface

uses
```

```
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 StdCtrls, FileCtrl, ExtCtrls, ComCtrls, Menus, UFuncoesUteis;

type
 TFormFacilitadores = class(TForm)
  GroupBox2: TGroupBox;
  PageControlPalavras: TPageControl;
  TabGerFac: TTabSheet;
  Label4: TLabel;
  Label5: TLabel;
  Label6: TLabel;
  Bevel2: TBevel;
  EditNovoFac: TEdit;
  ButtonAddCAteg: TButton;
  LBFacilitador: TListBox;
  ListaUsuario: TFileListBox;
  ButtonDelFac: TButton;
  ButtonDelUsu: TButton;
  ListFacitiladores: TFileListBox;
  PopupFac: TPopupMenu;
  EditarFacilitador1: TMenuItem;
  ExcluirFacilitador1: TMenuItem;
  PopupUsu: TPopupMenu;
  EditarUsuario1: TMenuItem;
  ExcluirUsuario1: TMenuItem;
  ButtonEditFac: TButton;
  ButtonEditUsu: TButton;
  procedure FormActivate(Sender: TObject);
  procedure LBFacilitadorMouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);
  procedure FormCreate(Sender: TObject);
  procedure ButtonDelFacClick(Sender: TObject);
  procedure PopupFacPopup(Sender: TObject);
  procedure PopupUsuPopup(Sender: TObject);
  procedure ButtonDelUsuClick(Sender: TObject);
  procedure EditarFacilitador1Click(Sender: TObject);
  procedure ExcluirFacilitador1Click(Sender: TObject);
  procedure EditarUsuario1Click(Sender: TObject);
  procedure ExcluirUsuario1Click(Sender: TObject);
  procedure ButtonEditFacClick(Sender: TObject);
  procedure ButtonEditUsuClick(Sender: TObject);
  procedure ButtonAddCAtegClick(Sender: TObject);
 private
  dirUsu: string;
  procedure tirarCharInvalidos(var texto: string);
 public
  procedure montaListaFacilitadores;
 end;

var
 FormFacilitadores: TFormFacilitadores;

implementation

{$R *.DFM}
```

```
procedure TFormFacilitadores.FormCreate(Sender: TObject);
begin
  dirUsu:= ExtractFileDir(Application.ExeName)+'\usuarios';
  ListFacitiladores.Directory:= dirUsu;
end;

procedure TFormFacilitadores.FormActivate(Sender: TObject);
begin
  montaListaFacilitadores;
end;

procedure TFormFacilitadores.LBFacilitadorMouseDown(Sender: TObject;
 Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ListaUsuario.Directory:= dirUsu+'\'+LBFacilitador.Items[LBFacilitador.ItemIndex];
  ListaUsuario.Update;
end;



procedure TFormFacilitadores.ButtonDelFacClick(Sender: TObject);
var delDir: string;
   i: integer;
begin
  for i := 0 to (LBFacilitador.Items.Count - 1) do begin
    try
      if LBFacilitador.Selected[i] then  begin
        deldir:= dirUsu+'\'+LBFacilitador.Items.Strings[i];
        if DirectoryExists(deldir) then begin
          if Application.MessageBox(PChar('Deseja realmente excluir o FACILITADOR
"'+LBFacilitador.Items.Strings[i]+'" ?'+#13+'Este facilitador possui '+IntToStr(LBFacilitador.Items.Count)+'
usuário(s) que serão EXCLUIDOS junto com o facilitador'+#13+'Esta operação não poderá ser
desfeita.'),PChar('Atenção'),MB_YESNO) = IDYES then begin
            DeleteDir(Self.Handle,deldir);
          end;
        end;
      end;
    except
    end;
  end;
  montaListaFacilitadores;
  ListaUsuario.Clear;
end;

procedure TFormFacilitadores.montaListaFacilitadores;
var i: integer;
  nomeDir: string;
begin
  ListFacitiladores.Update;
  LBFacilitador.Clear;
  for i:= 2 to ListFacitiladores.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir:= copy(ListFacitiladores.Items.Strings[i],2,length(ListFacitiladores.Items.Strings[i])-2);
    LBFacilitador.Items.Add(nomeDir);
  end;
```

```
end;

procedure TFormFacilitadores.PopupFacPopup(Sender: TObject);
begin
  EditarFacilitador1.Caption:= 'Editar Facilitador';
  ExcluirFacilitador1.Caption:= 'Excluir Facilitador';
  EditarFacilitador1.Enabled:= LBFacilitador.ItemIndex >=0 ;
  ExcluirFacilitador1.Enabled:= LBFacilitador.ItemIndex >=0 ;
  if EditarFacilitador1.Enabled then begin
    ExcluirFacilitador1.Caption:= 'Excluir Facilitador
('+LBFacilitador.Items.Strings[LBFacilitador.ItemIndex]+')';
    EditarFacilitador1.Caption:= 'Editar Facilitador ('+LBFacilitador.Items.Strings[LBFacilitador.ItemIndex]+')';
  end;
end;

procedure TFormFacilitadores.ButtonDelUsuClick(Sender: TObject);
var nome: string;
   i: integer;
begin
  for i := 0 to (ListaUsuario.Items.Count - 1) do begin
    try
      if ListaUsuario.Selected[i] then  begin
        nome:= ExtractFileName(ListaUsuario.FileName);
        nome:= copy(nome,0,length(nome)-4);
        if Application.MessageBox(PChar('Deseja realmente excluir a palavra "'+nome+'"
?'),PChar('Atenção'),MB_YESNO) = IDYES then begin
          DeleteFile(ListaUsuario.Items.Strings[i]);

        end;
      end;
    except
    end;
  end;
  ListaUsuario.Update;
  ListaUsuario.Refresh;
  ListaUsuario.Repaint;
end;

procedure TFormFacilitadores.PopupUsuPopup(Sender: TObject);
begin
  EditarUsuario1.Caption:= 'Editar Usuário';
  ExcluirUsuario1.Caption:= 'Excluir Usuário';
  EditarUsuario1.Enabled:= ListaUsuario.ItemIndex >=0 ;
  ExcluirUsuario1.Enabled:= ListaUsuario.ItemIndex >=0 ;
  if EditarUsuario1.Enabled then begin
    ExcluirUsuario1.Caption:= 'Excluir Usuário ('+ListaUsuario.Items.Strings[ListaUsuario.ItemIndex]+')';
    EditarUsuario1.Caption:= 'Editar Usuário ('+ListaUsuario.Items.Strings[ListaUsuario.ItemIndex]+')';
  end;
end;

procedure TFormFacilitadores.tirarCharInvalidos(var texto: string);
begin
  texto:= SubstituirCaracteres(texto,'/','-');
  texto:= SubstituirCaracteres(texto,'\','-');
  texto:= SubstituirCaracteres(texto,':','-');
```

```
  texto:= SubstituirCaracteres(texto,'*','-');
  texto:= SubstituirCaracteres(texto,'?',' -');
  texto:= SubstituirCaracteres(texto,'<','-');
  texto:= SubstituirCaracteres(texto,'>','-');
  texto:= SubstituirCaracteres(texto,'|','-');
end;

procedure TFormFacilitadores.EditarFacilitador1Click(Sender: TObject);
begin
  ButtonEditFac.Click;
end;



procedure TFormFacilitadores.ExcluirFacilitador1Click(Sender: TObject);
begin
  ButtonDelFac.Click;
end;

procedure TFormFacilitadores.EditarUsuario1Click(Sender: TObject);
begin
  ButtonEditUsu.Click;
end;

procedure TFormFacilitadores.ExcluirUsuario1Click(Sender: TObject);
begin
  ButtonDelUsu.Click;
end;

procedure TFormFacilitadores.ButtonEditFacClick(Sender: TObject);
var tex: string;
   urlFac, urlFacNovo: string;
begin

  urlFac:= PChar(dirUsu+'\'+LBFacilitador.Items.Strings[LBFacilitador.ItemIndex]);
  tex:= InputBox('Editar Categoria','Digite um nome para a nova
categoria',LBFacilitador.Items.Strings[LBFacilitador.ItemIndex]);
  //tex:= InputBox('Editar Categoria','Digite um nome para a nova categoria',urlFac+' | '+dirUsu+'\'+tex);

  if tex <> '' then begin
    tirarCharInvalidos(tex);
    urlFacNovo:=PChar(dirUsu+'\'+tex+'\');

    if DirectoryExists(urlFacNovo)then
      MessageBox(0,PChar('O FACILITADOR "'+tex+'" já existe.'+#13+'Escolha outro nome'),'Erro ao
renomear',mb_ok)
    else begin
      CreateDir(urlFacNovo);
      CopiaConteudoDir(urlFac+'\*.*',urlFacNovo);
      DeleteDir(Self.Handle,urlFac);
      LBFacilitador.Update;
      montaListaFacilitadores;
    end;
  end;
```

```
end;

procedure TFormFacilitadores.ButtonEditUsuClick(Sender: TObject);
var tex, ext, auxS: string;
begin
  ext:= ExtractFileExt(ListaUsuario.Items.Strings[ListaUsuario.ItemIndex]);
  auxS:=
copy(ListaUsuario.Items.Strings[ListaUsuario.ItemIndex],0,length(ListaUsuario.Items.Strings[ListaUsuario.Ite
mIndex])-length(ext));
  tex:= InputBox('Editar Usuário','Digite um nome para o usuário.',auxS);
  if tex <> '' then begin
    tirarCharInvalidos(tex);
    if FileExists(ListaUsuario.Items.Strings[ListaUsuario.ItemIndex])then begin
      RenameFile (ListaUsuario.Items.Strings[ListaUsuario.ItemIndex],tex+ext);
      ListaUsuario.Update;
    end;
  end;
end;

procedure TFormFacilitadores.ButtonAddCAtegClick(Sender: TObject);
begin
  if DirectoryExists(dirUsu+'\'+EditNovoFac.Text) then begin
    showmessage('A categoria '+EditNovoFac.Text+' já existe!');
  end else begin
    CreateDir(PChar(dirUsu+'\'+EditNovoFac.Text));
    LBFacilitador.Items.Add(EditNovoFac.Text);
    LBFacilitador.Update;
  end;
end;

end.
```

```
unit UFormMenu;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 UTeclado, ExtCtrls;

type
 TFormMenu = class(TForm)
  POp2: TPanel;
  POp1: TPanel;
  POp3: TPanel;
  procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure FormCreate(Sender: TObject);
  procedure FormShow(Sender: TObject);
  procedure POp1Click(Sender: TObject);
  procedure POp2Click(Sender: TObject);
  procedure POp3Click(Sender: TObject);
  procedure POp1MouseMove(Sender: TObject; Shift: TShiftState; X,
```

```
    Y: Integer);
  private
   teclado: TTeclado;
   opcoes: array[1..3] of TPanel;
   iOpcao: integer;
   procedure atualizaMenu(selIndice: integer= -1);
  public
   { Public declarations }
  end;

var
 FormMenu: TFormMenu;
 OPCAOMENU: string;
implementation

{$R *.DFM}

procedure TFormMenu.FormCreate(Sender: TObject);
var j: integer;
begin

  teclado:= TTeclado.create(ExtractFileDir(Application.ExeName)+'\config\teclas.ini');

  iOpcao:= 1;
  opcoes[1]:= POp1;
  opcoes[2]:= POp2;
  opcoes[3]:= POp3;

  for j:=1 to Length(opcoes) do begin
    if opcoes[j].Enabled then
      opcoes[j].Color:= clWhite;
  end;

  opcoes[1].Color:= clLime;

end;

procedure TFormMenu.FormShow(Sender: TObject);
begin
  OPCAOMENU:= '';
end;


procedure TFormMenu.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
    //função da seta para cima
    if (Key = teclado.SetaCima) then begin
      repeat
        Dec(iOpcao);
          if iOpcao < 1 then iOpcao:= Length(opcoes); //volta para a primeira posição ao chegar no fim do
array
      until opcoes[iOpcao].Enabled = true;
      atualizaMenu;
    end;
```

```
    //função da seta para baixo
    if (Key = teclado.SetaBaixo) then begin
      repeat
        Inc(iOpcao);
         if iOpcao > Length(opcoes) then iOpcao:= 1; //volta para a primeira posição ao chegar no fim do
array
      until opcoes[iOpcao].Enabled = true;
      atualizaMenu;
    end;

    //função da seta para esquerda
    if (Key = teclado.SetaEsq) then begin
      repeat
        Dec(iOpcao);
          if iOpcao < 1 then iOpcao:= Length(opcoes); //volta para a primeira posição ao chegar no fim do
array
      until opcoes[iOpcao].Enabled = true;
      atualizaMenu;
    end;

    //função da seta para direita
    if (Key = teclado.SetaDir) then begin
      repeat
        Inc(iOpcao);
          if iOpcao > Length(opcoes) then iOpcao:= 1; //volta para a primeira posição ao chegar no fim do
array
      until opcoes[iOpcao].Enabled = true;
      atualizaMenu;
    end;

    //função da tecla SIM ou OK
    if (Key = teclado.BotSim) then begin
      if iOpcao = 1 then OPCAOMENU:= 'novaPalavra';
      if iOpcao = 2 then OPCAOMENU:= 'reiniciar';
      if iOpcao = 3 then OPCAOMENU:= 'sair';
      self.Close;
    end;

    //função da tecla MENU
    if (Key = teclado.BotMenu) then begin
      self.Close;
    end;
end;

procedure TFormMenu.atualizaMenu(selIndice: integer= -1);
var j: integer;
begin
  for j:=1 to Length(opcoes) do begin
    if opcoes[j].Enabled then
      opcoes[j].Color:= clWhite;
  end;
  if selIndice <> -1 then
    opcoes[selIndice].Color:= clLime
```

```
  else
    opcoes[iOpcao].Color := clLime;
end;



procedure TFormMenu.POp1Click(Sender: TObject);
begin
  OPCAOMENU := 'novaPalavra';
  self.Close;
end;

procedure TFormMenu.POp2Click(Sender: TObject);
begin
  OPCAOMENU := 'reiniciar';
  self.Close;
end;

procedure TFormMenu.POp3Click(Sender: TObject);
begin
  OPCAOMENU := 'sair';
  self.Close;
end;


procedure TFormMenu.POp1MouseMove(Sender: TObject; Shift: TShiftState; X,
 Y: Integer);
begin
  atualizaMenu(StrToInt(TPanel(Sender).Name[4]));
end;

end.
```

```
unit UFormReport;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, C ontrols, Forms, Dialogs,
 ComCtrls, StdCtrls, FileCtrl,  UFuncoesUteis,Inifiles, UUsuario, Grids,
 OleCtrls, SHDocVw, ExtCtrls, Buttons;

type
 TFormReport = class(TForm)
   GBMin: TGroupBox;
   StatusBar1: TStatusBar;
   ListaUsuario: TFileListBox;
   Label2: TLabel;
   Button3: TButton;
   MemoReport: TMemo;
   PMin: TPanel;
   Label3: TLabel;
```

```
    ComboFacilitador: TComboBox;
    ListFacitiladores: TFileListBox;
    GBMax: TGroupBox;
    PanelCaption: TPanel;
    PMax: TPanel;
    BtMin: TButton;
    Panel3: TPanel;
    BtMax: TButton;
    PWebbrowser: TPanel;
    WebBrowser1: TWebBrowser;
    btPrint: TBitBtn;
    BtPrint2: TBitBtn;
    procedure FormCreate(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure ListaUsuarioClick(Sender: TObject);
    procedure ComboFacilitadorChange(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure BtMaxClick(Sender: TObject);
    procedure BtMinClick(Sender: TObject);
    procedure btPrintClick(Sender: TObject);
  private
    dirUsu: string;
    relGerado: string;
  public
    jogador: TUsuario;
  end;

var
  FormReport: TFormReport;

implementation

{$R *.DFM}

procedure TFormReport.FormCreate(Sender: TObject);
begin
  dirUsu:= ExtractFileDir(Application.ExeName)+'\usuarios';
  ListFacitiladores.Directory:= dirUsu;
  jogador:= TUsuario.create;
end;

procedure TFormReport.FormActivate(Sender: TObject);
var i: integer;
    nomeDir: string;
begin
  ComboFacilitador.Clear;
  for i:= 2 to ListFacitiladores.Items.C ount-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
     nomeDir:= copy(ListFacitiladores.Items.Strings[i],2,length(ListFacitiladores.Items.Strings[i])-2);
     ComboFacilitador.Items.Add(nomeDir);
  end;
  ComboFacilitador.ItemIndex:= 0;
  ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador.Text;
  ListaUsuario.Update;
end;
```

```
procedure TFormReport.Button3Click(Sender: TObject);
begin
  Self.close;
end;

procedure TFormReport.ListaUsuarioClick(Sender: TObject);
var
i, nPal, certas, percentualTotal, percentualPorPal: integer;
acertou, estilo: string;
begin
  MemoReport.Lines.Clear;
  MemoReport.Lines.Add('<html>                                                    ');
  MemoReport.Lines.Add('<head>                                                    ');
  MemoReport.Lines.Add('<meta    http-equiv="Content-Type"    content="text/html;    charset=iso-8859-1">
');
  MemoReport.Lines.Add('<title>Untitled Document</title>                          ');
  MemoReport.Lines.Add('<style type="text/css">                                   ');
  MemoReport.Lines.Add('body{                                                   ');
  MemoReport.Lines.Add('font-family: Verdana, Helvetica, Arial, sans-serif;       ');
  MemoReport.Lines.Add('font-size: 14px;                                        ');
  MemoReport.Lines.Add('color: #666666;                                         ');
  MemoReport.Lines.Add('}                                                     ');
  MemoReport.Lines.Add('h1{                                                     ');
  MemoReport.Lines.Add('   font-family: Verdana, Helvetica, Arial, sans-serif;    ');
  MemoReport.Lines.Add('   font-size: 18px;                                     ');
  MemoReport.Lines.Add('   color: #666666;                                      ');
  MemoReport.Lines.Add('}                                                     ');
  MemoReport.Lines.Add('.titulo {                                               ');
  MemoReport.Lines.Add('   font-family: Verdana, Helvetica, Arial, sans-serif;    ');
  MemoReport.Lines.Add('   font-size: 10px;                                     ');
  MemoReport.Lines.Add('   font-weight: bold;                                   ');
  MemoReport.Lines.Add('   color: #666666;                                      ');
  MemoReport.Lines.Add('   height:21;                                           ');
  MemoReport.Lines.Add('   vertical-align:middle;                               ');
  MemoReport.Lines.Add('   BORDER-RIGHT: #b5b5b5 1px solid;                       ');
  MemoReport.Lines.Add('   BORDER-LEFT: #b5b5b5 1px solid;                      ');
  MemoReport.Lines.Add('   background-image:url(top_bar.gif);                   ');
  MemoReport.Lines.Add('}                                                     ');
  MemoReport.Lines.Add('.linhaPar{                                              ');
  MemoReport.Lines.Add('   background-color: #FAFAFA;                           ');
  MemoReport.Lines.Add('   font-family: Verdana, Arial, Helvetica, sans-serif;    ');
  MemoReport.Lines.Add('   font-size: 12px;                                     ');
  MemoReport.Lines.Add('   color: #666666;                                      ');
  MemoReport.Lines.Add('   border-top: 1px none #b5b5b5;                        ');
  MemoReport.Lines.Add('   border-right: 1px solid #b5b5b5;                     ');
  MemoReport.Lines.Add('   border-bottom: 1px solid #b5b5b5;                    ');
  MemoReport.Lines.Add('   border-left: 1px solid #b5b5b5;                      ');
  MemoReport.Lines.Add('}                                                     ');
  MemoReport.Lines.Add('.linhaImpar{                                            ');
  MemoReport.Lines.Add('   background-color: #FFFFFF;                           ');
  MemoReport.Lines.Add('   font-family: Verdana, Arial, Helvetica, sans-serif;    ');
  MemoReport.Lines.Add('   font-size: 12px;                                     ');
  MemoReport.Lines.Add('   color: #666666;                                      ');
  MemoReport.Lines.Add('   border-top: 1px none #b5b5b5;                        ');
  MemoReport.Lines.Add('   border-right: 1px solid #b5b5b5;                     ');
```

```
  MemoReport.Lines.Add('  border-bottom: 1px solid #b5b5b5;                                  ');
  MemoReport.Lines.Add('  border-left: 1px solid #b5b5b5 ;                                 ');
  MemoReport.Lines.Add('}                                                                  ');
  MemoReport.Lines.Add('</style>                                                           ');
  MemoReport.Lines.Add('</head>                                                            ');
  MemoReport.Lines.Add('                                                                   ');
  MemoReport.Lines.Add('<body  leftmargin="0"  topmargin="0"  marginwidth="0"  marginheight="0">
');
   MemoReport.Lines.Add('<table width="95%" border="0" align="center" cellpadding="0" cellspacing="5">
');
  MemoReport.Lines.Add('  <tr>                                                             ');
  MemoReport.Lines.Add('                                <td      align="center"      class="titulo"><font
size="4">Relat&oacute;rio</font></td>  ');
  MemoReport.Lines.Add('  </tr>                                                            ');
  MemoReport.Lines.Add('  <tr>                                                             ');

  try
    jogador.setNome(ListaUsuario.FileName);
    jogador.LoadFromFile;
    nPal:= length(jogador.Palavras);
    MemoReport.Lines.Add('                          <td><b>Usu&aacute;rio:</b>'+jogador.Nome+'<br>
');
    MemoReport.Lines.Add('              <b>Total  de  palavras:</b>  '+IntToStr(nPal)+'  palavras.<br>
');
    certas:= 0;
    for i:= 0 to nPal-1 do
      if jogador.Palavras[i].Acertou then //ini.ReadBool  ('Palavra'+IntToStr(i),'acertouPalavra',false) then
        inc(certas);
    if nPal <> 0 then
      percentualTotal:= round(certas*100/nPal)
    else
      percentualTotal:= 0;

    MemoReport.Lines.Add('               <b>Palavras  certas:</b>  '+IntToStr(certas)+'  palavras.<br>
');
     MemoReport.Lines.Add('          <b>Aproveitamento:</b>  '+IntToStr(percentualTotal)+' %.</td>
');

    MemoReport.Lines.Add('  </tr>                                                         ');
    MemoReport.Lines.Add('  <tr>                                                          ');
     MemoReport.Lines.Add('     <td><table width="100%" border="0" cellspacing="0" cellpadding="0">
');
    MemoReport.Lines.Add('      <tr>                                                      ');
    MemoReport.Lines.Add('        <td align="center" class="titulo"> </td>           ');
    MemoReport.Lines.Add('        <td align="center" class="titulo">Palavra</td>          ');
    MemoReport.Lines.Add('        <td align="center" class="titulo">Acertou</td>          ');
    MemoReport.Lines.Add('        <td align="center" class="titulo">Letras certas </td>   ');
    MemoReport.Lines.Add('        <td align="center" class="titulo">Letras erradas </td>  ');
    MemoReport.Lines.Add('        <td align="center" class="titulo">Data</td>             ');
    MemoReport.Lines.Add('        <td align="center" class="titulo">Tempo</td>            ');
    MemoReport.Lines.Add('       <td align="center" class="titulo">(%)</td> ');
    MemoReport.Lines.Add('      </tr>                                                     ');

    for i:= 0 to nPal-1 do begin
      if jogador.Palavras[i].Acertou then //if ini.ReadBool  ('Palavra'+IntToStr(i),'acertouPalavra',false) then
```

```
        acertou:= 'SIM'
      else
        acertou:= 'NAO';
      percentualPorPal:=          (10-(length(jogador.Palavras[i].LetrasErradas)          div          2))*10;
//(length(jogador.Palavras[i].LetrasErradas) div 2 é devido a cada letra errada, ter uma virgula separando;

      if i mod 2 = 0 then
        estilo:= 'linhaImpar'
      else
        estilo:= 'linhaPar';

      MemoReport.Lines.Add('      <tr>                                                    ');
      MemoReport.Lines.Add('                          <td  class="'+estilo+'"> '+IntToStr(i+1)+'</td>
');
      MemoReport.Lines.Add('          <td class="'+estilo+'"> '+jogador.Palavras[i].Nome+'</td>
');
      MemoReport.Lines.Add('                              <td  class="'+estilo+'"> '+acertou+'</td>
');
      MemoReport.Lines.Add('                                                                  <td
class="'+estilo+'"> '+jogador.Palavras[i].LetrasCertas+'</td>      ');
      MemoReport.Lines.Add('                                                                  <td
class="'+estilo+'"> '+jogador.Palavras[i].LetrasErradas+'</td>     ');
       MemoReport.Lines.Add('          <td class="'+estilo+'"> '+jogador.Palavras[i].data+'</td>
');
       MemoReport.Lines.Add('          <td class="'+estilo+'"> '+jogador.Palavras[i].tempo+'</td>
');
      MemoReport.Lines.Add('         <td class="'+estilo+'"> '+IntToStr(percentualPorPal)+'%</td>
');
      MemoReport.Lines.Add('      </tr>                                                   ');
    end;
  finally

  end;
  MemoReport.Lines.Add('    </table></td>                                                 ');
  MemoReport.Lines.Add('  </tr>                                                         ');
  MemoReport.Lines.Add('</table>                                                         ');
  MemoReport.Lines.Add('</body>                                                          ');
  MemoReport.Lines.Add('</html>                                                          ');
  relGerado:= ExtractFileDir(Application.ExeName)+'\relatorios\'+jogador.Nome+'.htm';
  MemoReport.Lines.SaveToFile(relGerado);
  WebBrowser1.Navigate(relGerado);
end;



procedure TFormReport.ComboFacilitadorChange(Sender: TObject);
begin
  ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador.Text;
  ListaUsuario.Update;
end;

procedure TFormReport.BtMaxClick(Sender: TObject);
begin

  GBMax.Visible:= true;
```

```
  GBMin.Visible:= false;
  PWebbrowser.Parent:= PMax;
  WebBrowser1.Navigate(relGerado);
end;

procedure TFormReport.BtMinClick(Sender: TObject);
begin
  GBMax.Visible:= false;
  GBMin.Visible:= true;
  PWebbrowser.Parent:= PMin;
  WebBrowser1.Navigate(relGerado);
end;

procedure TFormReport.btPrintClick(Sender: TObject);
var eQuery:variant;
begin
 try eQuery:=WebBrowser1.QueryStatusWB(OLECMDID_PRINT);
   If eQuery and OLECMDF_ENABLED then
      WebBrowser1.ExecWB(OLECMDID_PRINT,OLECMDEXECOPT_PROMPTUSER,EmptyParam,EmptyParam)
     else MessageBox(0,'Falha ao imprimir!','Erro!',mb_ok);
 except end;
end;

end.
```

```
unit UFormTecladoCfg;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 StdCtrls, ExtCtrls, ComCtrls, Buttons, IniFiles, UTeclado;

type
 TFormTeclado = class(TForm)
   botSim: TImage;
   botNao: TImage;
   SetaEsq: TImage;
   setaCima: TImage;
   setaBaixo: TImage;
   SetaDir: TImage;
   Label1: TLabel;
   EditSetaCima: TEdit;
   EditSetaBaixo: TEdit;
   EditSetaDir: TEdit;
   EditBotNao: TEdit;
   EditBotSim: TEdit;
   EditSetaEsq: TEdit;
   Label2: TLabel;
   Label3: TLabel;
   Label4: TLabel;
   Label5: TLabel;
```

```
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    botOK: TBitBtn;
    botCancelar: TBitBtn;
    procedure FormActivate(Sender: TObject);
    procedure EditSetaCimaKeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure EditSetaBaixoKeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure EditSetaDirKeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure EditSetaEsqKeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure EditBotSimKeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure EditBotNaoKeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure botOKClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    teclado: TTeclado;
    procedure carregarTeclas;
    procedure salvarTeclas;
  end;

var
  FormTeclado: TFormTeclado;

implementation

{$R *.DFM}

//---------------------------------------------------------------------------

procedure TFormTeclado.FormCreate(Sender: TObjec t);
begin
  //
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.FormActivate(Sender: TObject);
begin

   carregarTeclas;

   if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas \acima.bmp') then
      setaCima.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas \acima.bmp');
   if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas \abaixo.bmp') then
      setaBaixo.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas \abaixo.bmp');
   if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas \direita.bmp') then
      setaDir.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas \direita.bmp');
```

```
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas \esquerda.bmp') then
    setaEsq.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas \esquerda.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas \SIM.bmp') then
    botSim.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas \SIM.bmp');
  if FileExists(ExtractFileDir(Application.ExeName)+'\config\teclas \NAO.bmp') then
    botNao.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\teclas \NAO.bmp');

end;

//---------------------------------------------------------------------------

procedure TFormTeclado.EditSetaCimaKeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  EditSetaCima.Text:= teclado.vetKeys[key];//inttostr(key);
  teclado.SetaCima:= key;
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.EditSetaBaixoKeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  EditSetaBaixo.Text:= teclado.vetKeys[key];//inttostr(key);
  teclado.SetaBaixo:= key;
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.EditSetaDirKeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  EditSetaDir.Text:= teclado.vetKeys[key];//inttostr(key);
  teclado.SetaDir:= key;
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.EditSetaEsqKeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  EditSetaEsq.Text:= teclado.vetKeys[key];//inttostr(key);
  teclado.SetaEsq:= key;
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.EditBotSimKeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  EditBotSim.Text:= teclado.vetKeys[key];//inttostr(key);
  teclado.BotSim:= key;
end;

//---------------------------------------------------------------------------
```

```
procedure TFormTeclado.EditBotNaoKeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  EditBotNao.Text:= teclado.vetKeys[key];//inttostr(key);
  teclado.BotMenu:= key;
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.c arregarTeclas;
begin
  teclado.LoadFromFile;
  EditBotNao.Text:= teclado.vetKeys[teclado.BotMenu];//IntToStr(teclado.BotNao);
  EditBotSim.Text:= teclado.vetKeys[teclado.BotSim];//IntToStr(teclado.BotSim);
  EditSetaBaixo.Text:= teclado.vetKeys[teclado.SetaBaixo];//IntToStr(teclado.SetaBaixo);
  EditSetaCima.Text:= teclado.vetKeys[teclado.SetaCima];//IntToStr(teclado.SetaCima);
  EditSetaEsq.Text:= teclado.vetKeys[teclado.SetaEsq];//IntToStr(teclado.SetaEsq);
  EditSetaDir.Text:= teclado.vetKeys[teclado.SetaDir];//IntToStr(teclado.SetaDir);
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.salvarTeclas;
begin
  teclado.SaveToFile;
end;

//---------------------------------------------------------------------------

procedure TFormTeclado.botOKClick(Sender: TObject);
begin
  salvarTeclas;
end;

//---------------------------------------------------------------------------

end.
```

```
unit UFormUsuario;

interface

uses
 Windows, Messages,SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 ComCtrls, StdCtrls, FileCtrl,  UFuncoesUteis,Inifiles, UUsuario, Grids;

type
 TFormUsuario = class(TForm)
   GroupBoxNovoUsu: TGroupBox;
   EditUsuNome: TEdit;
   Label1: TLabel;
```

```
    Button1: TButton;
    GroupBoxListaUsu: TGroupBox;
    StatusBar1: TStatusBar;
    ListaUsuario: TFileListBox;
    Label2: TLabel;
    Button2: TButton;
    Button3: TButton;
    MemoInfoUsu: TMemo;
    ListViewUsu: TListView;
    ListFacitiladores: TFileListBox;
    ComboFacilitador: TComboBox;
    Label3: TLabel;
    ListFacitiladores2: TFileListBox;
    ComboFacilitador2: TComboBox;
    Label4: TLabel;
    Button4: TButton;
    procedure FormCreate(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure EditUsuNomeExit(Sender: TObject);
    procedure ListaUsuarioClick(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure ListViewUsuClick(Sender: TObject);
    procedure ComboFacilitadorExit(Sender: TObject);
    procedure ComboFacilitador2Change(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
  private
    dirUsu: string;
  public
    jogador: TUsuario;
    procedure AtualizaFacilitadores1;
    procedure AtualizaFacilitadores2;
  end;

var
  FormUsuario: TFormUsuario;

implementation

uses UFormReport;

{$R *.DFM}

procedure TFormUsuario.FormCreate(Sender: TObject);
begin
   dirUsu:= ExtractFileDir(Application.ExeName)+'\usuarios';
   ListaUsuario.Directory:= dirUsu;
   jogador:= TUsuario.create;
   ListFacitiladores.Directory:= dirUsu;
end;

procedure TFormUsuario.FormActivate(Sender: TObject);
begin
```

```
  AtualizaFacilitadores1;
  A tualizaFacilitadores2;
  ComboFacilitador.ItemIndex:= 0;
  ComboFacilitador2.ItemIndex:= 0;
  ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador2.Text;
  ListaUsuario.Update;
end;

procedure TFormUsuario.Button3Click(Sender: TObject);
begin
  Self.close;
end;

procedure TFormUsuario.Button1Click(Sender: TObject);
var podeCriarUsu: boolean;
   facilitador: string;
   iSel: integer;
begin

  if (EditUsuNome.text = '') then
    MessageBox(0,'O campo USUÁRIO não pode ser nulo','Erro ao cadastar usuário',mb_ok)
  else if (ComboFacilitador.text = '') then
    MessageBox(0,'O campo FACILITADOR não pode ser nulo','Erro ao cadastar usuário',mb_ok)
  else begin
    if (not DirectoryExists(dirUsu+'\'+ComboFacilitador.Text)) then
      CreateDir(dirUsu+'\'+ComboFacilitador.Text);

    facilitador:= ComboFacilitador.Text;
    podeCriarUsu:= true;
    if EditUsuNome.text <> '' then  begin
      if FileExists(dirUsu+'\'+facilitador+'\'+EditUsuNome.text+'.txt') then begin
        if Application.MessageBox(PChar('Já existe um usuário com o nome "'+EditUsuNome.Text+'". Deseja
substituir o usuário existente?'),PChar('Atenção'),MB_YESNO) = IDNO then begin
          podeCriarUsu:= false;
        end;
      end;
      if podeCriarUsu then begin
        MemoInfoUsu.Lines.SaveToFile(dirUsu+'\'+facilitador+'\'+EditUsuNome.text+'.txt');
        ListaUsuario.Update;
        ListaUsuario.Refresh;
        ListaUsuario.Repaint;
      end;
    end;
    iSel:= ComboFacilitador.ItemIndex;
    AtualizaFacilitadores1;
    AtualizaFacilitadores2;
    ComboFacilitador.Text:= facilitador;
    ComboFacilitador2.Text:= facilitador;

    ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador2.Text;
    ListaUsuario.Update;
  end;

end;
```

```
procedure TFormUsuario.EditUsuNomeExit(Sender: TObject);
begin
  if EditUsuNome.Text <> '' then begin
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'/','-');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'\','-');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,':','_');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'*','_');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'?','_');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'<','_');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'>','_');
    EditUsuNome.Text:= SubstituirCaracteres(EditUsuNome.Text,'|','_');
  end;

end;

procedure TFormUsuario.ListaUsuarioClick(Sender: TObject);
var
i, j, nPal, certas, percentualTotal, percentualPorPal: integer;
acertou: string;
begin
  //MemoInfoUsu.Lines.LoadFromFile(ListaUsuario.FileName);

  try
    jogador.setNome(ListaUsuario.FileName);
    jogador.LoadFromFile;

    MemoInfoUsu.Lines.Clear;
    nPal:= length(jogador.Palavras);
    MemoInfoUsu.Lines.Add('Usuário: '+jogador.Nome);
//        MemoInfoUsu.Lines.Add('');
    MemoInfoUsu.Lines.Add('Total de palavras: '+IntToStr(nPal)+' palavras.');

    certas:= 0;
    for i:= 0 to nPal-1 do
      if jogador.Palavras[i].Acertou then //ini.ReadBool  ('Palavra'+IntToStr(i),'acertouPalavra',false) then
        inc(certas);
    if nPal <> 0 then
      percentualTotal:= round(certas*100/nPal)
    else
      percentualTotal:= 0;

    MemoInfoUsu.Lines.Add('Palavras certas: '+IntToStr(certas)+' palavras.');
    MemoInfoUsu.Lines.Add('Aproveitamento:  '+IntToStr(percentualTotal)+' %.');

    for i:= 0 to ListViewUsu.Items.Count-1 do
      ListViewUsu.Items.Clear;

    for i:= 0 to nPal-1 do begin
      if jogador.Palavras[i].Acertou then //if ini.ReadBool  ('Palavra'+IntToStr(i),'acertouPalavra',false) then
        acertou:= 'SIM'
      else
        acertou:= 'NAO';
      percentualPorPal:=          (10-(length(jogador.Palavras[i].LetrasErradas)          div          2))*10;
//(length(jogador.Palavras[i].LetrasErradas) div 2 é devido a cada letra errada, ter uma virgula separando;
      ListViewUsu.Items.Add;
```

```
        ListViewUsu.Items[i].Caption:= IntToStr(i+1);
        ListViewUsu.Items[i].SubItems.Add(jogador.Palavras[i].Nome);
        ListViewUsu.Items[i].SubItems.Add(acertou);
        ListViewUsu.Items[i].SubItems.Add(jogador.Palavras[i].LetrasErradas);
        ListViewUsu.Items[i].SubItems.Add(jogador.Palavras[i].LetrasCertas);
        ListViewUsu.Items[i].SubItems.Add(jogador.Palavras[i].tempo);
        ListViewUsu.Items[i].SubItems.Add(jogador.Palavras[i].data);
        ListViewUsu.Items[i].SubItems.Add(IntToStr(percentualPorPal));
      end;
    finally

    end;


end;


procedure TFormUsuario.Button2Click(Sender: TObject);
begin
  if Application.MessageBox(PChar('Deseja realmente excluir o usuário "'+jogador.Nome+'"
?'),PChar('Atenção'),MB_YESNO) = IDYES then begin
    DeleteFile(ListaUsuario.FileName);
    ListaUsuario.Update;
    ListaUsuario.Refresh;
    ListaUsuario.Repaint;
  end;
end;

procedure TFormUsuario.AtualizaFacilitadores1;
var i: integer;
  nomeDir: string;
begin
  ListFacitiladores.Update;
  ComboFacilitador.Clear;
  for i:= 2 to ListFacitiladores.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir:= copy(ListFacitiladores.Items.Strings[i],2,length(ListFacitiladores.Items.Strings[i])-2);
    ComboFacilitador.Items.Add(nomeDir);
  end;
end;

procedure TFormUsuario.AtualizaFacilitadores2;
var i: integer;
  nomeDir: string;
begin
  ListFacitiladores.Update;
  ComboFacilitador2.Clear;
  for i:= 2 to ListFacitiladores.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir:= copy(ListFacitiladores.Items.Strings[i],2,length(ListFacitiladores.Items.Strings[i])-2);
    ComboFacilitador2.Items.Add(nomeDir);
  end;
end;

procedure TFormUsuario.FormShow(Sender: TObject);
begin
  if ListaUsuario.FileName <> '' then
```

```
    MemoInfoUsu.Lines.LoadFromFile(ListaUsuario.FileName);

end;

procedure TFormUsuario.ListViewUsuClick(Sender: TObject);
begin
  //
end;

procedure TFormUsuario.ComboFacilitadorExit(Sender: TObject);
begin
  if ComboFacilitador.Text <> '' then begin
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'/','-');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'\','-');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,':','_');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'*','_');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'?','_');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'<','_');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'>','_');
    ComboFacilitador.Text:= SubstituirCaracteres(ComboFacilitador.Text,'|','_');
  end;
end;

procedure TFormUsuario.ComboFacilitador2Change(Sender: TObject);
begin
  ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador2.Text;
  ListaUsuario.Update;
end;

procedure TFormUsuario.Button4Click(Sender: TObject);
begin
  FormReport.showmodal;
end;

end.
```

```
unit UMainForm;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls , Forms, Dialogs;

type
 TForm1 = class(TForm)
 private
   { Private declarations }
 public
   { Public declarations }
 end;

var
 Form1: TForm1;
```

```
implementation

{$R *.DFM}

end.
```

```
unit UPalavra;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ComCtrls, StdCtrls, ExtDlgs, ExtCtrls, FileCtrl, jpeg, UFuncoesUteis,
  Menus;

type
  TFormPalavra = class(TForm)
    GroupBox1: TGroupBox;
    Label1: TLabel;
    EditPalNome: TEdit;
    Button1: TButton;
    StatusBar1: TStatusBar;
    Image1: TImage;
    Label2: TLabel;
    Bevel1: TBevel;
    OpenPictureDialog1: TOpenPictureDialog;
    GroupBox2: TGroupBox;
    LabelAddImg: TLabel;
    Button4: TButton;
    PageControlPalavras: TPageControl;
    TabGerCat: TTabSheet;
    CategoriasImg: TFileListBox;
    PanelMostraPalavras: TPanel;
    Label3: TLabel;
    ListaPalavras: TFileListBox;
    ImgGde: TImage;
    Button2: TButton;
    Button3: TButton;
    Label4: TLabel;
    EditNovaCateg: TEdit;
    Label5: TLabel;
    Label6: TLabel;
    ButtonAddCAteg: TButton;
    Bevel2: TBevel;
    ListBoxListaCateg: TListBox;
    ListaPalporCateg: TFileListBox;
    ButtonDelCateg: TButton;
    ButtonDelPal: TButton;
    ComboCateg: TComboBox;
    Label7: TLabel;
    Button5: TButton;
    PopupPal: TPopupMenu;
```

```
    PopupCat: TPopupMenu;
    Excluirpalavra1: TMenuItem;
    EditarPalavra1: TMenuItem;
    ExcluirCategoria1: TMenuItem;
    EditarCategoria1: TMenuItem;
    ButtonEditPal: TButton;
    ButtonEditCat: TButton;
    procedure Image1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure EditPalNomeExit(Sender: TObject);
    procedure ListaPalavrasClick(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure PageControlPalavrasChange(Sender: TObject);
    procedure EditNovaCategExit(Sender: TObject);
    procedure ListBoxListaCategClick(Sender: TObjec t);
    procedure ButtonAddCAtegClick(Sender: TObject);
    procedure ButtonDelCategClick(Sender: TObject);
    procedure ButtonDelPalClick(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure PopupCatPopup(Sender: TObject);
    procedure ExcluirCategoria1Click(Sender: TObject);
    procedure Excluirpalavra1Click(Sender: TObject);
    procedure PopupPalPopup(Sender: TObject);
    procedure EditarCategoria1Click(Sender: TObject);
    procedure EditarPalavra1Click(Sender: TObject);
    procedure ButtonEditCatClick(Sender: TObject);
    procedure ButtonEditPalClick(Sender: TObject);
  private
    procedure montaListaCateg;
    procedure montaComboCateg;
    procedure tirarCharInvalidos(var texto: string);
  public
    dirPal: string;
  end;

var
  FormPalavra: TFormPalavra;

implementation

{$R *.DFM}

procedure TFormPalavra.FormCreate(Sender: TObject);
var i: integer;
    nomeDir: string;
begin
  dirPal:= ExtractFileDir(Application.ExeName)+'\palavras';
  CategoriasImg.Directory:= dirP al;

  for i:= 2 to CategoriasImg.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    with TTabSheet.Create(Self) do begin
      PageControl := PageControlPalavras;
```

```
      Caption := CategoriasImg.Items.Strings[i];
      name:= 'tab'+copy(CategoriasImg.Items.Strings[i],2,length(CategoriasImg.Items.Strings[i])-2);
    end;
  end;
  montaListaCateg;
  montaComboCateg;
end;

procedure TFormPalavra.Image1Click(Sender: TObject);
begin
  try
    if OpenPictureDialog1.Execute then begin
      Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
      LabelAddImg.Visible:= false;
    end;
  except

  end;
end;


procedure TFormPalavra.Button3Click(Sender: TObject);
begin
  self.close;
end;

procedure TFormPalavra.Button1Click(Sender: TObject);
var caminhoImg: string;
begin
  if editPalNome.Text = '' then
    ShowMessage('O campo PALAVRA não pode ser nulo!')
  else if ComboCateg.Text = '' then
    ShowMessage('O campo CATEGORIA não pode ser nulo!')
  else if O penPictureDialog1.FileName = '' then
    ShowMessage('O campo IMAGEM não pode ser nulo!')
  else begin
    caminhoImg:= '\'+ComboCateg.Text+'\'+EditPalNome.text+ExtractFileExt(OpenPictureDialog1.filename);
    if FileExists(dirPal+caminhoImg) then begin
      if Application.MessageBox(PChar('Esta palavra já existe. Deseja substitui-
lá?'),PChar('Atenção'),MB_YESNO) = IDYES then
        CopyFile(PChar(OpenPictureDialog1.filename),PChar(dirPal+caminhoImg),false)
    end else
      CopyFile(P Char(OpenPictureDialog1.filename),PChar(dirPal+caminhoImg),false);
    ListaPalporCateg.Update;
    ListaPalavras.Update;
    ListaPalavras.Refresh;
  end;
end;

procedure TFormPalavra.EditPalNomeExit(Sender: TObject);
var i: integer;
   auxNome: string;
begin
  if editPalNome.Text <> '' then begin
    auxNome:= editPalNome.Text;
```

```
    tirarCharInvalidos(auxNome);
    editPalNome.Text:= auxNome;
    i:= 1;
    repeat
      if editPalNome.Text <> '' then
        if editPalNome.Text[i] = ' ' then begin
          editPalNome.Text:= copy(editPalNome.Text,i+1,length(editPalNome.Text));
          inc(i);
        end else exit;
    until ((editPalNome.Text[i] <> ' ')or(length(editPalNome.Text)=1))
  end;
end;

procedure TFormPalavra.ListaPalavrasClick(Sender: TObject);
begin
  ImgGde.Picture.LoadFromFile(ListaPalavras.FileName);
  ImgGde.Visible:= true;
end;

procedure TFormPalavra.Button2Click(Sender: TObject);
var nome: string;
   i: integer;
begin

  for i := 0 to (ListaPalavras.Items.Count - 1) do begin
    try
      if ListaPalavras.Selected[i] then  begin
        nome:= ExtractFileName(ListaPalavras.FileName);
        nome:= copy(nome,0,length(nome)-4);
        if Application.MessageBox(PChar('Deseja realmente excluir a palavra "'+nome+'"
?'),PChar('Atenção'),MB_YESNO) = IDYES then begin
          DeleteFile(ListaPalavras.Items.Strings[i]);

        end;
      end;
    except
    end;
  end;
  ListaPalporCateg.Update;
  ListaPalporCateg.Refresh;
  ListaPalporCateg.Repaint;
end;

procedure TFormPalavra.FormShow(Sender: TObject);
var i: integer;
begin
  ListBoxListaCateg.ItemIndex:= -1;
  ListaPalporCateg.ItemIndex:= -1;
end;

procedure TFormPalavra.PageControlPalavrasChange(Sender: TObject);
var
  PrevCaption, NextCaption: ShortString;
  nomeDir: string;
begin
```

```
  with (Sender as TPageControl) do begin
    if ActivePage.Name <> 'TabGerCat' then begin
      PanelMostraPalavras.Visible:= true;
      PanelMostraPalavras.Parent:= ActivePage;
      PanelMostraPalavras.Align:= alClient;
      nomeDir:= copy(ActivePage.Caption,2,length(ActivePage.Caption)-2);
      ListaPalavras.Directory:= dirPal+'\'+nomeDir;
      ListaPalavras.Update;
      ListaPalavras.Refresh;
      ImgGde.Visible:= false;
    end;
  end;
end;

procedure TFormPalavra.EditNovaCategExit(Sender: TObject);
var i: integer;
   auxNome: string;
begin
  if EditNovaCateg.Text <> '' then begin
    auxNome:= EditNovaCateg.Text;
    tirarCharInvalidos(auxNome);
    EditNovaCateg.Text:= auxNome;
    i:= 1;
    repeat
      if EditNovaCateg.Text <> '' then
        if EditNovaCateg.Text[i] = ' ' then begin
          EditNovaCateg.Text:= copy(EditNovaCateg.Text,i+1,length(EditNovaCateg.Text));
          inc(i);
        end else exit;
    until ((EditNovaCateg.Text[i] <> ' ')or(length(EditNovaCateg.Text)=1))
  end;

end;

procedure TFormPalavra.ListBoxListaCategClick(Sender: TObject);
var i: integer;
   nome: string;
begin
  ListaPalporCateg.Directory:= dirPal+'\'+ListBoxListaCateg.Items[ListBoxListaCateg.ItemIndex];
  ListaPalporCateg.Update;
  ListaPalporCateg.ItemIndex:= -1;
end;

procedure TFormPalavra.ButtonAddCAtegClick(Sender: TObject);
begin
  if DirectoryExists(dirPal+'\'+EditNovaCateg.Text) then begin
    showmessage('A categoria '+EditNovaCateg.Text+' já existe!');
  end else begin
    CreateDir(PChar(dirPal+'\'+EditNovaCateg.Text));
    ListBoxListaCateg.Items.Add(EditNovaCateg.Text);
    ListBoxListaCateg.Update;
    with TTabSheet.Create(Self) do begin
      PageControl := PageControlPalavras;
      Caption := '['+EditNovaCateg.Text+']';
      name:= 'tab'+ EditNovaCateg.Text;
```

```
    end;
  end;
  montaComboCateg;
end;

procedure TFormPalavra.ButtonDelCategClick(Sender: TObject);
var delDir: string;
   i: integer;
begin
  for i := 0 to (ListBoxListaCateg.Items.Count - 1) do begin
    try
      if ListBoxListaCateg.Selected[i] then  begin
        deldir:= dirPal+'\'+ListBoxListaCateg.Items.Strings[i];
        if DirectoryExists(deldir) then begin
          if Application.MessageBox(PChar('Deseja realmente excluir a categoria "'+ListBoxListaCateg.Items.Strings[i]+'" ?'+#13+'Esta categoria possui '+IntToStr(ListaPalporCateg.Items.Count)+' palavra(s) que serão EXCLUIDAS junto com a categoria'+#13+'Esta operação não poderá ser desfeita.'),PChar('Atenção'),MB_YESNO) = IDYES then begin
            DeleteDir(Self.Handle,deldir);
            CategoriasImg.Update;
            TTabSheet(FindComponent('tab'+ListBoxListaCateg.Items.Strings[i])).Free;
          end;
        end;
      end;
    except
    end;
  end;
  montaListaCateg;
  ListaPalporCateg.Clear;
end;

procedure TFormPalavra.ButtonEditCatClick(Sender: TObject);
var tex: string;
   urlCat, urlCatNovo: string;
begin

  urlCat:= PChar(dirPal+'\'+ListBoxListaCateg.Items.Strings[ListBoxListaCateg.ItemIndex]);
  tex:= InputBox('Editar Categoria','Digite um nome para a nova categoria',ListBoxListaCateg.Items.Strings[ListBoxListaCateg.ItemIndex]);
  //tex:= InputBox('Editar Categoria','Digite um nome para a nova categoria',urlFac+' | '+dirUsu+'\'+tex);

  if tex <> '' then begin
    tirarCharInvalidos(tex);
    urlCatNovo:=PChar(dirPal+'\'+tex+'\');

    if DirectoryExists(urlCatNovo)then
      MessageBox(0,PChar('A Categoria "'+tex+'" já existe.'+#13+'Escolha outro nome'),'Erro ao renomear',mb_ok)
    else begin
      CreateDir(urlCatNovo);
      CopiaConteudoDir(urlCat+'\*.*',urlCatNovo);
      DeleteDir(Self.Handle,urlCat);
      ListBoxListaCateg.Update;
      montaListaCateg;
    end;
```

```
  end;
end;

procedure TFormPalavra.ButtonDelPalClick(Sender: TObject);
var nome: string;
   i: integer;
begin
  for i := 0 to (ListaPalporCateg.Items.Count - 1) do begin
    try
      if ListaPalporCateg.Selected[i] then  begin
        nome:= ExtractFileName(ListaPalporCateg.FileName);
        nome:= copy(nome,0,length(nome)-4);
        if Application.MessageBox(PChar('Deseja realmente excluir a palavra "'+nome+'"
?'),PChar('Atenção'),MB_YESNO) = IDYES then begin
          DeleteFile(ListaPalporCateg.Items.Strings[i]);

        end;
      end;
    except
    end;
  end;
  ListaPalporCateg.Update;
  ListaPalporCateg.Refresh;
  ListaPalporCateg.Repaint;
end;

procedure TFormPalavra.ButtonEditPalClick(Sender: TObject);
var tex, ext, auxS: string;
begin
  ext:= ExtractFileExt(ListaPalporCateg.Items.Strings[ListaPalporCateg.ItemIndex]);
  auxS:=
copy(ListaPalporCateg.Items.Strings[ListaPalporCateg.ItemIndex],0,length(ListaPalporCateg.Items.Strings[List
aPalporCateg.ItemIndex])-length(ext));
  tex:= InputBox('Editar Categoria','Digite um nome para a nova categoria',auxS);
  if tex <> '' then begin
    tex:= UpperCase(tex);
    tirarCharInvalidos(tex);
    if FileExists(ListaPalporCateg.Items.Strings[ListaPalporCateg.ItemIndex])then begin
      RenameFile (ListaPalporCateg.Items.Strings[ListaPalporCateg.ItemIndex],tex+ext);
      ListaPalporCateg.Update;
    end;
  end;
end;

procedure TFormPalavra.montaListaCateg;
var i: integer;
   nomeDir: string;
begin
  ListBoxListaCateg.clear;
  for i:= 2 to CategoriasImg.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir:= copy(CategoriasImg.Items.Strings[i],2,length(CategoriasImg.Items.Strings[i])-2);
    ListBoxListaCateg.Items.Add(nomeDir);
  end;
end;
```

```
procedure TFormPalavra.montaComboCateg;
var i: integer;
   nomeDir: string;
begin
  ComboCateg.Clear;
  for i:= 2 to CategoriasImg.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
     nomeDir:= copy(CategoriasImg.Items.Strings[i],2,length(CategoriasImg.Items.Strings[i])-2);
     ComboCateg.Items.Add(nomeDir);
  end;
end;

procedure TFormPalavra.Button5Click(Sender: TObject);
begin
  Winexec('teclado.exe',SW_SHOWNORMAL);
end;

procedure TFormPalavra.PopupCatPopup(Sender: TObject);
begin
  EditarCategoria1.Caption:= 'Editar Categoria';
  ExcluirCategoria1.Caption:= 'Excluir Categoria';
  EditarCategoria1.Enabled:= ListBoxListaCateg.ItemIndex >=0 ;
  ExcluirCategoria1.Enabled:= ListBoxListaCateg.ItemIndex >=0 ;
  if EditarCategoria1.Enabled then begin
     ExcluirCategoria1.Caption:=                    'Excluir                    Categoria
('+ListBoxListaCateg.Items.Strings[ListBoxListaCateg.ItemIndex]+')';
     EditarCategoria1.Caption:=                     'Editar                     Categoria
('+ListBoxListaCateg.Items.Strings[ListBoxListaCateg.ItemIndex]+')';
  end;
end;

procedure TFormPalavra.ExcluirCategoria1Click(Sender: TObject);
begin
  ButtonDelCateg.Click;
end;

procedure TFormPalavra.Excluirpalavra1Click(Sender: TObject);
begin
  ButtonDelPal.Click;
end;

procedure TFormPalavra.PopupPalPopup(Sender: TObject);
begin
  EditarPalavra1.Caption:= 'Editar Palavra';
  ExcluirPalavra1 .Caption:= 'Excluir Palavra';
  EditarPalavra1.Enabled:= ListaPalporCateg.ItemIndex >=0 ;
  ExcluirPalavra1.Enabled:= ListaPalporCateg.ItemIndex >=0 ;
  if EditarPalavra1.Enabled then begin
     ExcluirPalavra1.Caption:=                      'Excluir                      Palavra
('+ListaPalporCateg.Items.Strings[ListaPalporCateg.ItemIndex]+')';
     EditarPalavra1.Caption:=                       'Editar                       Palavra
('+ListaPalporCateg.Items.Strings[ListaPalporCateg.ItemIndex]+')';
  end;
end;

procedure TFormPalavra.EditarCategoria1Click(Sender: TObject);
```

```
begin
  ButtonEditCat.Click;
end;

procedure TFormPalavra.EditarPalavra1Click(Sender: TObject);
begin
  ButtonEditPal.Click;
end;

procedure TFormPalavra.tirarCharInvalidos(var texto: string);
begin
  texto:= SubstituirCaracteres(texto,'/','-');
  texto:= SubstituirCaracteres(texto,'\','-');
  texto:= SubstituirCaracteres(texto,':','-');
  texto:= SubstituirCaracteres(texto,'*','-');
  texto:= SubstituirCaracteres(texto,'?','-');
  texto:= SubstituirCaracteres(texto,'<','-');
  texto:= SubstituirCaracteres(texto,'>','-');
  texto:= SubstituirCaracteres(texto,'|','-');
  texto:= SubstituirCaracteres(texto,'Â','A');
  texto:= SubstituirCaracteres(texto,'â','A');
  texto:= SubstituirCaracteres(texto,'Ã','A');
  texto:= SubstituirCaracteres(texto,'ã','A');
  texto:= SubstituirCaracteres(texto,'Á','A');
  texto:= SubstituirCaracteres(texto,'á','A');
  texto:= SubstituirCaracteres(texto,'À','A');
  texto:= SubstituirCaracteres(texto,'à','A');
  texto:= SubstituirCaracteres(texto,'Ä','A');
  texto:= SubstituirCaracteres(texto,'ä','A');
  texto:= SubstituirCaracteres(texto,'Ê','E');
  texto:= SubstituirCaracteres(texto,'ê','E');
  texto:= SubstituirCaracteres(texto,'É','E');
  texto:= SubstituirCaracteres(texto,'é','E');
  texto:= SubstituirCaracteres(texto,'È','E');
  texto:= SubstituirCaracteres(texto,'è','E');
  texto:= SubstituirCaracteres(texto,'Ë','E');
  texto:= SubstituirCaracteres(texto,'ë','E');
  texto:= SubstituirCaracteres(texto,'Î','I');
  texto:= SubstituirCaracteres(texto,'î','I');
  texto:= SubstituirCaracteres(texto,'Í','I');
  texto:= SubstituirCaracteres(texto,'í','I');
  texto:= SubstituirCaracteres(texto,'Ì','I');
  texto:= SubstituirCaracteres(texto,'ì','I');
  texto:= SubstituirCaracteres(texto,'Ï','I');
  texto:= SubstituirCaracteres(texto,'ï','I');
  texto:= SubstituirCaracteres(texto,'Ô','O');
  texto:= SubstituirCaracteres(texto,'ô','O');
  texto:= SubstituirCaracteres(texto,'Õ','O');
  texto:= SubstituirCaracteres(texto,'õ','O');
  texto:= SubstituirCaracteres(texto,'Ó','O');
  texto:= SubstituirCaracteres(texto,'ó','O');
  texto:= SubstituirCaracteres(texto,'Ò','O');
  texto:= SubstituirCaracteres(texto,'ò','O');
  texto:= SubstituirCaracteres(texto,'Ö','O');
  texto:= SubstituirCaracteres(texto,'ö','O');
```

```
  texto:= SubstituirCaracteres(texto,'Û','U');
  texto:= SubstituirCaracteres(texto,'û','U');
  texto:= SubstituirCaracteres(texto,'Ú','U');
  texto:= SubstituirCaracteres(texto,'ú','U');
  texto:= SubstituirCaracteres(texto,'Ù','U');
  texto:= SubstituirCaracteres(texto,'ù','U');
  texto:= SubstituirCaracteres(texto,'Ü','U');
  texto:= SubstituirCaracteres(texto,'ü','U');
  texto:= SubstituirCaracteres(texto,'Ñ','N');
  texto:= SubstituirCaracteres(texto,'ñ','N');
  texto:= SubstituirCaracteres(texto,'Ç','C');
  texto:= SubstituirCaracteres(texto,'ç','C');
end;

end.
```

```
unit UPrincipal;

interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 StdCtrls, ExtCtrls, Buttons, Menus, FileCtrl,
 MMSystem, IniFiles, D b, ADODB, UUsuario,UFuncoesUteis, UTeclado, ImgList ;

type
 TKind=(circle,rectangle,Line); {kinds of elements to draw figure}
 THPiece=class(TObject)  {pieces used to draw victim}
   kind:TKind;
   start,stop: TPoint;
 end;

 TFormPrincipal = class(TForm)
   PA: TPanel;
   PB: TPanel;
   PC: TPanel;
   PD: TPanel;
   PE: TPanel;
   PF: TPanel;
   PG: TPanel;
   PH: TPanel;
   PI: TPanel;
   PJ: TPanel;
   PK: TPanel;
   PL: TPanel;
   PM: TPanel;
   PN: TPanel;
   PO: TPanel;
   PP: TPanel;
   PQ: TPanel;
   PR: TPanel;
   PS: TPanel;
   PT: TPanel;
```

```
    PU: TPanel;
    PV: TPanel;
    PW: TPanel;
    PX: TPanel;
    PY: TPanel;
    PZ: TPanel;
    ForcaImagem: TImage;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    labelFimJogo: TLabel;
    Cadastrarusurio1: TMenuItem;
    Cadastrar1: TMenuItem;
    Usurio1: TMenuItem;
    Palavra1: TMenuItem;
    Ajuda1: TMenuItem;
    Documentao1: TMenuItem;
    Sobre1: TMenuItem;
    SelecionarUsurio1: TMenuItem;
    N1: TMenuItem;
    NovoJogo1: TMenuItem;
    ListaPalavras: TFileListBox;
    PainelTempo: TPanel;
    Timer1: TTimer;
    Relatrios1: TMenuItem;
    Dados1: TMenuItem;
    Image1: TImage;
    N2: TMenuItem;
    ConfiguraodoTeclado1: TMenuItem;
    Bevel1: TBevel;
    ImgGde: T Image;
    LabelNome: TLabel;
    aPalavra: TLabel;
    letrasCertas: TLabel;
    letrasErradas: TLabel;
    ImageVaso: TImage;
    Label1: TLabel;
    Label2: TLabel;
    Facilitador1: TMenuItem;
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure FormCreate(Sender: TObject);
    procedure clicarNaLetra(Sender: TObject);
    procedure Cadastrarusurio1Click(Sender: TObject);
    procedure Usurio1Click(Sender: TObject);
    procedure Palavra1Click(Sender: TObject);
    procedure SelecionarUsurio1Click(Sender: TObject);
    procedure NovoJogo1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Sobre1Click(Sender: TObject);
    procedure Dados1Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure ConfiguraodoTeclado1Click(Sender: TObject);
    procedure Facilitador1Click(Sender: TObject);
  private
```

```
    HangmanList:TList;
    piececount:integer;
    lastcolor:TColor;
    palavraSecreta:string;
    palavraSecretaSemCaractEsp:string;
    LetrasSelecionadas:set of char;
    procedure CriarCorpo;
    procedure ChecarUmaLetra(ch:char);
    procedure DesenharParteCorpo(piececolor:TColor);
    procedure iniciaJogo(mesmaPalavra: boolean= false);
    procedure ligarCronometro;
    procedure desligarCronometro;
  public
    Letras: array[1..26] of TPanel;
    iLetraAtual: integer;
    jogoIniciado: boolean;
    fimDeJogo: boolean;
    dirConfig: string;
    dirPal: string;
    dirUsu: string;
    jogador: TUsuario;
    teclado: TTeclado;

    procedure atualizaPosicaoAlfabeto(indice: integer); overload;
    procedure atualizaPosicaoAlfabeto; overload;
    procedure desabilitaBotao(indice: integer);
    procedure inicializaBotoesLetras;
    function  escolhePalavra: string;
    procedure selecionarUsuario;
    function tirarCharAcentuados(str: String): String;
  end;

var
  FormPrincipal: TFormPrincipal;

implementation

uses UFormUsuario, UPalavra, USelUsu, USobre, UFormTecladoCfg,
  UDialogsFimJogo, UFormReport, UFormFacilitadores, UFormMenu;

{$R *.DFM}

procedure TFormPrincipal.FormCreate(Sender: TObject);
begin

   iLetraAtual:= 1;
   Letras[1]:= PA;
   Letras[2]:= PB;
   Letras[3]:= PC;
   Letras[4]:= PD;
   Letras[5]:= PE;
   Letras[6]:= PF;
   Letras[7]:= PG;
   Letras[8]:= PH;
   Letras[9]:= PI;
```

```
  Letras[10]:= PJ;
  Letras[11]:= PK;
  Letras[12]:= PL;
  Letras[13]:= PM;
  Letras[14]:= PN;
  Letras[15]:= PO;
  Letras[16]:= PP;
  Letras[17]:= PQ;
  Letras[18]:= PR;
  Letras[19]:= PS;
  Letras[20]:= PT;
  Letras[21]:= PU;
  Letras[22]:= PV;
  Letras[23]:= PW;
  Letras[24]:= PX;
  Letras[25]:= PY;
  Letras[26]:= PZ;

  dirConfig:= ExtractFileDir(Application.ExeName)+'\config';
  if not DirectoryExists(dirConfig) then
    CreateDir(PChar(dirConfig));

  dirPal:= ExtractFileDir(Application.ExeName)+'\palavras';
  if not DirectoryExists(dirPal) then
    CreateDir(PChar(dirPal));

  dirUsu:= ExtractFileDir(Application.ExeName)+'\usuarios';
  if not DirectoryExists(dirUsu) then
    CreateDir(PChar(dirUsu));

  ListaPalavras.Directory:= dirPal;
  ListaPalavras.Update;
  CriarCorpo;
  jogoIniciado:= false;
  jogador:= TUsuario.create;

  teclado:= TTeclado.create(ExtractFileDir(A pplication.ExeName)+'\config\teclas.ini');

  if FileExists(ExtractFileDir(Application.ExeName)+'\config\vaso\0.bmp') then
    ImageVaso.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\vaso\0.bmp');
end;

procedure TFormPrincipal.FormA ctivate(Sender: TObject);
begin
  //selecionarUsuario;
  iniciaJogo;
end;

procedure TFormPrincipal.atualizaPosicaoAlfabeto;
var j: integer;
begin
  for j:=1 to 26 do begin
    if Letras[j].Enabled then
      Letras[j].Color:= clWhite;
  end;
```

```
  Letras[iLetraAtual].Color:= clLime;

end;

procedure TFormPrincipal.atualizaPosicaoAlfabeto(indice: integer);
var i: integer;
begin
  i:= indice;
  if (Letras[i].Enabled = false) then begin
    inc(i);
    atualizaPosicaoAlfabeto(i);
  end else begin

  end;

end;

procedure TFormPrincipal.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  if ((not fimDeJogo) and (jogoIniciado)) then begin

    //função da seta para cima
    if (Key = teclado.SetaCima) then begin
      repeat
        iLetraAtual:= iLetraAtual-9;
         if iLetraAtual < 1 then iLetraAtual:= iLetraAtual+26; //volta para a primeira posição ao chegar no fim
do array
      until Letras[iLetraAtual].Enabled = true;
      atualizaPosicaoAlfabeto;
    end;

    //função da seta para direita
    if (Key = teclado.SetaDir) then begin
      repeat
        Inc(iLetraAtual);
       if iLetraAtual > 26 then iLetraAtual:= 1; //volta para a primeira posição ao chegar no fim do array
      until Letras[iLetraAtual].Enabled = true;
      atualizaPosicaoAlfabeto;
    end;


    //função da seta para esquerda
    if (Key = teclado.SetaEsq) then begin
      repeat
        Dec(iLetraAtual);
        if iLetraAtual < 1 then iLetraAtual:= 26; //volta para a primeira posição ao chegar no fim do array
      until Letras[iLetraAtual].Enabled = true;
      atualizaPosicaoAlfabeto;
    end;

    //função da seta para baixo
    if (Key = teclado.SetaBaixo) then begin
      repeat
        iLetraAtual:= iLetraAtual+9;
```

```
        if iLetraAtual > 26 then iLetraAtual:= iLetraAtual-26; //volta para a primeira posição ao chegar no fim
do array
        until Letras[iLetraAtual].Enabled = true;
        atualizaPosicaoAlfabeto;
      end;

      //função da tecla SIM ou OK
      if (Key = teclado.BotSim) then begin
          ChecarUmaLetra(Letras[iLetraAtual].Name[2])
      end;
    end;

    if (Key = teclado.BotMenu) then begin
      Timer1.Enabled:= false;
      FormMenu.ShowModal;
      Timer1.Enabled:= true;
      if OPCAOMENU = 'sair' then Application.Terminate;
      if OPCAOMENU = 'novaPalavra' then iniciaJogo;
      if OPCAOMENU = 'reiniciar' then iniciaJogo(true);
    end;

end;

procedure TFormPrincipal.desabilitaBotao(indice: integer);
begin
  Letras[indice].Enabled:= false;
  Letras[indice].Color:= clBtnFace;
  Letras[indice].Font.Color:= clGray;
end;

//---------------------------------------------------------------------------

procedure TFormPrincipal.iniciaJogo(mesmaPalavra: boolean);
var i,k: integer;
begin
  //tira filtro de imagem para ver qtos categorias (pastas) existem
  ListaPalavras.Mask:= '*.*';
  if (ListaPalavras.Items.Count > 2) then begin  // o primeiro item da lista é [..] e o segundo é [.]
    //adicionar filtro para imagens
    //ListaPalavras.Mask:= '*.jpg;*.jpeg;*.bmp';
    if FileExists(ExtractFileDir(Application.ExeName)+'\config\vaso\0.bmp') then
      ImageVaso.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\vaso\0.bmp');
    if jogador.Nome <> '' then begin
      inicializaBotoesLetras;
      fimDeJogo:= false;

      ForcaImagem.visible:=true;
      if not mesmaPalavra then begin
        palavraSecreta:= escolhePalavra;
        for k:=1 to length(palavraSecreta) do
          palavraSecretaSemCaractEsp:= tirarCharAcentuados(palavraSecreta);
      end;

      aPalavra.caption:='';
      for i:=1 to length(palavraSecreta) do begin
```

```
        if (palavraSecreta[i] = ' ') then
          aPalavra.caption:=aPalavra.caption+'  '
        else if (palavraSecreta[i] = '-') then
          aPalavra.caption:=aPalavra.caption+'- '
        else
        aPalavra.caption:=aPalavra.caption+'_ ';

      end;

      LetrasSelecionadas:=[];
      letrasErradas.caption:='';
      letrasCertas.caption:='';
  //neo with ForcaImagem do canvas.rectangle(clientrect);
      {for i:=0 to HangManList.count-1 do drawapiece(color); }
      piececount:=0;
      labelFimJogo.visible:=false;
      iLetraAtual:= 1;
      ligarCronometro;
      jogoIniciado:= true;
    end else begin
      selecionarUsuario;
    end;

  end else begin
    showmessage('  Não há palavras cadastradas  ');
    Palavra1.Click;
  end;
end;

//---------------------------------------------------------------------------

procedure TFormPrincipal.CriarCorpo;
var
  piece:THPiece;
begin
  {Define all of the hangman pieces}
  piececount:=0;
  HangManList:=TList.create;
  piece:=THPiece.create;
  with piece do
  begin
    kind:=line;  {base}
    start:=point(200,350);
    stop:=point(50,350);
    HangmanList.add(piece);
  end;
  piece:=THPiece.create;
  with piece do
  begin
    kind:=line;   {upright}
    start:=point(50,350);
    stop:=point(50,50);
    HangmanList.add(piece);
  end;
  piece:=THPiece.create;
```

```
with piece do
begin
  kind:=line;  {crosspiece}
  start:=point(50,50);
  stop:=point(125,50);
  HangmanList.add(piece);
end;
piece:=THPiece.create;
with piece do
begin
  kind:=line;  {rope}
  start:=point(125,50);
  stop:=point(125,75);
  HangmanList.add(piece);
end;
piece:=THPiece.create;
with piece do
begin
  kind:=circle;  {head}
  start:=point(100,75);
  stop:=point(150,125);
  HangmanList.add(piece);
end;
piece:=THPiece.create;
with piece do
begin
  kind:=line;     {body}
  start:=point(125,125);
  stop:=point(125,225);
  HangmanList.add(piece);
end;

piece:=THPiece.create;
with piece do
begin
  kind:=line;  {arm1}
  start:=point(125,150);
  stop:=point(75,175);
  HangmanList.add(piece);
end;
piece:=THPiece.create;
with piece do
begin
  kind:=line; {arm2}
  start:=point(125,150);
  stop:=point(175,175);
  HangmanList.add(piece);
end;
piece:=THPiece.create;
with piece do
begin    {leg1}
  kind:=line;
  start:=point(125,225);
  stop:=point(100,300);
  HangmanList.add(piece);
```

```
 end;
 piece:=THPiece.create;
 with piece do
 begin     {leg2}
  kind:=line;
  start:=point(125,225);
  stop:=point(150,300);
  HangmanList.add(piece);
 end;
end;

//---------------------------------------------------------------------------

procedure TFormPrincipal.ChecarUmaLetra(ch: char);
var
 i:integer;
 s:string;
 bomChute:boolean;
 pal: TPalavra;
 chOriginal: char;
begin
 desabilitaBotao(iLetraAtual);
 bomChute:=false;
 if not (ch in LetrasSelecionadas)then begin
  LetrasSelecionadas:=LetrasSelecionadas+[ch];
  s:=aPalavra.caption;
  for i:=1 to length(palavraSecreta) do begin{see if the letter is in the word}
   if ch=palavraSecretaSemCaractEsp[i] then begin
    chOriginal:= palavraSecreta[i];
    s[2*i-1]:=chOriginal; {fill in the  letter in display}
    bomChute:=true;
    if not EhSubStr(ch, letrasCertas.caption)then
     letrasCertas.caption:=letrasCertas.caption+ch+',';
    PlaySound(PChar(ExtractFileDir(Application.ExeName)+'\config\som\certo.wav'), 1, SND_ASYNC);
   end
  end;
  if not bomChute then begin
   letrasErradas.caption:=letrasErradas.caption+ch+',';
   PlaySound(PChar(ExtractFileDir(Application.ExeName)+'\config\som\errado.wav'), 1, SND_ASYNC);
  end;
  aPalavra.caption:=s;
  if not bomChute then DesenharParteCorpo(clred);
  If pos('_',aPalavra.caption)=0 then begin {all underscores replaced by letters}
   fimDeJogo:= true;
   desligarCronometro;
   pal.Nome:= palavraSecreta;
   pal.LetrasErradas:= letrasErradas.Caption;
   pal.LetrasCertas:= LetrasCertas.Caption;
   pal.Acertou:= true;
   pal.tempo:= PainelTempo.Caption;
   pal.data:= dateToStr(now);
   jogador.addPalavra(pal);
   jogador.SaveToFile(pal);
   //showmessage('Você ganhou!');
   PlaySound(PChar(ExtractFileDir(Application.ExeName)+'\config\som\parabens.wav'), 1, SND_ASYNC);
```

```
    if FormDialog = nil then FormDialog:= FormDialog.Create(self);
    FormDialog.dialogAcertou(palavraSecreta);
    FormDialog.showmodal;
    FormMenu.ShowModal;

   end else If piececount=Hangmanlist.count then begin
    labelFimJogo.visible:=true;
    fimDeJogo:= true;
    desligarCronometro;
    pal.Nome:= palavraSecreta;
    pal.LetrasErradas:= letrasErradas.Caption;
    pal.LetrasCertas:= LetrasCertas.Caption;
    pal.Acertou:= false;
    pal.tempo:= PainelTempo.Caption;
    pal.data:= dateToStr(now);
    jogador.addPalavra(pal);
    jogador.SaveToFile(pal);
    //showmessage('Fim de Jogo!'+#13+'Você não descobriu a palavra secreta ('+palavraSecreta+')');
    PlaySound(PChar(ExtractFileDir(Application.ExeName)+'\config\som\nao.wav'), 1, SND_ASYNC);
    if FormDialog = nil then FormDialog:= FormDialog.Create(self);
    FormDialog.dialogErrou(palavraSecreta);
    FormDialog.showmodal;
    FormMenu.ShowModal;
   end;

 end else messagebeep(mb_IconExclamation);
end;

//---------------------------------------------------------------------------

procedure TFormPrincipal.DesenharParteCorpo(piececolor: TColor);
var
 piece:THPiece;
 w,h:integer;
begin
 inc(piececount);    {get to the next piece}

 if FileExists(ExtractFileDir(Application.ExeName)+'\config\vaso\'+IntToStr(piececount)+'.bmp') then

ImageVaso.Picture.LoadFromFile(ExtractFileDir(Application.ExeName)+'\config\vaso\'+IntToStr(piececount)+'.
bmp');

 if piececount<=HangManList.count then
 with ForcaImagem, canvas, piece do
 begin
  lastcolor:=piececolor;
  piece:=Hangmanlist[piececount-1];
  case piece.kind of
   line:
   begin
    pen.width:=4;
    pen.color:=piececolor;
    if piececolor=color {to erase face}
    then brush.color:=piececolor;
    moveto(start.x,start.y);
```

```
      lineto(stop.x,stop.y);
    end;
    circle: {The face}
    begin
      ellipse(start.x,start.y,stop.x,stop.y);
      w:=stop.x-start.x;
      h:=stop.y-start.y;
      {right eye}
      moveto(start.x+2*w div 10,
             start.y+3*h div 10);
      lineto(start.x+4*w div 10,
             start.y+3*h div 10);
      moveto(start.x+3*w div 10,
             start.y+2*h div 10);
      lineto(start.x+3*w div 10,
             start.y+4*h div 10);
      {left eye}
      {right eye}
      moveto(start.x+6*w div 10,
             start.y+3*h div 10);
      lineto(start.x+8*w div 10,
             start.y+3*h div 10);
      moveto(start.x+7*w div 10,
             start.y+2*h div 10);
      lineto(start.x+7*w div 10,
             start.y+4*h div 10);
      {mouth}
      ellipse(start.x+4*w div 10,
             start.y+7*h div 10,
             start.x+6*w div 10,
             start.y+8*h div 10);
    end;
  end; {case}
 end;
end;

procedure TFormPrincipal.clicarNaLetra(Sender: TObject);
begin

   if ((not fimDeJogo) and (jogoIniciado)) then begin
     if Sender is TPanel then begin
       iLetraAtual:= TPanel(Sender).tag;
       atualizaPosicaoAlfabeto;
       ChecarUmaLetra(Letras[iLetraAtual].Name[2]);
     end;
  end;
end;

procedure TFormPrincipal.inicializaBotoesLetras;
var i: integer;
begin
  for i:= 1 to 26 do begin
    Letras[i].Enabled:= true;
    Letras[i].Color:= clWhite;
    Letras[i].Font.color:= clBlack;
```

```
  end;
end;

procedure TFormPrincipal.Cadastrarusurio1Click(Sender: TObject);
begin
  Self.close;
end;

procedure TFormPrincipal.Usurio1Click(Sender: TObject);
begin
  if FormUsuario = nil then FormUsuario:= TFormUsuario.Create(nil);
  FormUsuario.ShowModal;
end;

procedure TFormPrincipal.Palavra1Click(Sender: TObject);
begin
  if FormPalavra = nil then FormPalavra:= TFormPalavra.Create(nil);
  FormPalavra.ShowModal;
end;


procedure TFormPrincipal.Facilitador1Click(Sender: TObject);
begin
  if FormFacilitadores = nil then FormFacilitadores:= TFormFacilitadores.Create(nil);
  FormFacilitadores.showmodal;
end;

procedure TFormPrincipal.SelecionarUsurio1Click(Sender: TObject);
begin
  selecionarUsuario;
end;

procedure TFormPrincipal.Dados1Click(Sender: TObject);
begin
  if FormReport = nil then FormReport:= TFormReport.Create(nil);
  FormReport.ShowModal;
end;

procedure TFormPrincipal.Sobre1Click(Sender: TObject);
begin
  if FormSobre = nil then FormSobre:= TFormSobre.Create(nil);
  FormSobre.ShowModal;
end;

procedure TFormPrincipal.NovoJogo1Click(Sender: TObject);
begin
  iniciaJogo;
end;

function TFormPrincipal.escolhePalavra: string;
var i: integer;
   nome: string;
begin
  Randomize;
  i:=Random(ListaPalavras.Items.Count-1);
```

```
  nome:= ExtractFileName(ListaPalavras.Items[i]);
  nome:= copy(nome,0,length(nome)-4);
  result:= nome;
  ImgGde.Picture.LoadFromFile(dirPal+'\'+ListaPalavras.Items[i]);
end;

procedure TFormPrincipal.selecionarUsuario;
var i,j: integer;
   flbCateg, flbPals: TFileListBox;
   catName: string;
begin
  if FormSelUsu = nil then FormSelUsu:= TFormSelUsu.Create(nil);
  FormSelUsu.ShowModal;
  if FileExists(FormSelUsu.UsuarioSelecionado) then begin
     jogador.setNome(FormSelUsu.UsuarioSelecionado);
    if dirPal <> FormSelUsu.dirPalSel then begin //usuario selecionaou uma categoria qq
       dirPal:= FormSelUsu.dirPalSel;
       ListaPalavras.Directory := dirPal;
       ListaPalavras.Update;
    end else begin // usuario selecionou TODAS AS CATEGORIAS
      //crio filelistbox para ler categorias, para tal, seto seu diretorio com o dir raiz das palavras
       flbCateg:= TFileListBox.Create(self);
       flbCateg.Parent:= self;
       flbCateg.FileType:= [ftDirectory];
       flbCateg.Mask:= '*.*';
       flbCateg.Directory:= dirPal;
       flbCateg.Update;
       //crio outro  filelistbox, para ler as palavras de cada categoria
       flbPals:= TFileListBox.Create(self);
       flbPals.Parent:= self;
       flbPals.FileType:= [ftNormal];

       //limpar a lista de palavras
       ListaPalavras.Clear;
       for i:= 2 to flbCateg.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
         //carrego na lista de palavras, todas as palavras de todas as categs
         catName:= copy(flbCateg.Items.Strings[i],2,length(flbCateg.Items.Strings[i])-2);
         flbPals.Directory:= dirPal+'\'+catName;
         flbPals.Update;
         for j:= 0 to flbPals.Items.Count-1 do begin
           ListaPalavras.Items.Add('\'+catName+'\'+flbPals.Items.Strings[j]);
         end;
       end;
       flbCateg.Free;
       flbPals.Free;
    end;
    jogador.LoadFromFile;
    LabelNome.Caption:= 'Olá '+jogador.Nome;
    iniciaJogo;
  end;
end;


procedure TFormPrincipal.Timer1Timer(Sender: TObject);
var a: string;
```

```
begin
  a:=TimeToStr(StrToTime(PainelTempo.Caption)+StrToTime('00:00:01'));
  PainelTempo.Caption:=a;
end;

procedure TFormPrincipal.ligarCronometro;
begin
  PainelTempo.Caption:= '00:00:00';
  timer1.Enabled:=True;
end;

procedure TFormPrincipal.desligarCronometro;
begin
  timer1.Enabled:= false;
end;


procedure TFormPrincipal.FormClose(Sender: TObject;
 var Action: TCloseAction);
begin
  if jogador <> nil then
    jogador.Free;
end;

procedure TFormPrincipal.ConfiguraodoTeclado1Click(Sender: TObject);
begin
  if FormTeclado = nil then
    FormTeclado:= TFormTeclado.Create(nil);
  FormTeclado.teclado:= teclado;
  FormTeclado.ShowModal;
  teclado:= FormTeclado.teclado;
end;


function TFormPrincipal.tirarCharAcentuados(str: String): String;
var
i: Integer;
begin
  for i := 1 to Length ( str ) do
    case str[i] of
     'á': str[i]:= 'a';
     'é': str[i]:= 'e';
     'í': str[i]:= 'i';
     'ó': str[i]:= 'o';
     'ú': str[i]:= 'u';
     'à': str[i]:= 'a';
     'è': str[i]:= 'e';
     'ì': str[i]:= 'i';
     'ò': str[i]:= 'o';
     'ù': str[i]:= 'u';
     'â': str[i]:= 'a';
     'ê': str[i]:= 'e';
     'î': str[i]:= 'i';
     'ô': str[i]:= 'o';
     'û': str[i]:= 'u';
```

```
      'ä': str[i]:= 'a';
      'ë': str[i]:= 'e';
      'ï': str[i]:= 'i';
      'ö': str[i]:= 'o';
      'ü': str[i]:= 'u';
      'ã': str[i]:= 'a';
      'õ': str[i]:= 'o';
      'ñ': str[i]:= 'n';
      'ç': str[i]:= 'c';
      'Á': str[i]:= 'A';
      'É': str[i]:= 'E';
      'Í': str[i]:= 'I';
      'Ó': str[i]:= 'O';
      'Ú': str[i]:= 'U';
      'À': str[i]:= 'A';
      'È': str[i]:= 'E';
      'Ì': str[i]:= 'I';
      'Ò': str[i]:= 'O';
      'Ù': str[i]:= 'U';
      'Â': str[i]:= 'A';
      'Ê': str[i]:= 'E';
      'Î': str[i]:= 'I';
      'Ô': str[i]:= 'O';
      'Û': str[i]:= 'U';
      'Ä': str[i]:= 'A';
      'Ë': str[i]:= 'E';
      'Ï': str[i]:= 'I';
      'Ö': str[i]:= 'O';
      'Ü': str[i]:= 'U';
      'Ã': str[i]:= 'A';
      'Õ': str[i]:= 'O';
      'Ñ': str[i]:= 'N';
      'Ç': str[i]:= 'C';
    end;
Result := str;
end;

end.
```

```
unit USelUsu;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, FileCtrl;

type
  TFormSelUsu = class(TForm)
    GroupBox2: TGroupBox;
    Label2: TLabel;
    ListaUsuario: TFileListBox;
    BotaoOK: TButton;
```

```
    BotaoCancelar: TButton;
    BotaoNovoUsu: TButton;
    ComboCateg: TComboBox;
    Label1: TLabel;
    CategoriasImg: TFileListBox;
    ComboFacilitador: TComboBox;
    ListFacitiladores: TFileListBox;
    Label3: TLabel;
    botaoNovoFAc: TButton;
    procedure FormCreate(Sender: TObject);
    procedure BotaoCancelarClick(Sender: TObject);
    procedure BotaoOKClick(Sender: TObject);
    procedure BotaoNovoUsuClick(Sender: TObject);
    procedure ListaUsuarioDblClick(Sender: TObject);
    procedure ListaUsuarioKeyPress(Sender: TObject; var Key: Char);
    procedure ListaUsuarioKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure FormActivate(Sender: TObject);
    procedure ComboFacilitadorChange(Sender: TObject);
    procedure botaoNovoFAcClick(Sender: TObject);
  private
    procedure AtualizaFacilitadores;
  public
    UsuarioSelecionado: string;
    dirPal,dirPalSel: string;
    dirUsu: string;
  end;

var
  FormSelUsu: TFormSelUsu;

implementation

uses UFormUsuario, UFormFacilitadores;

{$R *.DFM}

procedure TFormSelUsu.FormCreate(Sender: TObject);
begin
  dirUsu:= ExtractFileDir(Application.ExeName)+'\usuarios';
  ListFacitiladores.Directory:= dirUsu;
  dirPal:= ExtractFileDir(Application.ExeName)+'\palavras';
  CategoriasImg.Directory:= dirPal;
end;

procedure TFormSelUsu.FormActivate(Sender: TObject);
var i: integer;
    nomeDir: string;
begin
  ComboCateg.Clear;
  ComboCateg.Items.Add('Todas as categorias');
  for i:= 2 to CategoriasImg.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir:= copy(CategoriasImg.Items.Strings[i],2,length(CategoriasImg.Items.Strings[i])-2);
    ComboCateg.Items.Add(nomeDir);
  end;
```

```
  ComboCateg.ItemIndex := 0;

  ComboFacilitador.Clear;
  for i := 2 to ListFacitiladores.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir := copy(ListFacitiladores.Items.Strings[i],2,length(ListFacitiladores.Items.Strings[i])-2);
    ComboFacilitador.Items.Add(nomeDir);
  end;
  ComboFacilitador.ItemIndex := 0;
  ListaUsuario.Directory := dirUsu+'\'+ComboFacilitador.Text;
  ListaUsuario.Update;
end;

procedure TFormSelUsu.BotaoCancelarClick(Sender: TObject);
begin
  self.close;
end;


procedure TFormSelUsu.BotaoOKClick(Sender: TObject);
begin
  if ListaUsuario.ItemIndex = -1 then
    ShowMessage('Você deve selecionar um usuário')
  else if (ComboCateg.Text = '') then
    ShowMessage('Você deve selecionar uma categoria')
  else if (ComboFacilitador.Text = '') then
    ShowMessage('Você deve selecionar um facilitador')
  else begin
    UsuarioSelecionado := ListaUsuario.FileName;
    if (ComboCateg.Text <> 'Todas as categorias') then
      dirPalSel := dirpal+'\'+ComboCateg.Text
    else
      dirPalSel := dirpal;
    self.close;
  end
end;

procedure TFormSelUsu.BotaoNovoUsuClick(Sender: TObject);
begin
  if FormUsuario = nil then FormUsuario := TFormUsuario.Create(nil);
  FormUsuario.ShowModal;

  ListFacitiladores.Update;
  AtualizaFacilitadores;

  ListaUsuario.Directory := dirUsu+'\'+ComboFacilitador.Text;
  ListaUsuario.Update;
  ListaUsuario.Refresh;
  ListaUsuario.Repaint;
end;

procedure TFormSelUsu.botaoNovoFAcClick(Sender: TObject);
begin
  if FormFacilitadores = nil then FormFacilitadores := TFormFacilitadores.Create(nil);
  FormFacilitadores.showmodal;
```

```
  ListFacitiladores.Update;
  AtualizaFacilitadores;

  ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador.Text;
  ListaUsuario.Update;
  ListaUsuario.Refresh;
  ListaUsuario.Repaint;
end;

procedure TFormSelUsu.ListaUsuarioDblClick(Sender: TObject);
begin
  BotaoOK.Click;
end;

procedure TFormSelUsu.ListaUsuarioKeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then
    BotaoOK.Click;
end;

procedure TFormSelUsu.ListaUsuarioKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
  if key = 27 then
    Close;
end;

procedure TFormSelUsu.ComboFacilitadorChange(Sender: TObject);
begin
  ListaUsuario.Directory:= dirUsu+'\'+ComboFacilitador.Text;
  ListaUsuario.Update;
end;


procedure TFormSelUsu.AtualizaFacilitadores;
var i: integer;
  nomeDir: string;
begin
  ListFacitiladores.Update;
  ComboFacilitador.Clear;
  for i:= 2 to ListFacitiladores.Items.Count-1 do begin //começa em 2 pois o 0 é [.] e o 1 é [..]
    nomeDir:= copy(ListFacitiladores.Items.Strings[i],2,length(ListFacitiladores.Items.Strings[i])-2);
    ComboFacilitador.Items.Add(nomeDir);
  end;
  ComboFacilitador.ItemIndex:= 0;
end;

end.
```

```
unit USplash;

interface
```

```
uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 ExtCtrls;

type
 Tsplash = class(TForm)
  Timer1: TTimer;
  Image1: TImage;
  procedure Timer1Timer(Sender: TObject);
 private
  { Private declarations }
 public
  procedure Delay(msecs:integer);

end;

var
 splash: Tsplash;

implementation

{$R *.DFM}

procedure Tsplash.Delay(msecs: integer);
var
  FirstTickCount:longint;
begin
  Sleep(msecs);
//    FirstTickCount:=GetTickCount;
//   //Sleep(5000); repeat
//         Application.ProcessMessages; {allowing access to other
//                                    controls, etc.}
//    until ((GetTickCount-FirstTickCount) >= Longint(msecs));
end;

procedure Tsplash.Timer1Timer(Sender: TObject);
begin
  //Sleep(5000);
  Timer1.Enabled:= false;

  splash.Hide; {Esconde o FormAbertura}
  splash.close;
  splash.Free;
end;

end.
```

```
unit UTeclado;

interface

uses
  Controls, IniFiles, Sysutils;
```

```
type
 TTeclado = class

 private

 public
   urlFileKey: string;
   SetaCima:  word;
   SetaBaixo: word;
   SetaDir:   word;
   SetaEsq:   word;
   BotSim:    word;
   BotMenu:    word;
   vetKeys: array[1..255] of string;

   constructor create(urlFileTec: string);
   procedure LoadFromFile;
   procedure SaveToFile;
end;

implementation

{ TTeclado }

constructor TTeclado.create(urlFileTec: string);
begin
  urlFileKey:= urlFileTec;
  if FileExists(urlFileTec) then begin
    LoadFromFile;
  end else begin
    SetaCima:= 38;
    SetaBaixo:= 40;
    SetaDir:= 39;
    SetaEsq:= 37;
    BotSim:= 13;
    BotMenu:= 32;
    SaveToFile;
  end;
  vetKeys[16]:= '_';
  vetKeys[13]:= 'Enter';
  vetKeys[37]:= 'Seta Esquerda';
  vetKeys[38]:= 'Seta Acima';
  vetKeys[39]:= 'Seta Direita';
  vetKeys[40]:= 'Seta Abaixo';
  vetKeys[48]:= ')0';
  vetKeys[49]:= '!1';
  vetKeys[50]:= '@2';
  vetKeys[51]:= '#3';
  vetKeys[52]:= '$4';
  vetKeys[53]:= '5%';
  vetKeys[54]:= '¨6';
  vetKeys[55]:= '&7';
  vetKeys[56]:= '*8';
  vetKeys[57]:= '(9';
```

```
vetKeys[65]:= 'a';
vetKeys[66]:= 'b';
vetKeys[67]:= 'c';
vetKeys[68]:= 'd';
vetKeys[69]:= 'e';
vetKeys[70]:= 'f';
vetKeys[71]:= 'g';
vetKeys[72]:= 'h';
vetKeys[73]:= 'i';
vetKeys[74]:= 'j';
vetKeys[75]:= 'k';
vetKeys[76]:= 'l';
vetKeys[77]:= 'm';
vetKeys[78]:= 'n';
vetKeys[79]:= 'o';
vetKeys[80]:= 'p';
vetKeys[81]:= 'q';
vetKeys[82]:= 'r';
vetKeys[83]:= 's';
vetKeys[84]:= 't';
vetKeys[85]:= 'u';
vetKeys[86]:= 'v';
vetKeys[87]:= 'w';
vetKeys[88]:= 'x';
vetKeys[89]:= 'y';
vetKeys[90]:= 'z';
vetKeys[96]:= '0';
vetKeys[97]:= '1';
vetKeys[98]:= '2';
vetKeys[99]:= '3';
vetKeys[100]:= '4';
vetKeys[101]:= '5';
vetKeys[102]:= '6';
vetKeys[103]:= '7';
vetKeys[104]:= '8';
vetKeys[105]:= '9';
vetKeys[106]:= '*';
vetKeys[107]:= '+';
vetKeys[109]:= '-';
vetKeys[110]:= ',';
vetKeys[111]:= '/';
vetKeys[112]:= 'F1';
vetKeys[113]:= 'F2';
vetKeys[114]:= 'F3';
vetKeys[115]:= 'F4';
vetKeys[116]:= 'F5';
vetKeys[117]:= 'F6';
vetKeys[118]:= 'F7';
vetKeys[119]:= 'F8';
vetKeys[120]:= 'F9';
vetKeys[121]:= 'F10';
vetKeys[122]:= 'F11';
vetKeys[123]:= 'F12';
vetKeys[187]:= '+';
vetKeys[188]:= '<,';
```

```
  vetKeys[189]:= '-';
  vetKeys[190]:= '.>';
  vetKeys[191]:= ';:';
  vetKeys[192]:= '"';
  vetKeys[193]:= '/';
  vetKeys[194]:= '.';
  vetKeys[219]:= '`´';
  vetKeys[220]:= '}[';
  vetKeys[221]:= '}[';
  vetKeys[222]:= '~^';
  vetKeys[226]:= '|\';
  vetKeys[44]:= 'Print Scr';
  vetKeys[145]:= 'Scroll Lock';
  vetKeys[19]:= 'Pause';
  vetKeys[45]:= 'Insert';
  vetKeys[36]:= 'Home';
  vetKeys[33]:= 'PageUp';
  vetKeys[46]:= 'Delete';
  vetKeys[35]:= 'End';
  vetKeys[34]:= 'PageDown';
  vetKeys[144]:= 'Number Lock';
  vetKeys[27]:= 'Esc';
  vetKeys[20]:= 'Caps Lock';
  vetKeys[16]:= 'Shift';
  vetKeys[17]:= 'CTRL';
  vetKeys[91]:= 'janela win esq';
  vetKeys[18]:= 'Alt';
  vetKeys[32]:= 'Barra espaço';
  vetKeys[92]:= 'janela win dir';
  vetKeys[93]:= 'Pop up menu';


end;

procedure TTeclado.LoadFromFile;
var
ini: TIniFile;
begin

  if FileExists(urlFileKey) then begin
    ini := TIniFile.Create(urlFileKey);
    try
      SetaCima:= ini.ReadInteger('Teclas','SetaCima',38);
      SetaBaixo:=ini.ReadInteger('Teclas','SetaBaixo',40);
      SetaDir:=  ini.ReadInteger('Teclas','SetaDir',39);
      SetaEsq:=  ini.ReadInteger('Teclas','SetaEsq',37);
      BotSim:=   ini.ReadInteger('Teclas','BotSim',13);
      BotMenu:=   ini.ReadInteger('Teclas','BotNao',32);
    finally

    end;
    ini.Free;
  end;

end;
```

```
procedure TTeclado.SaveToFile;
var
ini: TIniFile;
begin
  if FileExists(urlFileKey) then
     DeleteFile(urlFileKey);

  ini := TIniFile.Create(urlFileKey);
  try
     ini.WriteInteger('Teclas','SetaCima',SetaCima);
     ini.WriteInteger('Teclas','SetaBaixo',SetaBaixo);
     ini.WriteInteger('Teclas','SetaDir',SetaDir);
     ini.WriteInteger('Teclas','SetaEsq',SetaEsq);
     ini.WriteInteger('Teclas','BotSim',BotSim);
     ini.WriteInteger('Teclas','BotNao',BotMenu);
  except

  end;
  ini.free;
end;

end.
```

```
unit UUsuario;

interface

uses
  Controls, IniFiles, Sysutils;

type
 TPalavra = record
   Nome : string;
   LetrasErradas: string;
   LetrasCertas:  string;
   Acertou: boolean;
   tempo: string;
   data: string;
 end;


 TUsuario = class

 private

 public
    Nome : string;
    Palavras: array of TPalavra;
    UrlArqUsu: string;
    constructor create;
    procedure setNome(urlFileUsu: string);
    function getNroPalavras: integer;
```

```
    procedure LoadFromFile(urlFileUsu: string = '');
    procedure SaveToFile(palav: TPalavra; urlFileUsu: string = ''); overload; //Apenas Adiciona uma palavra ao
marquivo do usuario
      procedure SaveToFile(urlFileUsu: string = ''); overload; // Limpa as informações existentes no arq do
usuario e´preenche com dados do usuario que estão na memoria do programa
   procedure addPalavra(pal: TPalavra);overload;
    procedure addPalavra(Nome: string; LetrasErradas: string; LetrasCertas: string; Acertou: boolean; tempo:
string; data: string);overload;
end;

implementation

{ TUsuario }

constructor TUsuario.create;
begin
  Nome:= '';
end;

procedure TUsuario.addPalavra(pal: TPalavra);
begin
  SetLength(Palavras,Length(Palavras)+1);
  Palavras[Length(Palavras)-1]:= pal; //o -1 é pq o vetor Palavras inicia em 0
end;

procedure TUsuario.addPalavra(Nome, LetrasErradas: string; LetrasCertas: string;
 Acertou: boolean; tempo, data: string);
begin
  SetLength(Palavras,Length(Palavras)+1);
  Palavras[Length(Palavras)].Nome:= Nome;
  Palavras[Length(Palavras)].LetrasErradas:= LetrasErradas;
  Palavras[Length(Palavras)].LetrasCertas:= LetrasCertas;
  Palavras[Length(Palavras)].Acertou:= Acertou;
  Palavras[Length(Palavras)].tempo:= tempo;
  Palavras[Length(Palavras)].data:= data;
end;

function TUsuario.getNroPalavras: integer;
begin
  result:= Length(Palavras);
end;

procedure TUsuario.LoadFromFile(urlFileUsu: string);
var
ini: TIniFile;
i, nPal: integer;
begin

  if urlFileUsu = '' then urlFileUsu:= UrlArqUsu;
  if FileExists(urlFileUsu) then begin
     ini := TIniFile.Create(urlFileUsu);
     try
        nPal:= ini.ReadInteger('arquivo','nroPal',0);
        SetLength(Palavras,nPal);
        for i:= 0 to nPal-1 do begin
```

```
        Palavras[i].Nome:= ini.ReadString('Palavra'+IntToStr(i),'palavraSecreta','');
        Palavras[i].LetrasErradas:= ini.ReadString('Palavra'+IntToStr(i),'letrasErradas','');
        Palavras[i].LetrasCertas:= ini.ReadString('Palavra'+IntToStr(i),'letrasCertas','');
        Palavras[i].Acertou:= ini.ReadBool('Palavra'+IntToStr(i),'acertouPalavra',false);
        Palavras[i].tempo:= ini.ReadString('Palavra'+IntToStr(i),'tempo','');
        Palavras[i].data:= ini.ReadString('Palavra'+IntToStr(i),'data','');
      end;
    finally

    end;
    ini.Free;
  end;
end;



procedure TUsuario.SaveToFile(urlFileUsu: string);
var
ini: TIniFile;
i: integer;
begin
  if urlFileUsu = '' then urlFileUsu:= UrlArqUsu;
  if FileExists(urlFileUsu) then begin
    DeleteFile(urlFileUsu);
    ini := TIniFile.Create(urlFileUsu);
    try
      ini.WriteInteger('arquivo','nroPal',getNroPalavras);
      for i:= 0 to getNroPalavras-1 do
        ini.WriteString('Palavra'+IntToStr(i),'palavraSecreta',palavras[i].Nome);
        ini.WriteString('Palavra'+IntToStr(i),'letrasErradas',palavras[i].LetrasErradas);
        ini.WriteString('Palavra'+IntToStr(i),'letrasCertas',palavras[i].LetrasCertas);
        ini.WriteBool  ('Palavra'+IntToStr(i),'acertouPalavra',palavras[i].Acertou);
        ini.WriteString('Palavra'+IntToStr(i),'tempo', palavras[i].tempo);
        ini.WriteString('Palavra'+IntToStr(i),'data', palavras[i].data);
    except

    end;
    ini.free;
  end;
end;

procedure TUsuario.SaveToFile(palav: TPalavra; urlFileUsu: string);
var
ini: TIniFile;
i: integer;
begin
  if urlFileUsu = '' then urlFileUsu:= UrlArqUsu;
  if FileExists(urlFileUsu) then begin
    ini := TIniFile.Create(urlFileUsu);
    try
      i:= ini.ReadInteger('arquivo','nroPal',0);
      //seção palavras
      ini.WriteString('Palavra'+IntToStr(i),'palavraSecreta',palav.Nome);
      ini.WriteString('Palavra'+IntToStr(i),'letrasErradas',palav.LetrasErradas);
```

```
        ini.WriteString('Palavra'+IntToStr(i),'letrasCertas',palav.LetrasCertas);
        ini.WriteBool  ('Palavra'+IntToStr(i),'acertouPalavra',palav.Acertou);
        ini.WriteString('Palavra'+IntToStr(i),'tempo', palav.tempo);
        ini.WriteString('Palavra'+IntToStr(i),'data', palav.data);

        inc(i);
        ini.WriteInteger('arquivo','nroPal',i);
      finally

      end;
      ini.Free;
    end;
  end;

procedure TUsuario.setNome(urlFileUsu: string);
begin
  Nome:= ExtractFileName(urlFileUsu);
  Nome:= copy(nome,0,length(nome)-4);
  UrlArqUsu:= urlFileUsu;
end;

end.
```