UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
Coordenação de Pós-Graduação em Informática

ESPECIFICAÇÃO DE UM SISTEMA DE PROTOTIPAGEM RÁPIDA DE INTERFACES HOMEM-MÁQUINA

SINONE DE OLIVEIRA BRANCO

CAMPINA GRANDE – PB
JANEIRO - 1990

Universidade Federal da Paraíba
Centro de Ciências e Tecnologia
Curso de Pós-Graduação em Informática

Especificação de um Sistema de Prototipagem Rápida de Interfaces Homem-Máquina

Simone de Oliveira Branco

Dissertação de Mestrado submetida à Coordenação do Curso de Pós-Graduação em Informática da Universidade Federal da Paraíba – Campus II, como parte dos requisitos necessários para a obtenção do grau de Mestre em Informática.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Engenharia de Software

Maria de Fátima Q. V. Turnell
Orientadora

Campina Grande – Paraíba - Brasil
Janeiro – 1990

B816e Branco, Simone de Oliveira
Especificacao de um sistema de prototipagem rapida de interfaces homem-maquina / Simone de Oliveira Branco. - Campina Grande, 1990.
97 f.

Dissertacao (Mestrado em Informatica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Interface Usuario Computador 2. Sistema de Prototipacao 3. Dissertacao - Informatica I. Turnell, Maria de Fatima Queiroz Vieira II. Universidade Federal da Paraiba - Campina Grande (PB)

CDU 004.5(043)

SIMONE DE OLIVEIRA BRANCO

ESPECIFICAÇÃO DE UM SISTEMA DE PROTOTIPAGEM RÁPIDA DE INTERFACES HOMEM-MÁQUINA

Dissertação apresentada ao Curso de MESTRADO EM INFORMÁTICA da Universidade Federal da Paraíba, em cumprimento às exigências para obtenção do Grau de Mestre.

Área de Concentração: Ciências Da Computação

Maria de Fátima Q. V. Turnell

MARIA DE FÁTIMA QUEIROZ VIEIRA TURNELL

(Orientadora)

PEDRO SÉRGIO NICOLLETTI - M.Sc

(Examinador)

PAULO ROBERTO C. DE ARAÚJO - M.Sc

(Examinador)

CAMPINA GRANDE - PB

JANEIRO - 1990

Aos meus pais,

Eunice Oliveira Branco

Mário Pereira Branco

AGRADECIMENTOS

O meu muito obrigado a Fátima Turnell, por ter sido muito mais que minha orientadora, tendo se integrado de corpo e alma ao meu trabalho, dando tudo de si para que ele fosse concluído, ultrapassando todos os empecilhos surgidos, sempre com paciência, dedicação e competência.

A Salomé Navaro e família pela amizade e gentileza em nos cederem suas instalações laboratoriais para confecção do trabalho.

A Cláudia Procópio que muito contribuiu no processo de tomada de decisões.

A Luca Sales pelo incentivo, força e confiança, constantemente expressos, pelas conversas e por todos os toques.

Aos professores e amigos: Marcos Sampaio, Peter e Roberto Faria, pela transmissão de seu valioso conhecimento, bem como aos demais professores.

À eficiência de Ana Lúcia Guimarães (Aninha), que sempre mostrou-se com boa vontade e habilidade para resolver os problemas de nós aluno, sem a qual nosso desempenho seria comprometido.

Aos meus pais (Eunice e Mário) e irmãos (Junior, Sinaida e Sinara) pelo carinho, estímulo e compreensão para comigo.

A todos os demais amigos (Valéria de Castro, Cláudia Queiroz, Claudete Alves, Laurino Campos, Ulisses, Isaque Douglas, Vicente Moreira, Luciano, Gabriela, Ed, Jorje César, Shirley, etc.) pelos toques e pela amizade.

A todos os funcionários do Departamento de Sistemas (Eliane, Liliane, Maggy, Terezinha, Vera, etc.) pela gentileza e disposição em contribuir.

RESUMO

Esta dissertação apresenta a especificação de um sistema de prototipagem rápida de interfaces homem-máquina – o sistema AGILE. Este sistema oferecerá ao projetista uma linguagem interativa e declarativa que facilitará e acelerará o processo de prototipagem. Comprando aos sistemas de prototipagem atualmente disponíveis oferece uma maior diversidade e funcionalidade nos recursos de prototipagem, e maior independência dos protótipos em relação ao sistema hospedeiro.

ABSTRACT

This dissertation presents the specification of a Rapid Prototyping System for computer-user interfaces, AGILE. This system is to offer the designer an interactive and declarative specification language which will simplify and speed up the prototyping process. Compared with the currently avilable prototyping systems, it will offer greater functionality.

CAPÍTULO I

1. INTRODUÇÃO

A preocupação com a forma de apresentar as informações para o usuário de computadores, tornando-a mais natural, surgiu com o aparecimento dos sistemas interativos e o uso de terminais de vídeo, e se intensificou a medida que estes se tornaram mais populares [HUCKLE 81]. No entanto esta preocupação se traduzia em esforços isolados dos projetistas de sistemas até que se consolidou em um esforço conjunto, hoje caracterizado pelas publicações e sociedades especializados na área de interfaces homem-máquina.

Ao se projetar uma interface, três fatores se destacam: as características do usuário, o tipo de aplicação(tarefa) e os recursos disponíveis [MARTIN73], [CARD 83].Percebe-se no entanto que, no projeto da maioria dos sistemas, ainda há uma preocupação maior com a aplicação do que com a sua interface, podendo acarretar em uma rejeição do sistema como um todo por parte do usuário.

Embora já existam ferramentas que apoiam tanto a especificação quanto o desenvolvimento destes projetos de interface, esta ainda é uma área com muito espaço para pesquisa [LOWGRE 88].

Uma das classes destas ferramentas é a dos Sistemas de Prototipagem rápida, alguns dos quais são apresentados no Capítulo 2. Estes sistemas aceleram a fase de especificação e antecipam a de validação, no ciclo de desenvolvimento de software (ver Fig. 1.1), na medida em que permitem uma avaliação junto ao usuário da proposta de projeto (ver Fig.2.2 na pág.16) antes mesmo que seja implementada, possibilitando modificações mais rápidas e menos onerosas.

Por outro lado, um recurso mais abrangente, também alvo de estudos da área de interfaces, são os Sistemas Gerenciadores de Interfaces de Usuários, SGIU

[MANHEIMER 89]. Estes sistemas, além de oferecerem recursos de prototipagem, são utilizados na geração do código da interface que será integrado à aplicação. O código assim gerado, gerencia a interface especificada liberando o projetista da aplicação de mais este aspecto da implementação do sistema, além de permitir a independência entre a aplicação e sua interface. Alguns destes sistemas e seus recursos são apresentados no Capítulo 2.

No Capítulo 3, apresentamos os resultados da análise fundamental que iniciou este trabalho, destacando as características de um sistema, inicialmente com recursos apenas para prototipagem rápida de interfaces, o sistema AGILE – mas projetado de forma modular, permitindo a ampliação de suas características funcionais para torná-lo um SGIU, capaz de gerar o código deste sistema, que influenciaram na especificação de sua interface.

O Capítulo 4, descreve a funcionalidade deste sistema, onde destacamos, além de suas funções, sua interface com o usuário (o projetista de interfaces) e como é possível transportar os resultados da prototipagem para o ambiente de projeto da aplicação.

O Capítulo 5, apresenta a especificação do software, utilizando a técnica HIPO, restringindo-se, entanto, aos diagramas hierárquicos representativos das funções do sistema. O detalhamento destes diagramas é apenas textual, tendo-se deixado a construção das tabelas como tarefa para o projetista do sistema. Ainda neste capítulo, abordam-se as restrições que deverão ser feitas durante a implementação no que diz respeito à escolha da linguagem de implementação e do sistema operacional.

No Capítulo 6, esboçamos a especificação do hardware necessário para satisfazer à especificação funcional.

Finalmente no Capítulo 7 são apresentadas as conclusões e discutidas as perspectivas futuras para o trabalho.

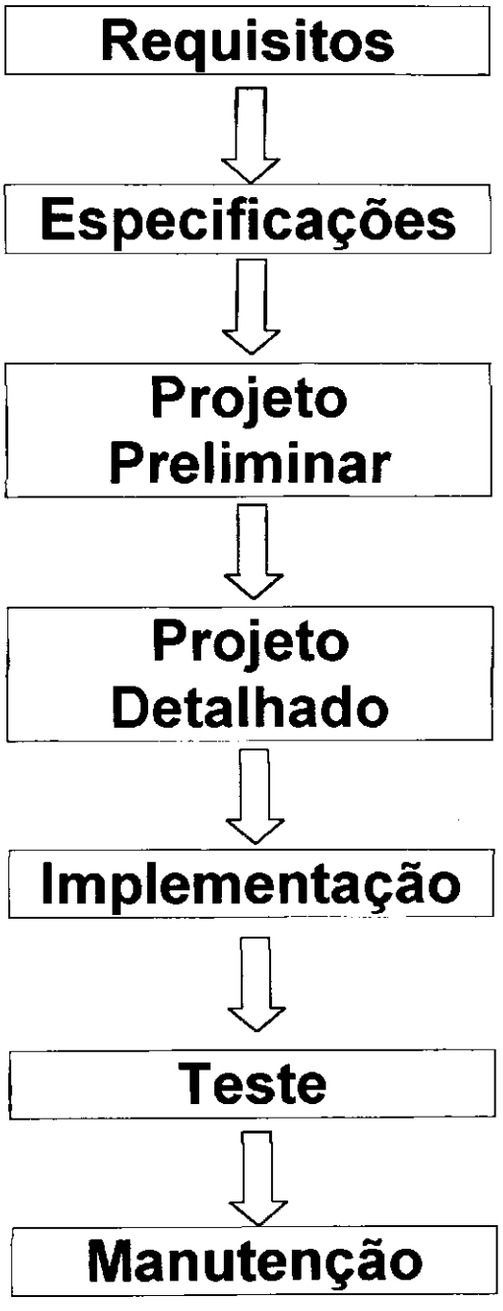

Fig. 1.1 – Ciclo de vida do desenvolvimento de software

CAPÍTULO II

2. SISTEMAS GERENCIADORES DE INTERFACE

A preocupação com a interface usuário-computador não é recente, há mais de 30 anos vem sendo demonstrada através de trabalhos nesta área [PERRY 89]. Um dos primeiros foi o "Sketchpd", tido como pioneiro nas interfaces gráficas de usuários, definidas como sendo a combinação de janelas, menus, ícones e mouse, cada vez mais populares em computadores pessoais e estações de trabalho. Foi construído em 1962 como tese de Ph.D. de Ivan E. Sutherland, pelo Massachussetts Institute of Technology.

O primeiro computador com interface de usuário "amigável" foi o STAR, desenvolvido pela Xerox's System Development Department e lançado no mercado em 1981.

Em 1984 o Macintosh da Apple Computer Inc., trouxe a interface "amgável" para milhares de usuários de computadores pessoais.

Hoje, com a popularidade das estações gráficas, reduz-se a demanda de conhecimento de programação por parte do projetista. Portanto, há um crescente interesse na programação da informação visual, cujos efeitos são evidenciados na redução do tempo nas curvas de aprendizado do usuário e na sua crescente produtividade.

Estes sistemas, em sua grande maioria, utilizam praticamente os mesmos conceitos de comunicação: menus, suporte ao conceito de janelas, dispositivos apontadores como o mouse, primitivas para o tratamento de gráficos bi-dimensionais e manipulação de ícones.

Esta evolução dos conceitos na área de interface, fez surgir um subconjunto dentro da tecnologia CASE (Computer Aided Software Engineering) que são os Sistemas de Gerência da Interface do Usuário, SGIU (User Interface Management Systems – UIMS).

2.1 – SGIU: O estado da técnica

Os SGIUs se caracterizam por oferecer recursos para o desenvolvimento e gerenciamento de interfaces que separam a interface do usuário da aplicação. Por gerenciamento da interface, entenda-se: como a interface com o usuário responde a eventos, como se comunica com a aplicação e, finalmente, como suas operações são sequenciadas.

Portanto, os SGIUs permitem uma maior flexibilidade no projeto das interfaces e simplificam o desenvolvimento da aplicação. Dentre estes recursos estão ferramentas para especificação, tais como editores de telas e de diálogos.

2.1.1 O Modelo de Seeheim

Um modelo que representa a arquitetura da maioria dos SGIU foi desenvolvido no workshop, realizado em Seeheim (Alemanha) em 1983 [LAWGREEN 85]. Este modelo, apresentado na Fig. 2.1, coloca entre a aplicação e o usuário, uma interface composta de três elementos:

Apresentação – responsável pela apresentação externa da interface com o usuário.

Controle de diálogo – define a estrutura do diálogo entre o usuário e o programa aplicativo.

Modelo da interface com a aplicação – representação da aplicação do ponto de vista da interface com o usuário.

Esta abordagem, embora adequada para descrever a maioria das interfaces atualmente disponíveis, começa a ser criticada quando utilizada na descrição de interfaces gráficas mais sofisticadas, como veremos mais adiante [LAWGREN 88].

Dentre os tópicos sendo pesquisados atualmente, destacam-se: as técnicas de especificação de diálogos, a relação entre o SGIU e a aplicação e as técnicas para validação de projeto.

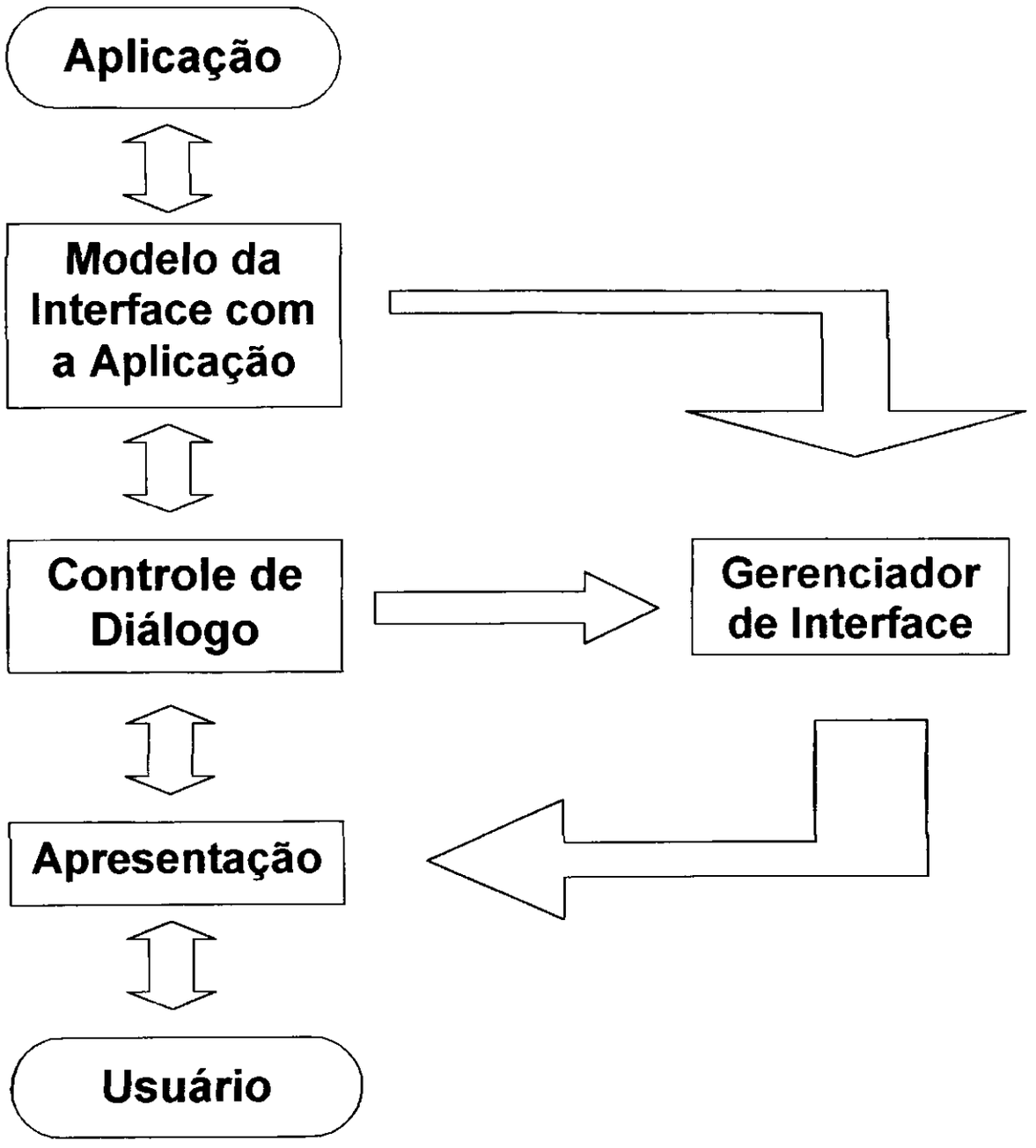

Fig. 2.1 – Modelo de Seeheim

2.1.2 Técnicas de especificação de diálogos

No início da década de 80, a estrutura dos diálogos era essencialmente linear, devido ao ambiente de interação se fundamentar em dispositivos como a teleimpressora, o monitor e o teclado. Isto levava estilos de diálogos também lineares, tais como comando e preenchimento de formulários. Hoje, com as interfaces gráficas mais interativas, este cenário mudou, levando os projetistas de SGIU a repensarem formas de especificação de diálogos além das já tradicionais.

As linguagens de especificação que tinham um caráter fortemente procedural (construções do tipo 'if-then-else' e chamadas de subrotinas), estão gradativamente migrando para uma estrutura declarativa, favorecendo um novo grupo de projetistas de interface, os engenheiros de fatores humanos, os quais não são, necessariamente, programadores experientes.

Nesta nova forma, a linguagem de especificação consiste de um conjunto de declarações dos objetos da interface de seus atributos, e de um analisador sintático que, a partir destas declarações, constrói a representação de interface (código executável).

Por conta destas mudanças, há críticas ao modelo de Seeheim, apresentado anteriormente [LOWGREN 88], e propostas de alternativas tais como o modelo cliente-servidor descrito mais adiante [MANHEIMER 89].

Se, por um lado, a especificação procedural traz a desvantagem de ser mais complexa para projetistas não programadores, por outro lado ela dá uma maior flexibilidade ao modelo da interface e aumenta sua capacidade de seqüenciar operações.

Por sua vez, a especificação declarativa, embora limite o escopo do modelo, por se basear em estruturas simples, pode ser mais facilmente aprendida por usuários não programadores aumentando a velocidade com que estes elaboram a especificação.

2.1.3 Relação entre o SGIU e a aplicação

A independência entre a interface e a aplicação vai depender do número de operações da interface que podem ser iniciadas pela aplicação. Isto é, algumas aplicações requerem um número de interações com a interface que permitem a separação, enquanto outras exigem um controle intensivo sobre a interface, dificultando a independência.

Portanto, há uma necessidade de se pesquisar arquiteturas de SGIUs que suportem uma intervenção abrangente e eficiente da aplicação. Uma destas propostas é a do modelo cliente-servidor, onde o SGIU é o servidor para as aplicações (clientes), permitindo que as aplicações definam e solicitem as ações da interface que serão executadas.

2.1.4 Técnicas para validação de projeto

Estas técnicas objetivaram apoiar o projetista nesta fase do ciclo de desenvolvimento do software, dando-lhe recursos para aprovar ou não decisões de projeto, com base em critérios tais como consistência, tempo de resposta, aceitação do usuário, etc.

Uma destas técnicas, a validação através de prototipagem, é discutida em detalhes na seção seguinte. Uma outra técnica consiste em registrar aspectos da interação com o usuário ao longo de uma ou mais seções visando dar maior subsídio a avaliação de certos aspectos da interface.

Já existem vários pacotes SGIU, alguns experimentais outros comercialmente disponíveis. A seguir, listamos alguns deles:

ESPEDI – Sistema de Especificação de Diálogos

Trata-se de uma ferramenta do SGIU-PUC, desenvolvido pelo Departamento de Informática da PUC – RJ [MELO 85]. Sua função é oferecer recursos para especificação de diálogo: linguagem formal de especificação, documentação da especificação efetuada e geração de um código de tabelas com o código do diálogo especificado para uso de outra ferramenta que compõe o SGIU-PUC (o interpretador do diálogo).

Toolbox (Macintosh):

Produzido pela Apple Computer Inc., traz uma interface baseada em menus, com uso de janelas, ícones e o mouse como dispositivo selecionador. Este pacote facilita o projeto de interfaces fundamentadas nestes recursos desenvolvidos para o ambiente Macintosh.

Windows:

Criado pela Microsoft, é uma interface amigável entre o DOS e o usuário, estruturada em menus [JAMSA 88]. No entanto, pode ser categorizada como uma ferramenta para projeto de diálogos que impõe um ambiente de interação e as características dos diálogos gerados.

LUIS (Lockheed User Interface System);

Este SGIU programado em C, roda em várias estações gráficas (SUN, Apollo, Macintosh). Suporta prototipagem, gerenciamento da interface e validação do projeto [MANHEIMER 89].

Sua linguagem de especificação de diálogos inclui componentes declarativos e procedurais objetivando maior flexibilidade para os usuários. A partir de estruturas de objeto, permite a criação de telas, janelas e diálogos baseados em menus, textos, gráficos, além de apresentar mensagens e instruções através de textos e imagens.

ITS (Interactive Transaction Systems):

SGIU produzido pela IBM Research, permite que uma mesma aplicação seja utilizada por uma variedade de diferentes diálogos ou rotinas de interação [WIECHA 89].

A idéia é codificar a experiência de projetistas de interfaces (especialistas em fatores humanos e artes gráficas) em regras, tal que as interfaces possam ser automaticamente geradas para vários grupos de usuários e ambientes de hardware. O conhecimento sobre os tipos de dados da aplicação seriam retidos, permitindo que a interface interaja mais inteligentemente com os usuários, enquanto permanece separada da aplicação.

Michey:

É um SGIU baseado na linguagem de programação Pascal, o qual mapeia a interface do usuário nos recursos do Macintosh (Apple), permitindo a geração de interface de usuário no ambiente Macintosh, sem precisar ser um programador experiente do mesmo [OLSEN 89].

Statemaster:

Um SGIU que utiliza Diagramas de Estado para Prototipagem e Implementação de interfaces [WELLNER 89]. Implementado em C++, especifica diálogos através de textos e gráficos. Estas especificações são independentes do hardware visando portabilidade.

Alguns destes sistemas, além de serem usados para desenvolvimento do código da interface, favorecem a prototipagem rápida, uma vez que reduzem o tempo necessário para desenvolvimento das interfaces.

2.2 Sistemas de Prototipagem Rápida

Prototipagem rápida é o processo de construir e avaliar rapidamente uma série de protótipos, que permite o teste de muitos aspectos de um diálogo antes de sua integração com a aplicação.

Os benefícios potenciais da prototipagem dependem criticamente da habilidade para modificar o comportamento do protótipo com muito menos esforço do que aquele necessário para modificar o software de produção [LUQI 89].

Prototipagem rápida assistida por computador (CAPS – Computer Aided Prototyping System) aumenta a eficiência e precisão do processo de desenvolvimento interativo, na medida em que auxilia o projetista na construção e execução do protótipo, rápida e sistematicamente, reduzindo seus esforços de produção e adaptação às necessidades do usuário.

"Em uma área carente de métodos comprovados para avaliação de seus resultados, a avaliação e o refinamento de projetos através de testes empíricos de protótipos é bastante válido" [MANHEIMER 89].

A utilização de prototipagem rápida traz uma série de vantagens ao projeto de interfaces:

- Rompe a barreira entre o usuário e o responsável pelo desenvolvimento, na medida em que o usuário não precisa mais ler as especificações feitas pelo projetista e este não terá mais tantas dificuldades em atender as necessidades do usuário;
- Permite uma avaliação do projeto através da experiência prática do usuário com o sistema;
- Propicia uma abordagem evolucionária do processo de desenvolvimento baseada no feedback do usuário (ver Fig. 2.2).

As ferramentas comuns aos sistemas de prototipagem de interface com o usuário são os construtores de telas e editores de diálogo.

Dentre os sistemas desenvolvidos apenas para prototipagem-rápida destacamos o Oregon Speedcode Universe (OSU),baseado na interface padrão do Macintosh que oferece um conjunto de ferramentas para criação,manipulação e execução de protótipos [LEWIS 89].

Umas das falhas comuns aos sistemas de prototipagem é tornar o protótipo o objetivo final ao invés de um primeiro passo no processo de avaliação de projetos. Para que isto possa ser mudado, é necessário que estes sistemas incorporem recursos que facilitem a avaliação dos resultados, tais como a capacidade de registrar o tempo de resposta do usuário em partes críticas da interface, o número de teclas acionadas, etc., de modo que possam ser, posteriormente analisadas pelo projetista, enriquecendo sua avaliação da reação do usuário ao projeto.

Outros problemas encontrados são listados a seguir:

- Dificuldade na utilização (o usuário deve aprender uma linguagem de prototipagem);
- Pouca funcionalidade (a maioria dos sistemas limita-se a menus e caixas de diálogo);
- Não há portabilidade (os sistemas dependem do hospedeiro);
- Dificuldade em separar a interface da aplicação.

Visando corrigir algumas destas falhas, apresentamos no capítulo seguinte, o resultado de nossa análise funcional.

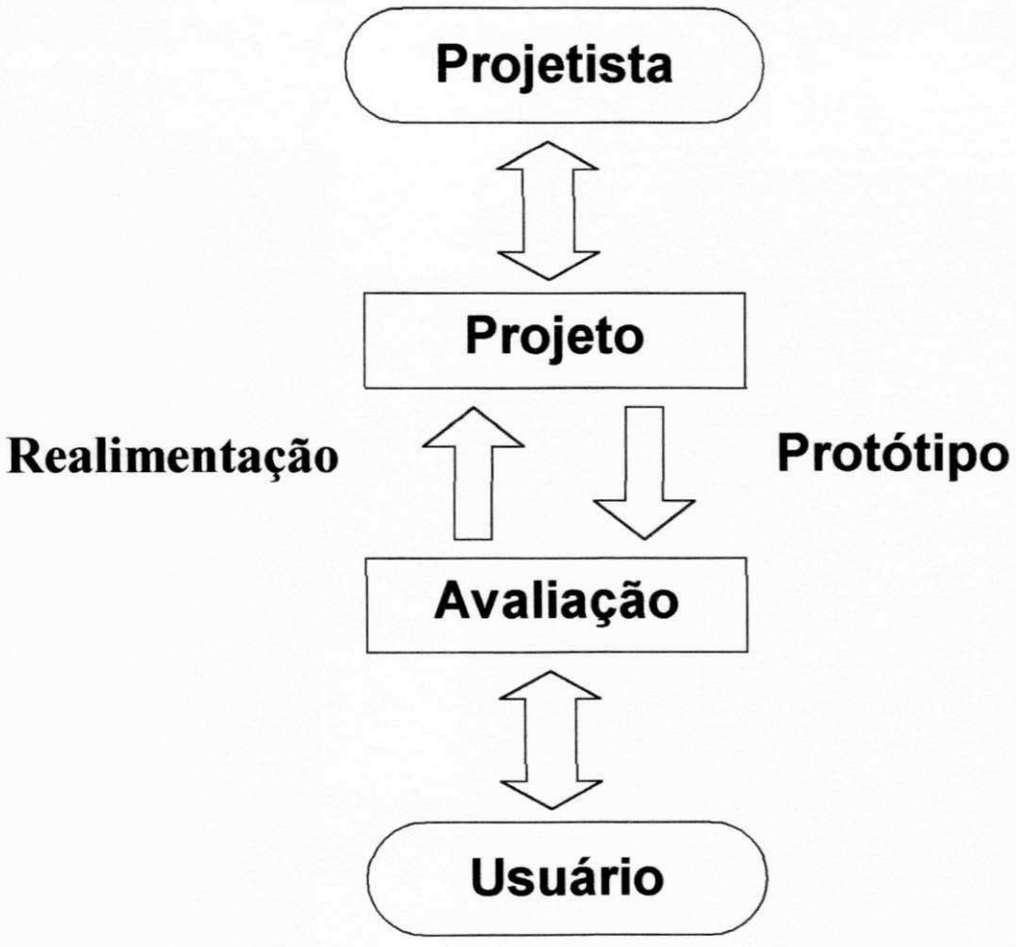

Fig. 2.2 – Prototipagem – Abordagem do Projeto Interativo

CAPÍTULO III

3. SISTEMA AGILE: ANÁLISE FUNCIONAL

3.1 O que é o sistema

Trata-se de um sistema de prototipagem rápida, com recursos para especificação de diálogos e edição de telas, cuja linguagem de especificação de diálogos é interativa e orientada (baseada em menus). Nele se destaca a diversidade de estilos de diálogo e de dispositivos periféricos disponíveis ao projeto do protótipo da interface.

Nesta abordagem procurou se explorar as vantagens de se extrair uma especificação do projetista, de forma declarativa, através da manipulação direta de um conjunto de objetos existentes na biblioteca de estilos de diálogo, mas garantindo-lhes a possibilidade de introduzir novos objetos e/ou novas relações entre os objetos já existentes. Exemplos de classes de objetos da interface são os objetos gráficos (ícones, janelas, etc.) e objetos texto (menus, mensagens, estilos de diálogo, etc.).

O sistema apresenta uma série de listas (menus) das quais é selecionada a coleção apropriada de objetos e ações e, através do preenchimento de formulários, o usuário define seu protótipo. As ações são os comportamentos definidos para os objetos de acordo com ao eventos gerados pela interação do usuário com a interface e pelo modelo da aplicação.

A hierarquia da conversação é especificada através do encadeamento de ações (em abordagem top-down) durante a especificação dos diálogos. Este sequenciamento é coordenado durante a simulação, pelo módulo Gerenciador de Diálogos, que interage com o Gerenciador de Teclas e o Gerenciador de Periféricos.

O sistema conta ainda com um módulo Gerador Automático de documentação que produz informações sobre a especificação efetuada, fornecendo ao projetista dados que auxiliam na análise do projeto e posterior codificação da interface. A documentação

que pode ser editada pelo projetista, incluindo seus comentários e observações, contém informações sobre:

- Descrição dos objetos especificados;
- Descrição do diálogo;
- Layout das telas;
- Ações ativadas no diálogo e os eventos que os originam.

Uma vez especificado, procede-se à geração do código do protótipo para o módulo simulador. Este processo ocorre em duas etapas: a primeira é a fase de validação da especificação, onde se detectam inconsistências tais como:

- Objetos referenciados e não definidos;
- Objetos definidos e não referenciados;
- Sequenciamento da interação equivocado.

Na Segunda etapa, é adicionado um módulo gerador de programa que, automaticamente, escreve um código fonte correspondente ao protótipo especificado. A geração deste código é feita através de análise sintática e da análise das especificações produzidas.

Este código fonte, facilita a integração do protótipo com os outros componentes da biblioteca do sistema e até com programas escritos em outras linguagens. Feita a integração, o código é compilado e o executável fica disponível para o usuário durante a simulação.

A arquitetura do sistema é apresentada na Fig. 3.1. nela são destacados os três componentes básicos do modelo de Seeheim: o módulo de apresentação, o módulo de controle de diálogo e o modelo da aplicação.

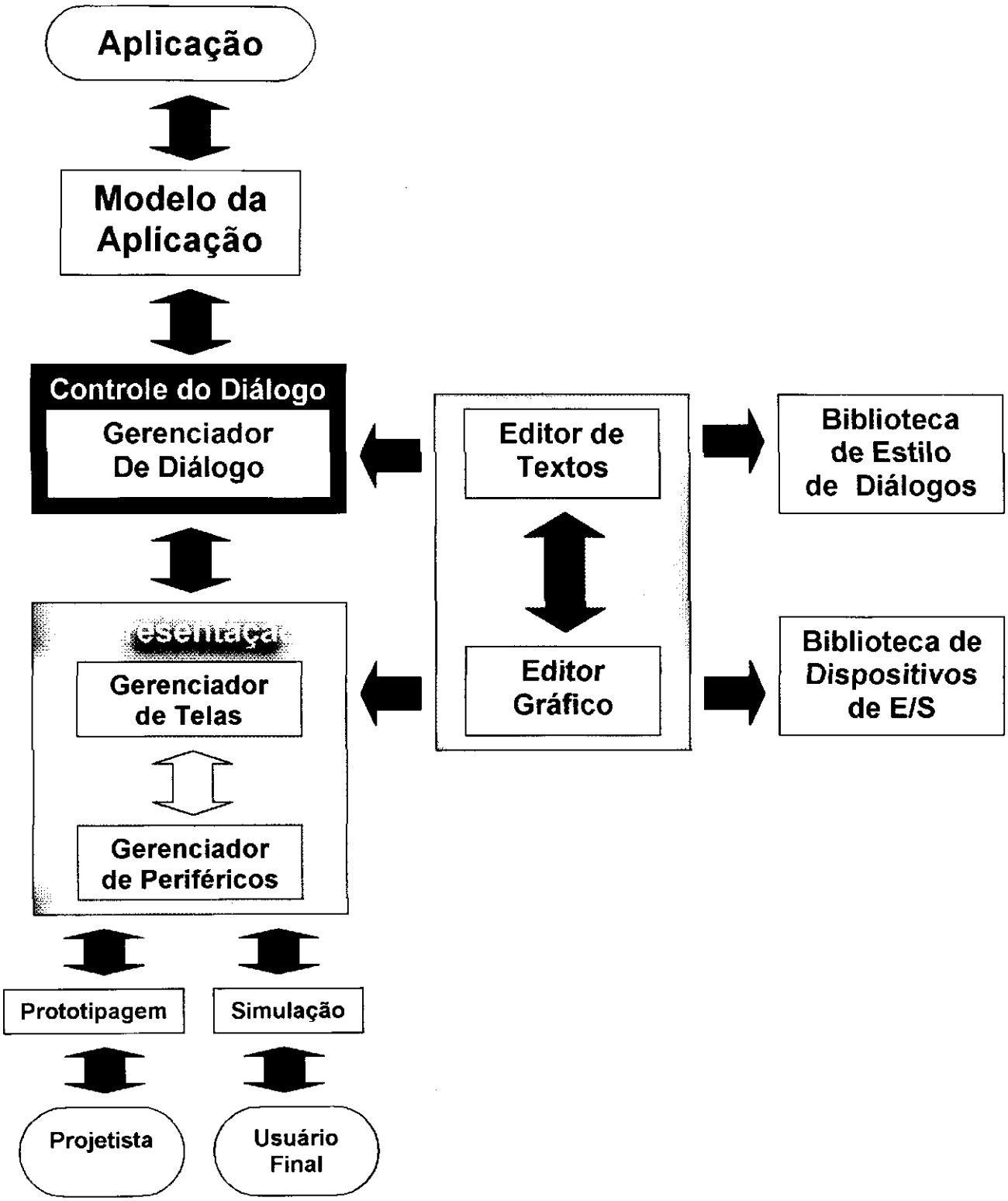

Fig. 3.1 – Arquitetura do Sistema **AGILE**

3.2 – A que se propõe

Este sistema, conforme já foi descrito na seção anterior, se propõe a oferecer recursos de prototipagem para o projetista de interfaces. No entanto, se restringe à prototipagem vazia, isto é, a interface com a aplicação é completamente simulada, mas o código da aplicação não é integrado ao protótipo de interface [LEWIS 89].

Pretendemos atingir os seguintes objetivos:

- Oferecer uma estrutura modular, visando tanto a portabilidade de seus recursos para o projeto das aplicações (ex. editor gráfico), quanto a integração de novos recursos à sua estrutura (processador de linguagem natural, drivers para novos dispositivos periféricos, etc.), à medida em que se façam disponíveis;
- Desenvolver uma metodologia de projetos de interface, que cuide de aspectos, tais como métodos de ajuda e de recuperação de erros;
- Oferecer uma documentação do protótipo especificado.

Numa versão futura, será produzido, também, o código do protótipo da interface gerada, o qual poderá ser integrado ao código da aplicação em desenvolvimento.

Pretende-se ainda:

- Oferecer recursos para a avaliação do protótipo especificado;
- Construir uma base de conhecimento dos projetos de interfaces especificados, utilizando o sistema.

3.3 – Análise do usuário

Em um sistema de prototipagem há duas classes de usuários: o projetista e o usuário da aplicação para o qual a interface foi prototipada. A este último chamaremos de usuário final, e a análise de suas características é da responsabilidade do projetista.

O projetista de interfaces pode ainda se classificar dentre um dos seguintes grupos: engenheiro de fatores humanos e projetista de software [MANHEIMER 89].

O projetista de software caracteriza-se por ter um conhecimento considerável de computação e da aplicação, e por estar mais preocupado com o desempenho do sistemas de prototipagem e com os recursos oferecidos à aplicação para manipular a interface gerada pelo sistema. Por outro lado, o engenheiro de fatores humanos exige uma maior variedade de estilos de interface e ferramentas de especificação de nível mais alto.

Visando atender usuários não necessariamente especialistas em programação, e até mesmo melhorar a produtividade (durante a especificação) daqueles mais experientes, optamos por uma interface com características declarativas.

Ao traçarmos o perfil do projetista que usará o sistema, foram considerados os seguintes aspectos:

- A experiência em programação não é exigida, uma vez que a linguagem de interação é declarativa. O nível de conhecimento do sistema de prototipagem vai enquadrá-lo no nível de interação adequado – com ou sem a orientação de menus;
- A freqüência de utilização do sistema vai ser um dos indicadores da necessidade de recorrer aos recursos de ajuda disponíveis.

Assim exigiu-se deste usuário apenas um bom conhecimento da aplicação cuja interface deve ser prototipada, de modo a explorar mais adequadamente os recursos de prototipagem disponíveis.

No capítulo seguinte, apresentamos a descrição funcional do sistema proposto.

CAPÍTULO IV

4.0 DESCRIÇÃO FUNCIONAL

4.1 interface com o usuário

Considerando o perfil do usuário, descrito no capítulo anterior, passaremos a descrever a interface do sistema AGILE.

4.1.1 Entrada no sistema

Uma vez no sistema operacional, o projetista deverá digitar "AGILE" e teclar <ENTER>. Esse procedimento trará a tela de abertura (ver figura A1), a qual mostrará a identificação do sistema e do ambiente default instalado. Em seguida perguntará ao usuário se ele deseja modificá-lo e se necessita do tutorial, o qual consiste de um texto explicativo sobre os recursos do sistema e de como interagir com ele (ver seção 4.1.6).

Definido o ambiente de interação, seja aceitando o default ou modificando-o, o usuário é apresentado a uma tela, cujo layout é função do ambiente de interação, do estilo de diálogo e do nível de ajuda especificados. A seguir, ele deverá selecionar o diretório e nele o arquivo onde se encontra o protótipo, para então selecionar o ambiente desejado: prototipagem ou simulação.

4.1.2 Segurança do sistema

Considerando-se que vários projetos podem estar sendo prototipados concomitantemente, colocou-se à disposição do usuário a opção de um esquema de segurança de acesso às informações sobre cada um destes protótipos.

A proteção das informações sobre um protótipo, consiste em criar uma senha de acesso durante a criação do diretório correspondente. A partir de então, só será possível acessar ou remover os arquivos deste diretório após o reconhecimento da respectiva senha.

Quanto à integridade dos dados de um protótipo, esta é assegurada através dos recursos que possibilitam a recuperação de erros cometidos durante a prototipagem (ver seção 4.1.8).

4.1.3 Ambientes de integração

As duas etapas: prototipagem e simulação são desenvolvidas em dois ambientes distintos de integração com o usuário, os quais são descritos abaixo:

- Ambiente de prototipagem: onde são definidas as características do protótipo da interface;
- Ambiente de simulação: onde o protótipo tem suas funções simuladas.

Visando uma clara diferenciação entre estes dois ambientes, existem layouts de tela específicos da prototipagem, um conjunto de teclas de função ativas (ver apêndice C) e um conjunto de dispositivos de entrada/saída de informação (ver seção 6.1), característicos da interação do projetistas com o sistema neste ambiente.

Por outro lado, durante a simulação, o layout da tela é definido pelo projetista da interface, portanto toda a tela fica disponível para simular o projeto. As teclas de função, pertencentes ao sistema AGILE, ficam disponíveis para a utilização no contexto do protótipo, enquanto este estiver sendo definido. Durante esta fase (prototipagem), as funções do AGILE, que estão associadas às teclas de função, devem ser solicitadas através da digitação de seus nomes. Quanto aos dispositivos periférico ativos, estes também deverão ter sido definidos durante a prototipagem (ver seção 6.2).

4.1.4 Estilos de diálogo

Levando em conta a diversidade das características dos projetistas quanto à necessidade de ajuda durante a navegação no sistema, são oferecidos os dois níveis de interação, descritos a seguir:

4.1.4.1 Modo Dirigido

Baseia-se numa combinação de menus, formulários e prompts que guiam o usuário durante sua tomada de decisões. É voltado para projetistas menos experientes com o sistema e que precisam ser guiados ao longo da interação.

Este é o modo default, podendo ser alternado com o Modo Direto (descrito a seguir) em qualquer instante da interação, desde que solicitado pelo usuário. Para isto deve-se ativar a função de AJUDA (ver seção 4.1.7) e selecionar o nível desejado.

4.1.4.2 Modo Direto

Este modo é voltado para projetistas mais experientes com o sistema, os quais já têm bastante familiaridade com suas funções e recursos, conhecendo seus comandos e a forma como eles se encadeiam.

Baseia-se em uma sintaxe de comandos (ver Apêndice B) completada pelo preenchimento de formulário nos níveis de maior detalhamento da informação.

4.1.5 Layouts de telas

Como foi mencionado anteriormente, o layout das telas é função do ambiente de interação (prototipagem ou simulação) e dos recursos de ajuda solicitados.

Na apresentação de informações, o sistema utiliza um misto de mosaico e janelas. No primeiro caso, a tela é dividida em áreas que se justapõem, enquanto que as janelas são áreas da tela que podem se sobrepor.

A apresentação em mosaicos predomina, visto que favorece a homogeneidade e consistência no projeto de telas, liberando o usuário da preocupação em localizar a informação, deixando-o livre para concentrar-se no teor desta informação durante a tomada de decisões [MARTIN 73], [GALITZ 81].

Por outro lado, para informações ocasionais e temporárias, tais como mensagens de erro, menus e textos de ajuda, optou-se por janelas que se superpõem ao mosaico, dando-lhes destaque sem forçar um fracionamento maior da tela, o que ocorreria caso estas áreas fossem permanentemente reservadas na forma de mosaico. Estas janelas são fechadas, no

caso das mensagens, tão logo a ação corretiva do erro tenha sido concluída satisfatoriamente. Quanto aos textos de ajuda, como o tutorial, a decisão de retirá-los cabe ao usuário (ver seção 4.1.7). E, finalmente, no caso dos menus, o texto é retirado pelo sistema após a seleção ter sido efetivada.

No sistema AGILE, são definidas áreas associadas a um conjunto de informações específico. Abaixo, descreveremos estas áreas:

LINHA DE STATUS:

Situada na primeira linha da tela (ver figura A.2), contém informações sobre:

- Ambiente de interação;
- Identificação do protótipo;
- Função sendo executada.

Esta linha está presente durante a prototipagem independentemente do nível de ajuda, e tem como propósito situar o usuário quanto à posição em que se encontra na interação. A linha de status também pode ser apresentada durante a simulação, se solicitada pelo usuário.

MENU DE FUNÇÕES:

Situado abaixo do menu de status, este menu de barras apresenta uma relação das funções válidas naquele instante da interação. Uma vez sendo escolhida uma função, é apresentado um menu pull-down com as sub-funções correspondentes.

ÁREA DE TRABALHO:

Esta área é reservada para a entrada das informações do protótipo tais como o projeto de telas e diálogo. Estas informações são solicitadas pelo sistema através de prompts e formulários no método dirigido. São, ainda, superpostas nesta área as janelas com mensagens e tutorial, que podem ser removidas com tecla ESC.

LINHA DE COMANDO:

Situada abaixo da área de trabalho, ecoa os comandos digitados pelo projetista, no modo direto de interação (ver Fig. A.4).

MENU DE TECLAS DE FUNÇÃO:

Apresentado sob solicitação do usuário, situa-se nas duas últimas linhas da tela, mostrando as teclas válidas naquele instante da interação (F1 a F10 e as funções correspondentes).

Assim, os layouts apresentados nas figuras A.2, A.3, A.4, A.5 E A.6, representam combinações dos ambientes de interação (prototipagem ou simulação) com os recursos de ajuda disponíveis.

4.1.6 O acesso às funções

4.1.6.1 Modo dirigido

Para acessar as funções do sistema, o projetista deve selecionar em um diretório os arquivos correspondentes ao seu projeto. Este procedimento poderia estar condicionado à entrada de uma senha que lhe seria solicitada.

Após a seleção do protótipo, é exibido o menu com as funções que podem ser executadas neste instante da interação. A partir de então, o acesso consiste em selecionar do menu a função desejada e a seguir, nos menus subsequentes, selecionar as subfunções.

4.1.6.2 Modo Direto

Neste nível de interação, o acesso a uma função consiste em digitar o comando correspondente, que será então ecoado na linha de comandos. A sintaxe destes comandos é apresentada no Apêndice B.

4.1.6.3 Como sair de uma função

Para finalizar a execução de uma função, tecla-se F9 no modo dirigido ou FIMFUNC no modo direto. Esta ação devolve o usuário à raiz da árvore de funções, isto é, no modo dirigido é apresentada a tela com as funções válidas, enquanto no modo direto o usuário pode entrar o comando correspondente a qualquer destas funções.

É possível finalizar a execução de uma função deixando-se pendências nas definições que podem ser resolvidas posteriormente. Caso estas venham a ser esquecidas pelo projetista, serão detectadas durante a geração do protótipo ou quando estas decisões se fizerem necessárias ainda na fase de prototipagem, durante a execução de uma outra função interdependente.

É possível ainda sair de uma função interrompendo-a temporariamente através de F5 ou INTERROMPE ou abortando sua execução, através de F7 ou ABORTA.

A interrupção de uma função oferece a possibilidade ao usuário de salvar o contexto da função sendo executada e acessar uma outra. Uma vez concluída a execução desta outra função, o sistema retorna a interação no ponto onde foi interrompida a função original.

4.1.7 Níveis de ajuda

Todos os níveis de ajuda disponíveis no sistema, são acessados através da função AJUDA(F1). Uma vez ativada, esta função exibe para o usuário um menu com as opções de ajuda disponíveis no sistema. Ao iniciar a interação, o sistema se encontra no estado default mostrado a seguir:

MENU DE AJUDA	STATUS DEFAULT
1 - nível de acesso	Modo dirigido
2 - mensagem contextuais	Ativas
3 - tutor	(*) Desativado
4 - menu de teclas de função	Ativo
5 - árvore de funções	(*) Desativada

(*) Para desativar estes recursos acionar a tecla ESC. Os demais recursos só poderão ser desativadas no próprio menu de ajuda.

A seguir, descreveremos cada uma destas opções:

Nível de acesso:

Chaveia entre os modos de diálogo dirigido e direto;

Mensagens contextuais:

Abre uma janela com o texto de uma mensagem contextual, isto é, relacionada com a última ação sendo executada. Este texto, extraído do tutor, pode ainda estar relacionado a palavras chave da tela de interação (exibidas em destaque), para as quais o usuário pode apontar;

Tutor :

Texto contendo a descrição detalhada dos recursos do sistema e de como solicitá-los, incluindo exemplos e ilustrações;

Menu de telas de função:

Menu de barras, que ao ser ativado, exibirá uma relação de comandos (ver seção 4.2) válidos naquele instante da interação e as teclas de função a eles associadas;

Árvore de funções:

Apresentada na área de trabalho, exibe as funções e todas as suas subfunções em um diagrama em forma de árvore, destacando o caminho percorrido até o ponto presente da interação.

4.1.8 Métodos de detecção e recuperação de erro

Para efeito de tratamento pela interface, classificamos os erros em erros lógicos e erros de sintaxe.

4.1.8.1 Erros lógicos

Erros lógicos são ocasionados pela escolha, embora válida, de uma função ou dado indesejado. Estes podem ocorrer no nível de função ou no nível de subfunção.

No nível de função é possível descartar o que foi feito ATÉ ENTÃO através do comando ABORTA(F7).

No nível de subfunção desfaz-se o seu efeito com o comando DESFAZ(F6).

4.1.8.2 Erros de sintaxe

Sempre que o sistema detectar um erro de sintaxe na entrada do usuário, será emitida uma mensagem discriminando o erro e, simultaneamente, será dado destaque na tela à informação incorreta facilitando sua re-edição. O processo de re-edição permitirá o aproveitamento de toda a informação submetida anteriormente, sendo necessário corrigir apenas a informação incorreta.

4.1.9 Mensagens do Sistema

4.1.9.1 Mensagens de Navegação

Estas mensagens objetivam esclarecer o usuário ocasional ou inexperiente, sobre como progredir na interação com o sistema. Estão presentes no método de diálogo dirigido.

Por exemplo, ao selecionar a função CriaDir (criar diretório), surge o campo correspondente à "senha". O preenchimento deste campo é opcional, devendo ser preenchido a critério do usuário. Um exemplo da mensagem de navegação nesta situação seria:

"*Caso não deseje proteger o diretório com senha, tecle* **<ENTER>**.".

A retirada destas mensagens da tela é função da interface do sistema.

4.1.9.2 Mensagens de Erro

Estas mensagens são apresentadas independentemente do nível de interação ou de ajuda. São exibidas em uma janela na área de trabalho. Para removê-las da tela o usuário deve eliminar o erro que as originou. Pretende-se, através destas mensagens, não apenas informar a ocorrência e erro, mas também, onde ocorreu e o procedimento para corrigí-lo.

Para exemplificar, suponhamos que foi solicitada a função Simular Protótipo sem que antes tenha sido feita a geração do protótipo, surge então a seguinte mensagem:
"*Não existe arquivo objeto do protótipo solicitado. Execute a função Integrar Protótipo*".

4.1.9.3 Mensagens Contextuais

Estas mensagens estão disponíveis a partir da função AJUDA desde que solicitadas pelo projetista. Seu objetivo é esclarecer, através de um texto extraído do tutor, aspectos da interação relativos ao contexto onde o projetista se encontra. Este texto pode ainda estar relacionado com palavras chave da tela de interação (exibidas em destaque), para as quais o usuário pode apontar solicitando esclarecimentos.

O texto correspondente, aparece em uma janela na área de trabalho, e uma vez lido pode ser removido com o acionamento da tecla, esta tecla remove o texto em exibição mas não desativa as mensagens, para tanto, o usuário deve desativá-las no menu de AJUDA.

Por exemplo, para solicitar informações sobre "senha", se as mensagens contextuais estiverem ativadas no menu de ajuda, o usuário necessita apenas apontar para a palavra em destaque na tela e o texto correspondente será exibido, no estilo de hipertexto.

"A senha é um mecanismo de proteção..."

4.2 Descrição das funções

As funções do sistema podem ser classificadas em três grupos: funções de prototipagem, função de simulação e função de apoio. A maioria destas funções, devido à sua complexidade, se subdivide em subfunções.

Estas funções e subfunções, podem ser acessadas no modo direto, através da sintaxe de comandos e, no modo dirigido, através da árvores de menus.

Há ainda os Comandos Gerais. Estes comandos ativam as funções mais freqüentes usadas na interação do usuário com o sistema: AJUDA, MARCA, IMPRIME, RETORNA, INTERROMPE, DESFAZ, ABORTA, SALVA e FIMFUNC.

Estes comandos estão disponíveis em sua maioria ao longo de toda a interação. Para maior facilidade, eles podem ser ativados através das teclas de função no modo dirigido (ver Apêndice C), ou a partir do teclado onde deve ser digitado seu nome no modo Direto.

Apresentamos, a seguir, uma descrição das funções do sistema.

4.2.1 Manipular Diretórios/Arquivos

Funções de apoio que permitem que sejam feitas operações sobre diretórios e arquivos onde residem as informações sobre os protótipos. Os diretórios são organizados em árvore e, para cada protótipo, deve existir um diretório com todos os arquivos a ele relacionados. (ver seção 5.1)

Estas funções, embora presentes no sistema operacional, deverão ser acessadas através de uma interface que facilite a interação com o usuário principalmente no modo dirigido. Esta interface é baseada no princípio de que o usuário não deve ser forçado a digitar informações que já se encontram em exibição na tela, tais como o nome de diretórios e arquivos. Neste caso o usuário necessita apenas especificar qual dos arquivos e a função a ser executada, reduzindo assim a incidência de erro por digitação. Portanto, a interface consiste, no Modo Dirigido em:

1. Selecionar a função MANIPULAR DIRETÓRIO (MANIPULAR ARQUIVO);

2. A seguir é exibida, na área de trabalho, uma relação com os diretórios (arquivos existentes). Simultaneamente, é exibida, na área do menu de funções, a lista dos comandos de manipulação de diretórios ou arquivos.

3. O usuário deve então selecionar, no menu de funções, aquele comando correspondeste à operação a ser executada.

4. Finalmente, o usuário através do comando MARCA(F2) (ver Apêndice C), seleciona diretamente na lista de diretórios ou arquivos aqueles sobre os quais deve ser feita a operação selecionada.

No modo Direto ,no entanto, a interface é baseada na sintaxe de comandos, acrescentando-se apenas as facilidades de crítica destes comandos com mensagens de erro. A descrição destes comandos e de sua sintaxe é apresentada no Apêndice B.

Apresentamos, a seguir, a estrutura das duas funções mencionadas e detalhes sobre sua interface.

4.2.1.1 Função: Manipular Diretórios

Subfunções:

CriaDir

RemovDir

SelecDir

RenomDir

Criadir: permite a criação de um novo diretório. O usuário deve entrar as seguintes informações: drive, path, nome e senha(opcional) em campos específicos da tela (ver figura A.11).

RemovDir: remove diretório da árvore de diretórios. Só é permitida sua execução se não houver nenhum arquivo neste diretório e se a senha (caso exista) estiver correta.

SelecDir: seleciona diretório, permite acessar um diretório existente e seus arquivos desde que fornecida a senha (se existir) correta. Uma vez executada esta função, exibe na tela, a lista dos arquivos existentes no diretório escolhido.

RenomDir: permite mudar o nome de um diretório existente, desde que seja fornecida a senha correta.

4.2.1.2 Função: Manipular Arquivos

Subfunções:

CriaArq

SelecArq

RemovArq

ImprmArq

CopiArq

RenomArq

CriArq: cria um novo arquivo dentro do diretório atual, ou especificado. O nome do arquivo aparecerá com a terminação . DEC, na lista de arquivos apresentada na área de

trabalho. Ver Apêndice D, sobre os tipos de arquivos do sistema e as operações que podem ser executadas sobre cada um deles;

SelecArq: seleciona um arquivo dentro do diretório atual, ou especificado. Se o arquivo for do tipo .DEC poderá ser prototipado, se for do tipo . EXE poderá ser simulado e, finalmente, do tipo .DOC poderá ser editado ou impresso;

RemovArq: permite a remoção de um ou mais arquivos do diretório atual ou em outro especificado. É possível com o comando MARCA(F2) selecionar os arquivos a serem removidos diretamente na tela;

ImprmArq: imprime um ou mais arquivos que tenham sido secionados;

CopiArq: possibilita a cópia de um arquivo do diretório atual ou mesmo selecionado, para outro arquivo, no mesmo diretório ou em outro especificado;

RenomArq: permite mudar o nome de um arquivo no diretório atual ou em outro especificado.

4.2.2 Compor Diálogo

Função de prototipagem que oferece recursos para especificação de diálogos e a sua conexão com as telas de apresentação e os periféricos de E/S.

Inicialmente, estão disponíveis para prototipagem quatro estilos de diálogo: Comandos, Formulário, Menu, Pergunta e Resposta. Além destes diálogos, é possível ainda editar textos como tutoriais e mensagens.

A escolha de um, dentre quatro estilos de diálogos, não é uma limitação, pois, é possível, também, combinar estilos de diálogo (ex. comandos com parâmetros retirados a partir de menus), isto se dá graças à estrutura de especificação destes diálogos. Nesta estrutura, se especifica o estilo da resposta do usuário (ex. resposta textual pelo teclado, seleção de uma opção em menu, etc.).

Esta mesma estrutura permite relacionar o diálogo com os dispositivos utilizados para entrada da informação (ex. seleção de uma opção em menu através de um mouse). É esta estrutura que passamos a descrever.

Função: Compor Diálogo

Subfunções:

Editar

Copiar

Remover

Detalhar:

- aspectos específicos de cada estilo de diálogo,
- comandos aplicáveis,
- recuperação de erros,
- ações, etc.

Inicialmente, ao ativar a função, o projetista deve especificar o estilo de diálogo a ser modelado, identificá-lo em termos de seu projeto, e posicioná-lo na tela associando-o a uma área de tela onde será apresentado. Caso esta área ainda não tenha sido definida, o sistema registrará a pendência indicando-a para o usuário, e este poderá ou não, interromper a função Compor Diálogo e chamar a função Projetar Tela.

Estas informações comuns aos quatro estilos de diálogo, completam o processo de Edição, que consiste em definir o texto do diálogo. É possível, também, remover do protótipo um diálogo já especificado ao ainda, copiá-lo, para um outro diálogo, bastando para isso sua identificação.

Devido às diferentes características de cada um destes estilos, reservou-se a subfunção Detalhar onde, através de formulários (em ambos os estilos de interação), o usuário complementa as informações necessárias à definição.

Dentre os aspectos a serem detalhados, três são comuns a todos os estilos de diálogo comandos aplicáveis e recuperação de erros e ações, os quais descrevemos abaixo:

Comandos aplicáveis

Quando da execução de uma função da interface prototipada, de acordo com o contexto da interação existirá um conjunto de entradas válidas do usuário. A estas entradas denominamos comandos aplicáveis.

Para especificar o contexto da interação, dividimo-la em um conjunto de etapas que varia de acordo com o estilo de diálogo. A cada etapa é associado um sub-conjunto válido

dos comandos do protótipo, cada um deles acarretando uma ação específica dentro do contexto da interação.

Por exemplo: no sistema AGILE o comando ABORTA, durante a execução de uma função, só é válido antes da execução do comando SALVA.

Recuperação de erros

Na recuperação de erros, o projetista indica que ações a interface deve executar nestas situações. Por exemplo, ao submeter um comando inválido, este deve ser rejeitado ou reeditado? Se reeditado, deve ser a partir do erro ou integramente?

Ao submeter uma entrada ao sistema, o usuário pode incorrer em dois tipos de erro. O primeiro consiste em uma entrada válida, porém indesejada; o segundo, por outro lado, consta de uma entrada inválida. Neste caso, a interface pode ter, em sua especificação, métodos de recuperação de erros.

Ações

As ações são os comportamentos definidos para os objetos da interface, de acordo com os eventos gerados pela interação do usuário com interface e desta com o programa aplicativo. A relação das ações com os diálogos, telas e periférico modelam a interface de uma aplicação.

A hierarquia da conversação é especificada através do encadeamento de ações durante a especificação dos diálogos.

Ao interromper o protótipo da interface, o usuário o faz através do estilo de diálogo especificado (dentre aqueles disponíveis no sistema ou de sua combinação), para cada um destes estilos, durante a especificação, associa-se uma ou mais ações à cada entrada do usuário, como nos exemplos a seguir:

Comandos: ao submeter um comando;

Formulário: com o preenchimento de um campo ou formulário como um todo;

Menu: ao escolher uma opção;

Pergunta e Resposta: ao responder uma pergunta;

Texto: ao se exibir um texto aplicativo;

Mensagem: ao ser exibida uma mensagem.

Durante a especificação de um diálogo, o encadeamento de ações pode ser definido ao se relacionar um evento a uma seqüência de ações. Este sequenciamento é gerado pelo Gerenciador de Diálogos.

A partir da análise de interfaces, inclusive a do protótipo do sistema AGILE, extraíram-se os seguintes tipos de ações que combinados são capazes de simular a maioria das funções de interface das aplicações analisadas:

- Ativar tela;
- Acionar periférico;
- Mudar atributo de periférico;
- Ativar mensagens;
- Ecoar string na tela;
- Encerrar execução de uma função;
- Time-out;
- Encerrar execução do protótipo;
- Nenhuma ação.

Ativar tela: esta ação deve ser associada a eventos que resultem em uma mudança na apresentação da informação no vídeo, com ela deve ser informada a identificação da tela mostrada.

Acionar periférico: esta ação deve ser associada a eventos que necessitam da ativação de um dispositivo de entrada/saída, o qual deve ser escolhido dentre aqueles disponíveis à prototipagem.

Mudar atributo de periférico: esta ação deve ser associada a eventos que resultem em uma mudança na forma como um dispositivo periférico trata a informação que lhe é enviada. Por exemplo a mudança de atributos de vídeo, atributos de impressão, etc.. o dispositivo deve ser identificado e o atributo especificado dentre aqueles que lhe são disponíveis.

Ativar mensagens: esta ação deve ser associada a um evento que requeira a apresentação de uma mensagem para o usuário. A ela deve ser associada a identificação da mensagem a ser exibida.

Ecoar string na tela: esta ação deve ser associada a eventos tais como a escolha de uma opção de menu, que deve ser escrita em uma posição da tela. A ela deve ser associada a identificação do string e a posição de eco na tela.

Encerrar execução de uma função: esta ação deve ser associada a eventos que resultem na terminação da execução de uma função da interface.

Time-out: esta ação permite a especificação de um tempo de espera, após o qual uma nova renovação deverá ser disparada.

Encerrar execução do protótipo: esta ação deve ser associada a eventos que resultem na terminação da execução da interface, devolvendo o controle ao ambiente de prototipagem do sistema AGILE.

Nenhuma ação: esta ação deve ser associada a eventos que resultem em ações não especificadas. Útil também durante a prototipagem em situações tais como a chamada a uma função da aplicação que não está presente.

Finalmente, os dados específicos para estilo de diálogo, tais como: definição de campos de formulários, parâmetros de comandos e opções de menus, serão discutidos a seguir. Apresentaremos, para cada um destes estilos, a relação das informações necessárias para complementar suas especificações. Para algumas informações, é mostrado um menu de opções, a partir do qual é feita a seleção do que se deseja.

Para efeito de esclarecimento, fazemos as seguintes observações:

1. Ao mostrarmos as definições e informações de esclarecimentos dos diálogos, colocaremos em negrito o que fizer parte do sistema e normal o que for entrada do projetista;
2. O projetista deve analisar quais as etapas pelas quais passa a interface e, a partir daí, definir os comandos aplicáveis e as ações correspondentes;
3. Definir como será efetuada a recuperação de erros (nas diversas circunstâncias em que possam ocorrer);
4. Identificar a/as ação/ações disparada(s) em função de uma tomada de decisão por parte do usuário.

A seguir são apresentados os procedimentos de definição e esclarecimentos para os tipos de diálogos tratados por AGILE, para os quais devem ser consideradas as observações colocadas anteriormente.

4.2.2.1 Comandos

Possibilita a edição de diálogos baseados na sintaxe de comandos (ver Fig. A.7)

Durante a edição do comando, uma vez tendo identificado e alocado uma área, o projetista pode, então, definir o comando preenchendo as seguintes informações em um formulário:

- Identificação do comando;
- Sintaxe do comando;
- Nome(s) do comando;
- Teclas de entrada (teclas de função, soft keys, etc.);
- Delimitadores dos parâmetros;
- Separadores dos parâmetros;
- Terminador do comando.

O próximo passo consiste em detalhar o comando. Isto é feito através da função Detalhar (parâmetros, comandos aplicáveis, recuperação de erros e ações).

Em seguida, mostraremos as informações necessárias para descrevermos um parâmetro de um comando:

- Identificação do comando;
- Parâmetro;
- Tipo;
- Faixa de validade.
- Obrigatoriedade ("sim" ou "não");
- Valor default;

Ações em caso de erro:

- Cancela o parâmetro;
- Cancela a partir do erro;
- Cancela o comando.

Método de entrada:

- Menu;
- Pergunta e Resposta;

- Formulário;
- String.

Para preenchimento de ações em caso de erro, o projetista pode selecionar uma ou mais opções aquelas mostradas no menu, conforme descrito.

Para método de entrada, o projetista deve selecionar uma das opções mostradas no menu, conforme descrito acima.

Para detalhamento do comando, no que diz respeito a comandos aplicáveis, recuperação de erros e ações, mostramos abaixo (com exemplos de preenchimento), as informações necessárias:

Identificação do comando;

ETAPA	COMANDOS APLICÁVEIS	AÇÕES
Antes de submeter o comando	ABORTA, HELP	Mostrar tela M2
Depois de submeter o campo	ABORTA, EXIT	Mostrar tela M5

Ação para recuperação de erros:

- Reedita o erro;
- Cancela a partir do erro;
- Cancela o comando.

Ação para execução de comandos:

- Encerra execução de uma função;
- Dispara nova tela;
- Encerra execução do protótipo;
- Aciona periférico;
- Dispara mensagem;
- Muda atributo de periférico;
- Nenhuma ação.

4.2.2.2 Formulário

Possibilita a especificação de diálogos no estilo preenchimento de formulário (Ver Fig. A.10).

Uma vez tendo identificado e alocado uma área, o usuário pode, então, editar o formulário. Para isto, ele dispõe dos recursos de um editor de textos e de um editor gráfico (ver seção 5.3).

Tendo editado o formulário, o próximo passo consiste em detalhá-lo para o sistema.

O detalhamento das informações sobre os campos do formulário em definição é feito selecionando-se, em primeiro lugar o campo que deve ser detalhado. A seguir, deve ser preenchido um formulário da prototipagem com as informações referentes ao campo escolhido.

As informações para definição de um diálogo no estilo formulário são:

- Identificação do formulário;
- Identificação do campo;
- Seqüência de ações;
- Tipo de campo (alfabético, numérico, etc.);
- Decimais;
- Tamanho do campo (em caracteres);
- Posição de início de campo (linha, coluna);
- Estilo de preenchimento;
- Método de entrada.

O detalhamento das informações, sobre os campos do formulário em definição, é feito selecionando-se, em primeiro lugar, o campo em que deve ser detalhado. A seguir, deve ser preenchido um formulário com as informações referentes ao campo escolhido. No modo dirigido, a identificação do campo a ser detalhado, pode ser obtida a partir do próprio formulário onde o usuário MARCA o campo desejado com a tecla F2 que, também, pode ser usada para marcar a posição onde deve iniciar o preenchimento dos campos e a seqüência de preenchimento dos mesmos.

O método de entrada, informa quando interpretar que o conteúdo de um campo já foi preenchido. Por exemplo: pressionando-se uma tecla, a qual deve ser indicada, ao preencher todo o espaço disponível para o campo, etc.

As informações para detalhamento do diálogo estilo formulário são:

Identificação do formulário;

Identificação do campo;

ETAPA	COMANDOS APLICÁVEIS	AÇÕES
Antes de submeter o campo	QUIT, UNDO, HELP	A1
Depois de submeter o campo	ABORT, EXIT	A5

Ações em caso de erro:

- Reeditar o(s) caracter(es);
- Cancelar o campo;
- Exibir mensagem.

Ações para formulário completo:

Ações relacionadas a cada campo;

4.2.2.3 Menu

Possibilita a especificação de diálogos no estilo seleção em Menu (ver Fig. A.8).

Uma vez tendo identificado e alocado uma área, o usuário pode então editar o menu. Para edição do menu o usuário dispõe dos recursos de um editor de textos e editor gráfico, descrições na seção 5.3.

Tendo editado o menu, o próximo passo consiste em detalhá-lo para o sistema.

Informações para definição de um diálogo estilo Menu:

- Identificação do menu;
- Método de seleção;
- Dispositivo apontador;(menu: mouse, cursor,..)
- Método de marcação;
- Posição de eco da opção selecionada.

Método de seleção é a forma como um item do menu poderá ser selecionado (ex. apontando, através de um número, digitando-se a opção, digitando-se a(s) inicial(ais) da opção,...)

Método de deslocamento sobre as opções é a forma de percorrer as opções do menu, isto é, a seqüência em que os itens devem ser apontados (ex. descendente, ascendente, direita-esquerda,...).

O método da marcação indica como confirmar a seleção de uma ou mais opções do menu (ex. recla ESC).

Posição de eco é o local na tela do protótipo onde o ítem selecionado deve ser ecoado (escrito). Pode ser indicada através da tecla MARCA.

As informações detalhadas para definição de um diálogo estilo Menu são:

Identificação do Menu

ETAPA	COMANDOS APLICÁVEIS	AÇÕES
Antes de confirmar seleção das opções	C2,C3	A2
Depois de confirmar seleção das opções	C2,C5	A3

Recuperação de erros

OPÇÃO INVÁLIDA	AÇÃO A SER TOMADA
O1	A3
O4	A2

Ações a partir de uma dada opção

OPÇÃO	AÇÃO
O1	A2
O2	A4
O3	A3

4.2.2.4 Pergunta e reposta

Possibilita a especificação de diálogos no estilo pergunta e resposta (ver Fig A.9).

Uma vez tendo identificado e alocado uma área, o usuário pode então editar a pergunta e respectivas respostas.

Tendo editado a pergunta, o próximo passo consiste em detalhar suas respostas para o sistema.

A definição e detalhamento das respostas à pergunta são feitos a partir das seguintes informações:

- Identificação da pergunta
- Pergunta (texto)
- Relação de respostas (texto)

ETAPAS	COMANDOS APLICÁVEIS	AÇÕES
Antes de entrar a resposta	C7, C3	A1
Depois de entrar a resposta	C3, C5	A3

RESPOSTA ERRADA	AÇÃO DE RECUPERAÇÃO DE ERROS
E1	M2
E3	M5

RESPOSTA	AÇÃO/AÇÕES A EXECUTAR
R3	A4
R5	A7

Exemplos para mensagens de recuperação de erros seriam: "Cancela a resposta", "Cancela a partir do erro" e "Repete a pergunta".

4.2.2.5 Editar Texto

Possibilita a edição de textos longos, tais como tutoriais, manuais, etc,...

Uma vez tendo identificado e alocado uma área, o usuário pode então editar o texto. Tendo editado o texto, o próximo passo consiste em detalhá-lo para o sistema.

O detalhamento dos comandos aplicáveis durante a exibição de um texto é apresentado a seguir.

Identificação do texto;

ETAPA	RELAÇÃO DE COMANDOS	AÇÕES
Durante a exibição do texto	C2, C5, C8	A2, A5

4.2.2.6 Editar Mensagem

Permite que sejam criadas mensagens para serem apresentadas ao longo da interação, as quais serão integradas a um banco de mensagens. Este banco armazena todas as mensagens já especificadas no protótipo e apresenta suas identificações ao projetista na forma de um menu, de modo que podem vir a ser reutilizadas.

Uma vez tendo identificado a mensagem e especificado a área da tela onde será exibida, o usuário pode então editar o texto da mensagem. Tendo editado o texto, o próximo passo consiste em detalhar a mensagem para o sistema.

O detalhamento dos comandos aplicáveis e das ações durante a exibição de uma mensagem é mostrado abaixo.

Identificação da mensagem

ETAPAS	COMANDOS APLICÁVEIS	AÇÕES
Durante a exibição da mensagem	C3, C5, C6	A2, A4

MENSAGEM	AÇÕES
M3	A2, A9

4.2.3 Projetar Tela

Função de prototipagem que oferece recursos para formalizar a apresentação da informação, tais como: dimensionamento da tela, divisão em áreas com especificação de seus respectivos atributos de localização, dimensão, cores, classificação em texto e/ou gráfica, etc... (ver Fig. A.13).

Para maior flexibilidade na prototipagem, é possivel apresentar a informação na tela através de janelas ou em mosaico. É permitida a designação de áreas superpostas (janelas), tendo-se que observar apenas como as mesmas se encadearão ao longo da interação.

A seguir, apresentamos a estrutura desta função

Função: Projetar Tela

Subfunções:

CriArea

RemovArea

LocArea

CopiArea

DetalhArea

CursorArea

DistCor

MovArea

Inicialmente, ao ativar a função, o projetista deve identificar a tela e dimensioná-la em número de linhas e colunas.

O dimensionamento da tela permite que o projetista defina as dimensões da tela onde será apresentada a interface do protótipo. O default do sistema são as dimensões padrão (24 linhas X 80 colunas) que é o padrão utilizado na tela do sistema AGILE. No entanto, caso as dimensões especificadas excedam as da tela do sistema, esta tela será utilizada como uma janela que pode ser deslocada sobre a tela do protótipo. O deslocamento sobre a tela especificada é tarefa do editor gráfico.

Para maior comodidade do projetista durante o projeto de telas estará disponível na mesma (a pedido do usuário) uma régua e as informações sobre a posição do cursor (linha e coluna).

Uma vez tendo definido as dimensões da tela, o projetista pode então definir o seu layout e outros aspectos descritos a seguir.

CriArea: cria uma área na tela do protótipo a partir da especificação de suas dimensões ou da manipulação de um recurso denominado "caixa-elástica". Este recurso exibe uma caixa de dimensões ajustáveis, a partir da utilização de um dispositivo apontador. Este dispositivo deve apontar para os pontos extremos da diagonal da caixa e reposicioná-la na tela de modo a atingir as dimensões desejadas.

Uma vez tendo definido as dimensões, o usuário deve especificar se a área será utilizada para exibição de texto ou gráfico.

RemovArea: possibilita a remoção de uma área da tela, previamente especificada. Esta remoção só será possível se não houver definições relacionadas com esta área (por exemplo a alocação de um diálogo a esta área).

LocArea: posiciona uma área especifica na tela, a partir da especificação da coordenadas superiores de sua diagonal, ou com um dispositivo apontador – o mouse ou o cursor. A superposição de área é permitida, deixando-se para o módulo Gerenciador de Telas o sequenciamento da apresentação da informação. Este sequenciamento é conseqüência das especificações do protótipo.

CopiArea: copia a especificação de uma área previamente definida para a área em especificação ou outra indicada.

DetalhArea: possibilita o detalhamento de uma área especificada. Este detalhamento cobre três aspectos: a movimentação do cursor, a utilização de cores e a movimentação da informação dentro da área.

CursorArea: possibilita a entrada de informações sobre o cursor de uma área especificada. Inicialmente, o cursor deve ser identificado e, a seguir, especifica-se o mecanismo de posicionamento do cursor (teclas, dispositivo apontador, etc.).

DistCor: possibilita ao usuário especificar as cores da área em definição ou de outra especificada. O usuário selecionará as cores a partir de um menu.

- Cor de frente (dois caracteres)

- Cor de fundo (da tela)
- Cor de contorno (das bordas da área)

MovArea: possibilita ao usuário definir a forma de movimentação da informação dentro da área em definição ou de uma outra especificada. Para tanto, o usuário deve especificar:

- O dispositivo usado para movimentar a área (tecla, dispositivo apontador, etc.);
- O tipo de apresentação (paginação ou rolamento da tela);
- O tipo de movimento – se o tipo de apresentação é paginação (avançar, retornar); se o tipo selecionado é rolamento da tela (direita, esquerda, para cima, para baixo).

4.2.4 Definir Relatório

Função de prototipagem que oferece recursos para especificação de formato e conteúdo de relatórios que fazem parte da interface em definição (ver Fig. A.14).

Função: Definir Relatório

Subfunções:

Editar

Imprimir

Remover

Copiar

Uma vez tendo identificado o relatório, o usuário pode então especificá-lo a partir das seguintes subfunções:

Editar: coloca à disposição do usuário os recursos de edição para a elaboração de relatórios: um editor de textos e um editor gráfico, ressaltando-se aqueles recursos relativos ao destaque da informação impressa.

Imprimir: imprime o relatório especificado.

Remover: remove do protótipo a definição do relatório especificado.

Copiar: possibilita ao usuário a cópia das definições de um relatório especificado para o relatório em definição ou para outro especificado.

4.2.5 Integrar Protótipo

Função de prototipagem que consiste em gerar o código do protótipo para o módulo Simulador (ver Fig. A.16).

A integração se dá em duas etapas. Na primeira, é feita a validação das especificações após testes de consistência, quando são detectados entre outros: objetos referenciados e não definidos, objetos definidos e não referenciados, sequenciamento ilegal da interação, etc. Após esta fase de validação, caso tenham sido detectados erros, o sistema emite uma relação destes erros, a partir da qual o usuário se referencia à especificação para efetivar as correções.

Se os testes de consistência forem satisfeitos, o sistema gera um código fonte em C, a partir do arquivo de definições. Este código pode ser compilado fora do sistema, caso se deseje introduzir alguma modificação (patching).

A próxima etapa de integração consiste na "linkagem" do protótipo com os módulos do sistema, necessários durante a simulação do protótipo (ex. Drivers de periféricos, gerenciador da interface, etc.) produzindo o executável do protótipo.

4.2.6 Simular Protótipo

Função de Simulação: consiste em executar o código do protótipo especificado (ver Fig. A.6).

Durante sua execução, um sub-conjunto dos comandos do sistema AGILE, relativo à solicitação de ajuda e retorno ao sistema operacional, está disponível. Porém, os recursos de prototipagem, necessários à modificação das especificações, só poderão ser acessados fora deste ambiente, isto é, no ambiente de prototipagem.

4.2.7 Documentar Protótipo

Função de apoio que facilita a transferência dos resultados da prototipagem para o ambiente de desenvolvimento da aplicação (ver Fig. A.15).

Embora os dados relativos às decisões do protótipo sejam atualizados à medida que estas decisões são incorporadas à especificação, ao solicitar esta função o sistema mais uma vez atualiza a apresentação destas informações e exibe para o usuário um documento capaz de descrever detalhadamente a especificação, contendo a descrição dos objetos especificados e suas relações:

- Layout de telas
- Diálogo
- Relação de dispositivos periféricos
- Método de ajuda
- Método de detecção e recuperação de erro
- Relação das funções do protótipo, etc.

Este documento, a partir de entará, está disponível ao usuário para impressão e edição permitindo-lhe acrescentar comentários que aumentem a clareza da especificação.

Função: Documentar Protótipo

Subfunções:

Editar

Imprimir

Uma vez tendo identificado o protótipo, o arquivo correspondente à sua documentação estará disponível ao usuário que poderá editá-lo ou imprimí-lo.

4.2.8 Copiar Pacotes

Função de apoio que facilita a transferência dos resultados da prototipagem para o ambiente de desenvolvimento da aplicação (ver Fig. A.17). Permite que sejam copiados os módulos de software de interesse para o projetista, a partir de uma relação que o sistema lhe oferece.

Esta opção permite que o projetista utilize recursos disponíveis no sistema, tais como: drivers de periféricos, processador de linguagem natural, etc., com o objetivo de integrá-los ao ambiente da aplicação.

No capítulo seguinte, será especificado o software a ser implementado para atender a esta descrição funcional

ESPECIFICAÇÃO DO SOFTWARE V

Para descrever nesta etapa os blocos básicos constituintes do sistema proposto, utilizou-se a técnica HIPO – Hierarquia mais Entrada/ Processo/ Saída [DAVIS 87].

Esta técnica consiste em representar a estrutura de um programa de forma descendente através de um "Quadro Hierárquico" representando os módulos do programa, e de um conjunto de "Quadros IPO" que descrevem as entradas, as saídas e o processo realizado por cada módulo representado no quadro hierárquico.

No quadro hierárquico, os módulos de alto nível são genéricos, realizando funções de controle, enquanto os níveis inferiores correspondem a funções detalhadas. Portanto, para um maior nível de detalhamento, os módulos devem ser decompostos até que representem uma única função lógica completa.

A técnica HIPO foi escolhida por satisfazer as necessidades de documentação da fase de especificação do projeto, permitir a avaliação e refinamento do projeto e corrigir falhas antes da implementação além de permitir uma abordagem modular na elaboração desta documentação.

Inicialmente, elaboramos apenas os quadros hierárquicos e uma descrição textual de seus módulos, enquanto que uma decomposição mais refinada dos módulos e os quadros IPO correspondentes foram deixados para o projetista do sistema. Esta divisão em etapas foi possível graças à modularidade desta técnica.

5.1 Diagrama Hierárquico

Na Fig. 5.1 apresentamos o "*Quadro Hierárquico de Alto Nível*" do Sistema, e nas Fig. 5.2 a 5.11, a decomposição de cada um dos módulos representados no Quadro Hierárquico de Alto Nível.

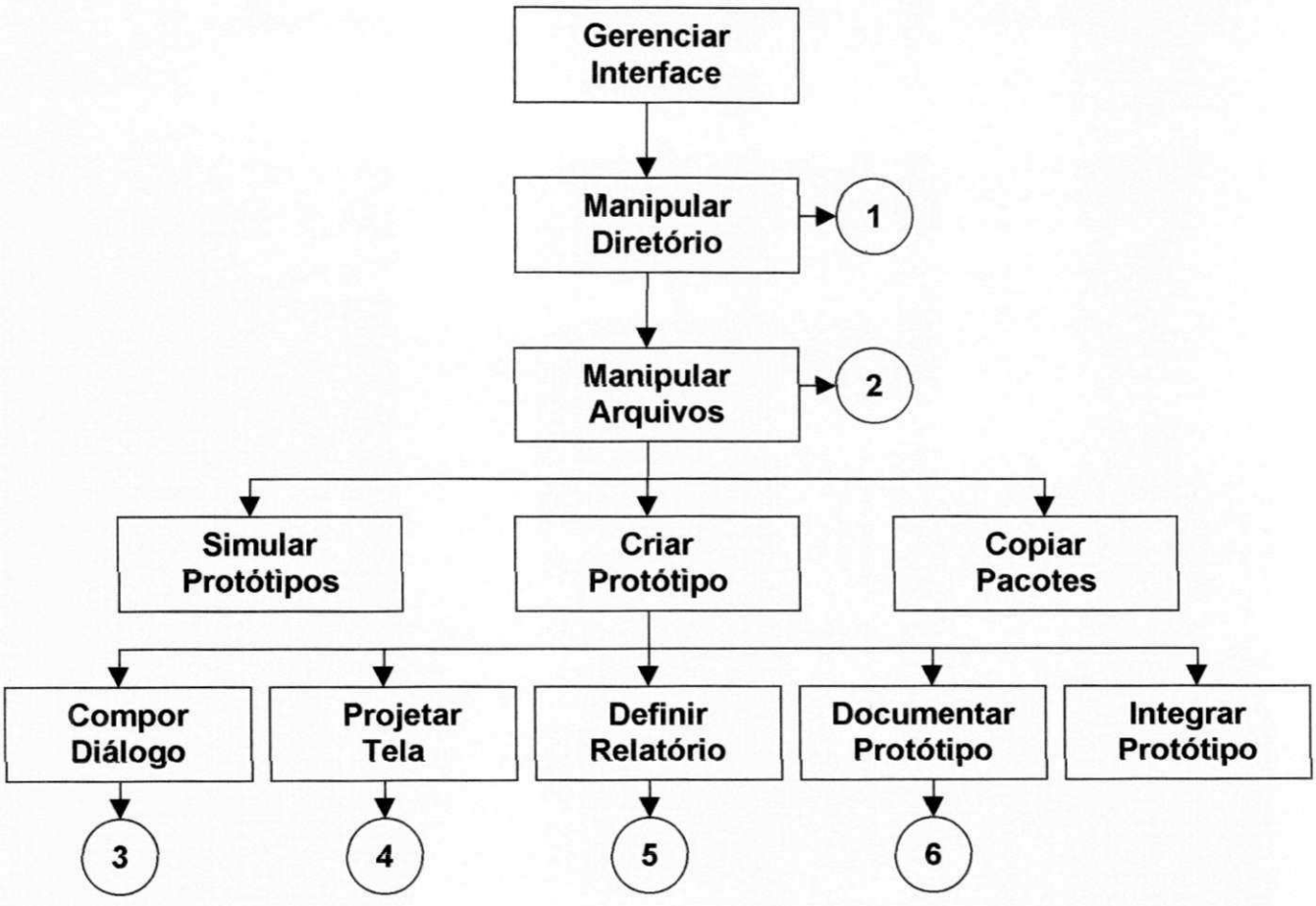

Fig. 5.1 – Quadro hierárquico de alto nível do sistema **AGILE**

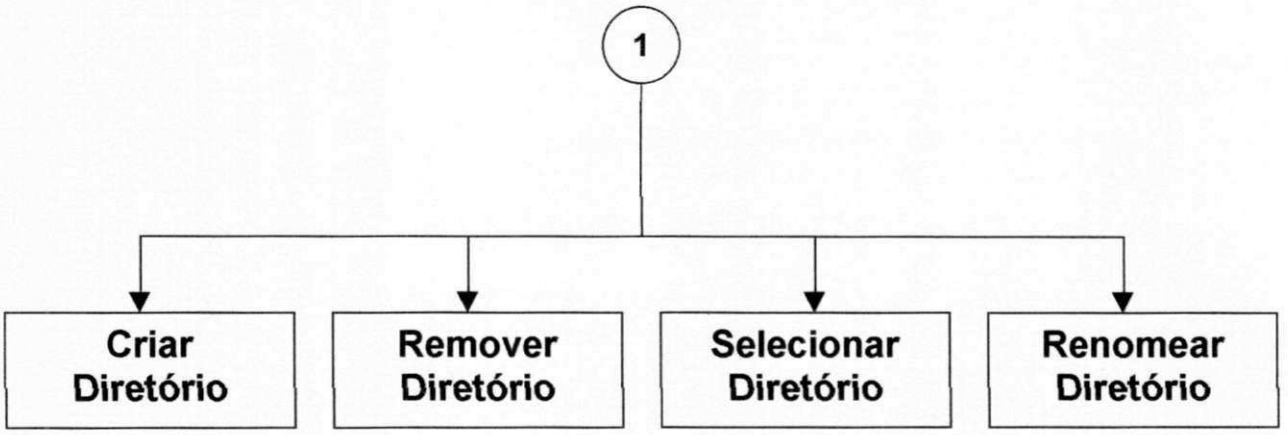

Fig. 5.2 – Decomposição funcional do módulo Manipular Diretório

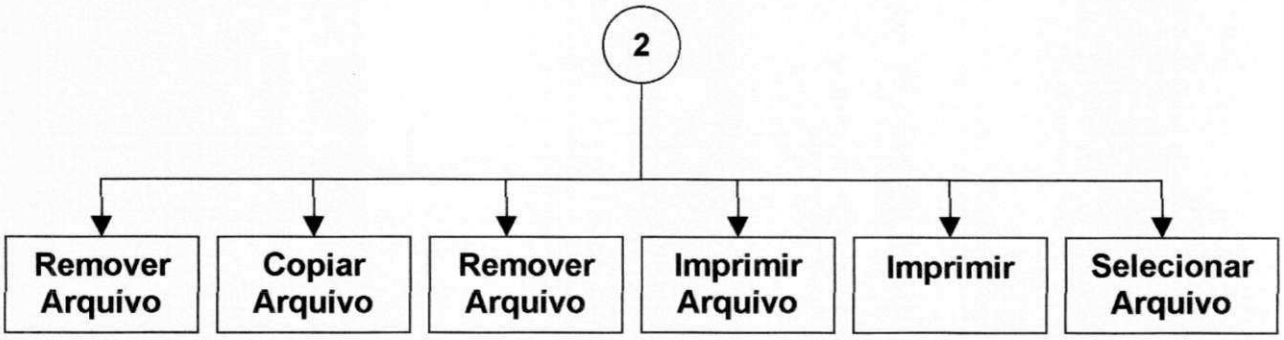

Fig. 5.3 – Decomposição funcional do módulo Manipular Arquivos

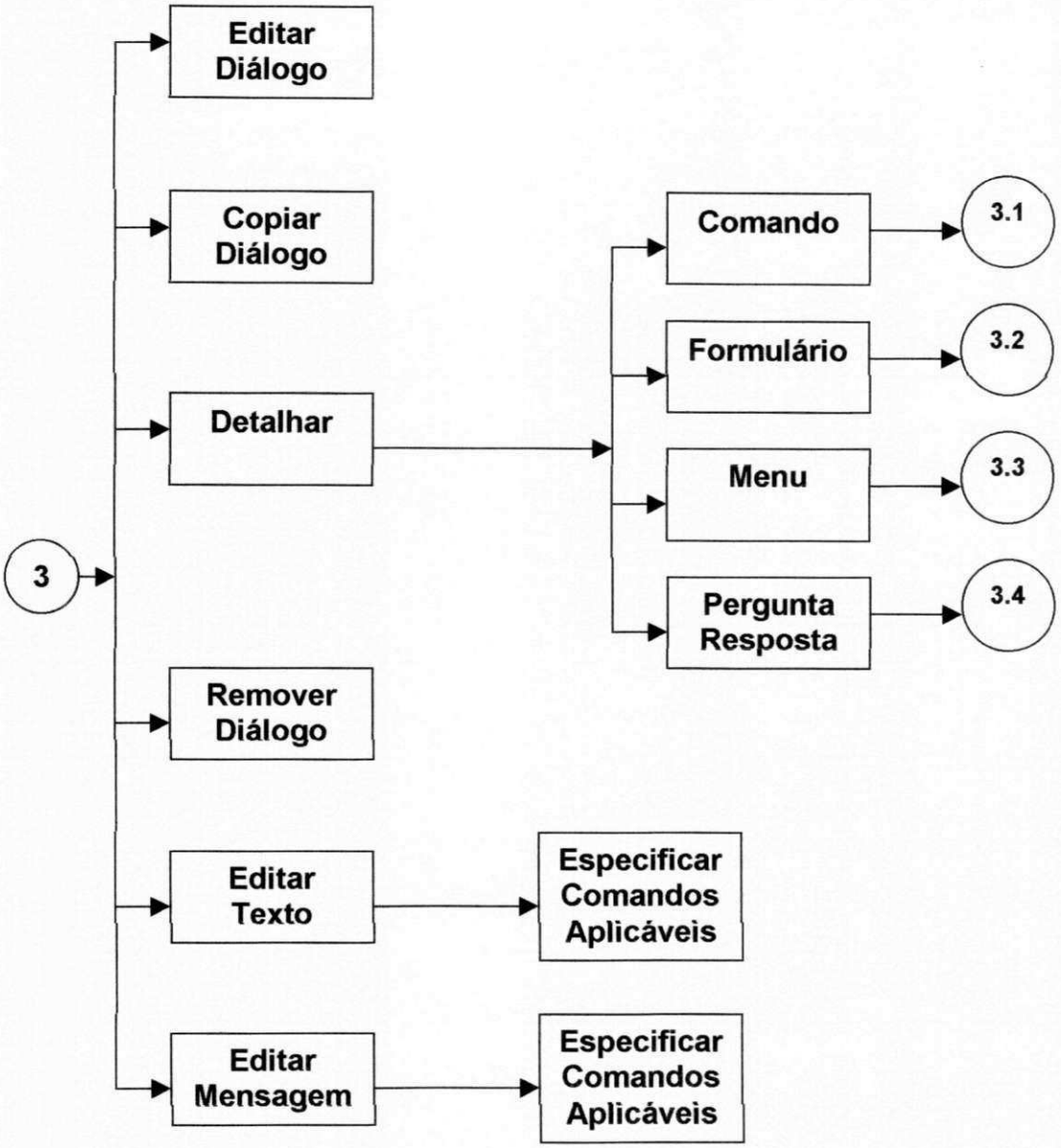

Fig. 5.4 – Decomposição funcional do módulo Compor Diálogo

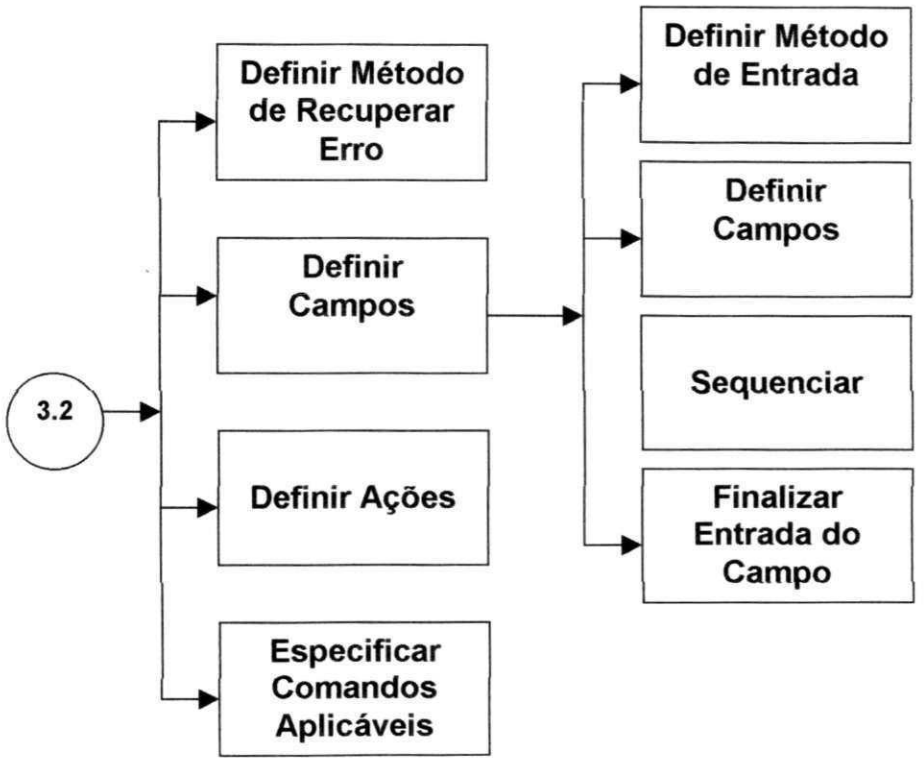

Fig. 5.6 – Decomposição funcional do sub- módulo Detalhar Formulário

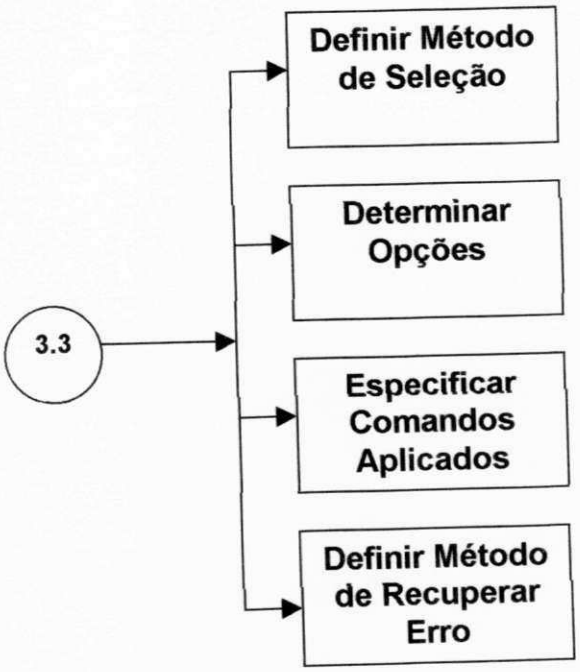

Fig. 5.7 – Decomposição funcional do sub-módulo Detalhar Menu

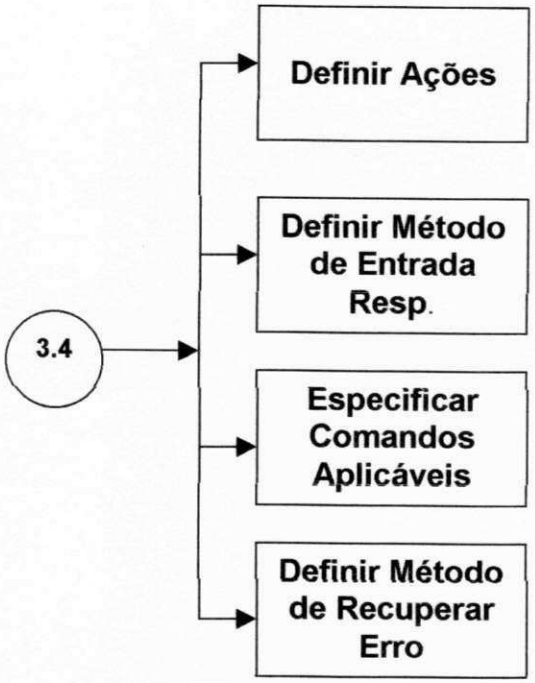

Fig. 5.8 – Decomposição funcional do sub-módulo Detalhar Pergunta e Resposta

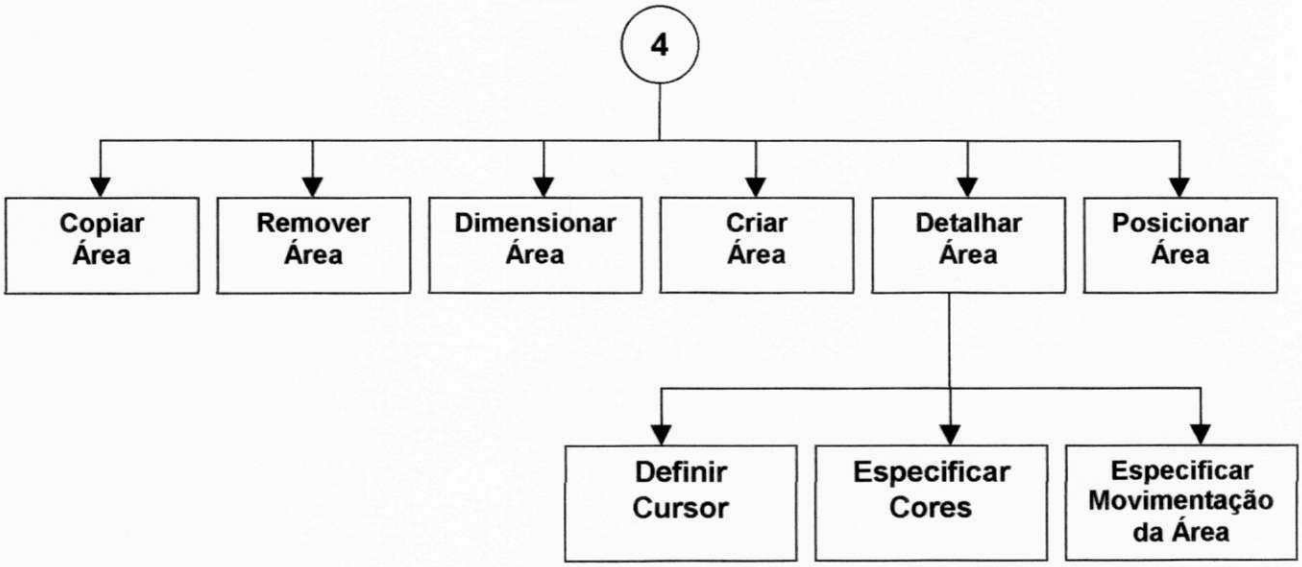

Fig. 5.9 – Decomposição funcional do módulo Projetar Tela

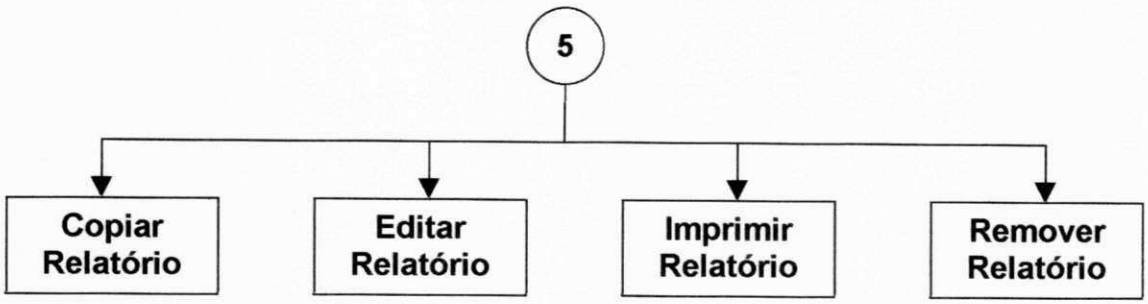

Fig. 5.10 – Decomposição funcional do módulo **Definir Relatório**

5.2 Descrição dos módulos

A descrição funcional destes módulos, à exceção do bloco "*Gerenciador Interface*", já foi apresentada ao longo da seção 4.2. Uma descrição mais detalhada em diagramas IPO, já está em andamento como primeira fase de uma dissertação de mestrado responsável pela implementação desta especificação.

O bloco Gerenciador da Interface tem como função, durante a prototipagem, coordenar a interação entre os vários blocos do sistema, e destes com as ferramentas disponíveis, a partir dos eventos gerados pelo projetista, e pelo sistema AGILE.

Por outro lado, durante a simulação , a tarefa de coordenação deste módulo é a mesma, apenas as origens dos eventos e as ações que decorrem deles são diferentes, uma vez que a interface agora é entre o modelo de uma aplicação e o usuário final desta.

Este bloco coordena três outros módulos:

- Gerenciador de Diálogo;
- Gerenciador de Periféricos;
- Gerenciador de Telas.

O módulo **Gerenciador de Diálogos** coordena o sequenciamento da interação, a partir das especificações dos diálogos e dos eventos gerados tanto pelo usuário quanto pelo modelo da aplicação.

O módulo **Gerenciador de Periféricos** coordena o acesso aos periféricos especificados no protótipo, a partir das solicitações do Gerenciador e Diálogos e Gerenciador de Telas.

O módulo **Gerenciador de Telas** coordena a apresentação da informação no vídeo, a partir das especificações do protótipo, e da interação com os dois módulos descritos acima.

5.3 – Especificação das ferramentas do sistema

O Editor de textos – Este recurso deverá oferecer ao projetista as facilidades a seguir:

- Manipulação de textos
- Destaque da informação na tela e na impressão (WYSWIG)
- Impressão de relatório

  Hard-copy da tela

  Documentação da interface

WYSWIG – "what you see is what you get" – significando que os recursos de destaque da informação, tais como negrito, sublinhar, etc. serão representados fielmente na tela durante a edição.

O **Editor-Gráfico** – Este editor permitirá a criação e manipulação de objetos gráficos que integrarão as telas dos protótipos; interfaceando-os com os dispositivos periférico de entrada/saída disponíveis no sistema.

A seguir, relacionamos as classes de objetos que devem ser tratadas por este editor:

- Texto
- Menu
- Calendário
- Relógio análogico e digital
- Icone
- Sprite
- Desenho Livre (Shape, Box, ...)
- Bargauge
- Barra de rolagem
- Mostrador analógico
- Mostrador digital

A especificação de cada um destes objetos, assim como das funções que podem ser aplicadas sobre cada um deles, é objeto de uma dissertação de mestrado em andamento.

O **Gerenciador de Banco de Dados** – O sistema necessitará de uma estrutura de Banco de Dados própria, com recursos para definição, manipulação e gerenciamento de dados.

Isto se justifica graças à diversidade dos dados armazenados, seu volume e, acima de tudo, devido à necessidade de relacioná-los para fins de estudos posteriores como a criação de uma base de conhecimento na área de interfaces.

Devemos armazenar os dados do sistema e um conjunto de dados para cada protótipo definido.

A estrutura dos dados e a abordagem adotada na implementação do banco de dados, são objeto de uma dissertação de mestrado em desenvolvimento.

5.4 restrições na Implementação

Considerando que a especificação aqui proposta se refere a uma versão inicial do sistema, visando sua expansão em termos de funções e até mesmo de ambiente de hardware, colocaram-se algumas restrições na implementação, as quais estão descritas a seguir.

5.4.1 O Sistema Operacional

A escolha de PCs deixa, poucas alternativas para escolha do sistema operacional. Sugerimos DOS, apesar do Windows oferecer vários recursos que facilitam o desenvolvimento do nosso sistema. No entanto, estes recursos criaram uma dependência em relação ao Windows que dificultaria sua migração para outros ambientes.

5.4.2 Linguagem de Implementação

Esta linguagem deve satisfazer os critérios a seguir, de modo a garantir a flexibilidade na expansão do sistema, a importação de módulos de software e o transporte do protótipo da interface para o ambiente da implementação.

- Portabilidade
- Fácil integração com outras linguagens
- Documentação disponível
- Compatibilidade entre versões
- Recursos para desenvolvimento e testes
- Experiência dos programadores

Apesar de se sugerir uma linguagem que satisfaça as restrições acima descritas, as decisões sobre a mesma e o sistema operacional são preteridas, de modo a permitir uma maior flexibilidade ao projetista, o qual deverá levar em consideração outros fatores, conhecidos apenas na fase de implementação.

ESPECIFICAÇÃO DO HARDWARE VI

6.1 O ambiente AGILE

Levando-se em conta a popularidade, custo e disponibilidade dos computadores da linha IBM-PC, este foi o ambiente escolhido para hardware do sistema AGILE.

Com esta escolha, asseguramos a compatibilidade entre a especificação dos protótipos gerados no sistema e o ambiente alvo para o qual a grande maioria deles será desenvolvida. Contamos ainda no mercado, com uma gama de periféricos compatíveis e softwares desenvolvidos e em desenvolvimento, os quais podem vir a ser integrados ao sistema.

Para ter uma margem de flexibilidade, durante o processo de especificação e simulação do protótipo, sugere-se o PC/AT-386, dada sua capacidade de memória e velocidade de processamento.

A configuração recomendável é a descrita abaixo:

- PC/AT-386;
- 120 MB de disco rígido;
- Unidade(s) de disco(s) flexível(eis): 5 ¼", 3 ½", etc.;
- Impressora;
- Placa e monitor VGA;
- Mouse;

6.2 O Ambiente de prototipagem

Para que se possa prototipar interfaces para diferentes aplicações, os recursos do sistema devem ser bastante variados. No que diz respeito ao hardware, isto significa mais opções em termos dos dispositivos usados na interação direta (dispositivos apontadores,

selecionadores, etc.) e indireta (monitores de vídeo, impressoras, etc.) com a interface. Dentre os dispositivos disponíveis para prototipagem, pretendemos oferecer, inicialmente:

- Mouse;
- Caneta ótica;
- Tela sensível ao toque;
- Tablete digitador;
- Joystick;
- Tracker ball.

Supondo que a maioria dos sistemas prototipados executarão em um PC, é interessante que o usuário perceba os resultados de sua prototipagem em um ambiente capaz de reproduzir os mesmos efeitos. Por exemplo, se a resolução de sua placa de vídeo é inferior àquela oferecida pelo sistema AGILE, as decisões em termos da apresentação da informação no protótipo, podem ser inadequadas. Sendo assim. Propõe-se que o usuário deva especifique o ambiente alvo de seu protótipo, configurando o sistema AGILE antes de iniciar sua prototipagem.

Por outro lado, ele deverá dispor da opção utilizar a configuração default do sistema, com a qual pode explorar outras alternativas de interface junto ao usuário final.

No capítulo seguinte, passaremos a apresentar as conclusões e propostas de trabalhos futuros.

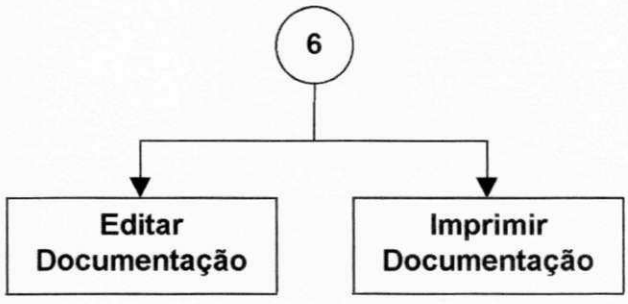

Fig. 5.11 – Decomposição funcional do módulo **Documentar Protótipo**

CONCLUSÕES VII

Até recentemente, o usuário de computadores foi tratado apenas como um parâmetro do sistema com o qual interagia. Subestimado em sua complexidade e ignorado durante o desenvolvimento do sistema, era apenas ouvido quando pouco poderia ser modificado para tornar seu trabalho mais agradável e produtivo.

Capítulos e até livros têm sido escritos sobre como modificar este quadro, porém, devido ao caráter multidisciplinar das considerações exigidas para esta mudança e os custos decorrentes das tentativas frustradas para atingir os objetivos por ela traçados, a exploração de alternativas de interfaces, foi adiada até o surgimento dos sistemas de prototipagem rápida.

Como podemos constatar, através das referências no capítulo 2, a área de interfaces vem se consolidando no que diz respeito a recursos para prototipagem rápida e gerenciamento de interfaces. Observa-se, no entanto que a maioria destes sistemas é dedicado à classes específicas de interfaces, ao contrário do que nos propomos a fazer.

A diversidade de aplicações para as quais o sistema AGILE se propõe a prototipar interfaces, será possível graças à diversidade de seus recursos: uma gama maior de estilos de diálogo, dispositivos de interação e métodos de apresentação da informação na tela (janelas, mosaico, etc.) que, combinados, podem dar origem a protótipos totalmente distintos para uma mesma aplicação e para aplicações totalmente distintas.

Por outro lado, sua linguagem de especificação interativa e com base fortemente declarativa, é apoiada por uma interface que pode ser configurada em vários níveis de ajuda, de acordo com as características do usuário. Desta forma, tenta-se superar o problema enfrentado pela maioria dos usuários de sistemas de prototipagem que é a dificuldade na interação com o sistema e a necessidade de dominar uma linguagem formal de especificação, isto é, fortemente procedural.

Finalmente, a modularidade de seu projeto, deverá garantir a flexibilidade necessária à importação de resultados obtidos na área, tanto do ponto de vista de hardware, quanto de software. Esta modularidade possibilitará, ainda, a exportação de recursos do sistema para o ambiente da aplicação sendo prototipada.

Um dos primeiros passos a serem dados na expansão do sistema, deverá ser para torná-lo um SGIU, capaz de gerar o código da interface especificada, o qual será integrado à aplicação.

Nas próximas etapas do projeto sugere-se lançar as bases de um sistema inteligente, fundamentado sobre a experiência acumulada ao longo dos vários projetos desenvolvidos. Para tanto, é necessário que se construa uma base de conhecimento a partir dos resultados obtidos durante a validação destes projetos.

Finalmente, a interface do sistema AGILE, ainda pode evoluir bastante, principalmente a linguagem de especificação, talvez utilizando os recursos do próprio sistema que, neste caso, estaria se auto-redefinindo.

Com este trabalho, espera-se estar iniciando a construção de uma base de experimentação e estudos na área de interfaces, onde o projetista é encorajado no desenvolvimento interativo envolvendo o usuário final. Neste processo, delineia-se uma metodologia de projetos onde sugere-se uma seqüência de passos e aspectos que devem ser cuidados na especificação de uma interface.

Espera-se, ainda, modificar a abordagem da inovação versus padronização na indústria dos computadores [POLTROCK 89], no caso do nosso mercado nacional. Na proposta destas interfaces, garantimos a presença no projeto de elementos característicos dos nossos usuários e sistemas.

BIBLIOGRAFIA

[CARD 83], Card, Stuart K., Moran, Thomas P., Newell, Allen, "*The Psychology of Human_Computer Interaction*", Lawrence Erlbaum Associates Publisheres, 1983.

[DAVIS 87] Davis W.S., "*Análise e Projeto de Sistemas – Uma Abordagem Estruturada*'', Ed. Livros Técnicos e Científicos S.A.,1987.

[GALITZ 81] Galitz, Wilbert O., "*Handbook of Screen Format Design*", North-Holland, 2nd Ed.,1981.

[GREEN 85] Green M., "*The University of Alberta User Interface Management System Development and Application*", ACM SIGGRAPH 85, Vol. 19, No. 3, July 85, pp. 205-213.

[GUTIERREZ 89] Gutierrez O., "*Prototyping Techniques for Different Problem Contexts*". Proc. CHI 89 Conference on Human Factors in Computing Systens, May 1989, pp. 259-264.

[HUCKLE 81] Huckle B.A., "The Man-Machine Interface: Guidelines for the Design of End-User/System Conversation", Savant Research Studies, Savant Institute, Jan 1981.

[JAMSA 88] Jamsa K., "*Windows – Guia do Usuário*" – McGraw-Hill Ltda, 1988.

[LEWIS 89] Lewis T.G., Handloser III F., Bose S., and Yang S., "*Prototipes from Standard User Interface Management System*", IEEE Computer, Vol. 32, No. 5, May 1989, pp. 51-56.

[LOWGREN 88] Lowgren J., "*History, State and Future of User Interface Management System*" ACM SIGGHI Bulletin, Vol.20, no.1, July 1988, pp. 32-44.

[LUQI 89] Luqi. "*Software Evolution Through Rapid Prototyping*", IEEE Computer, Vol. 22, No.5, May 1989, pp. 13-25.

[MANHEIMER 89] Manheimer J. M., Burnett R. C., and Wallers J.A., "*A Case Study of User Interface Management System Development and Application*". Proc. CHI 89 Conference on Human Factors in Computing Systems, May 1989, pp. 127-132.

[MARTIN 73] Martin J., "*Design of Man-Computer Dialogues*", Prentice Hall, Englewood Cliffs, N. J., 1973.

[MELO 85] Melo R. N., Silva S. D., "*Uma Ferramenta para Especificação de Sistemas Interativos*", Anais SCESU 85, S. P., pp. 658-663.

[OLSEN 89] Olsen Jr D. R., "*A Programming Language Basis for User Interfaces Management*". Proc. CHI 89 Conference on Human Factors in Computing Systems, May 1989, pp. 171-176.

[PERRY 89] Perry T. S., "*Of Mice and Menus: Designing the User-Friendly Interface*", IEEE Spectrum, Vol. 26 No. 9, September 1989, pp. 46-51.

[POLTROCK 89] Poltrock S. E., "*Innovation in User Interface Development: Obstacles and Opportunities*". Proc. CHI 89 Conference on Human Factors in Computing Systems, May 1989, pp. 191-195.

[WELLNER 89] Wellner P. D., "*Statemaster: A UUIMS based on Starcharts for Prototyping and Target Implementation*". Proc. CHI 89 Conference on Human Factors in Computing Systems, May 1989, pp.177-182.

[WIECHA 89] Wiecha C., Bennett W., Bores S., e Gould J., "*Generating Highly Interactive User Interfaces*", Proc, CHI 89 Conference on Human Factors in Computing Systems, May 1989, pp. 277-282.

APÊNDICE A – LAYOUT DE TELAS

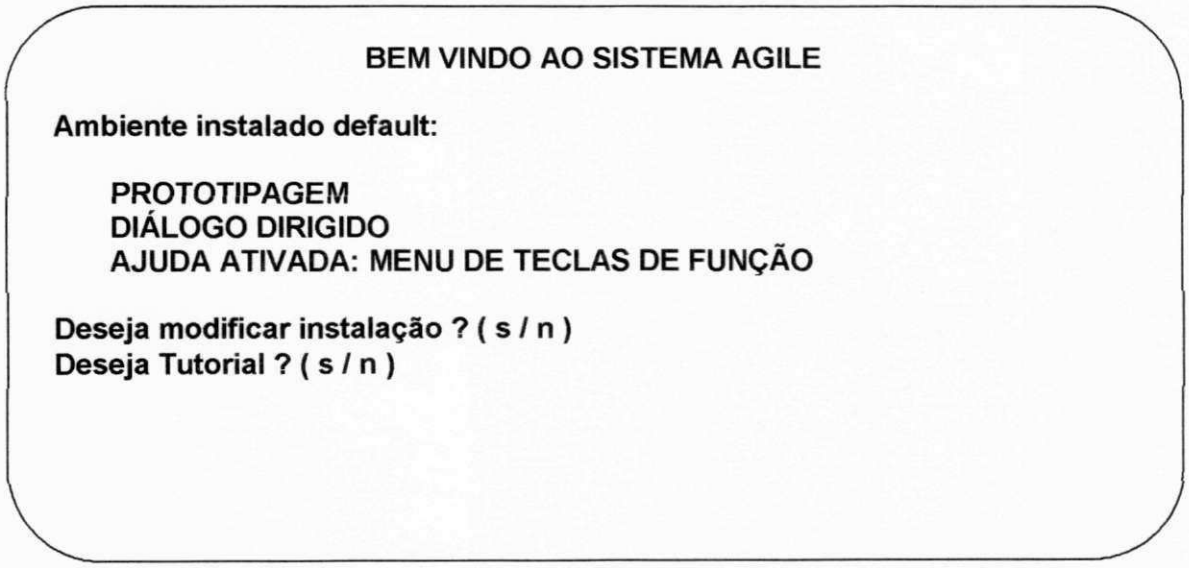

Fig. A.1 – Tela de Abertura

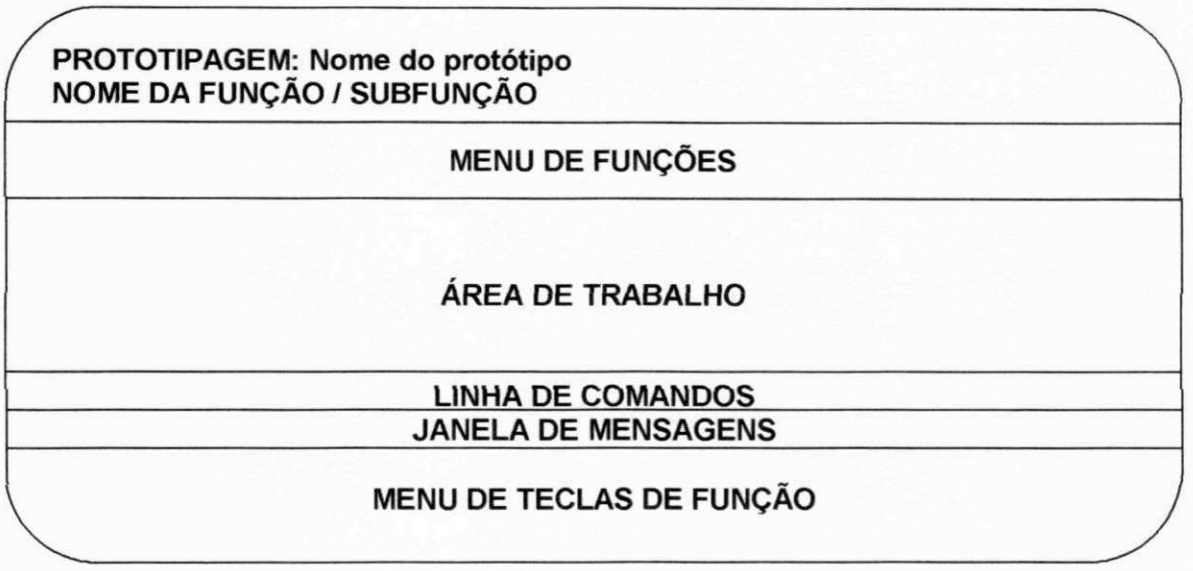

Fig. A.2 – Layout da tela no Ambiente de Prototipagem.

Diálogo Dirigido, com ajuda

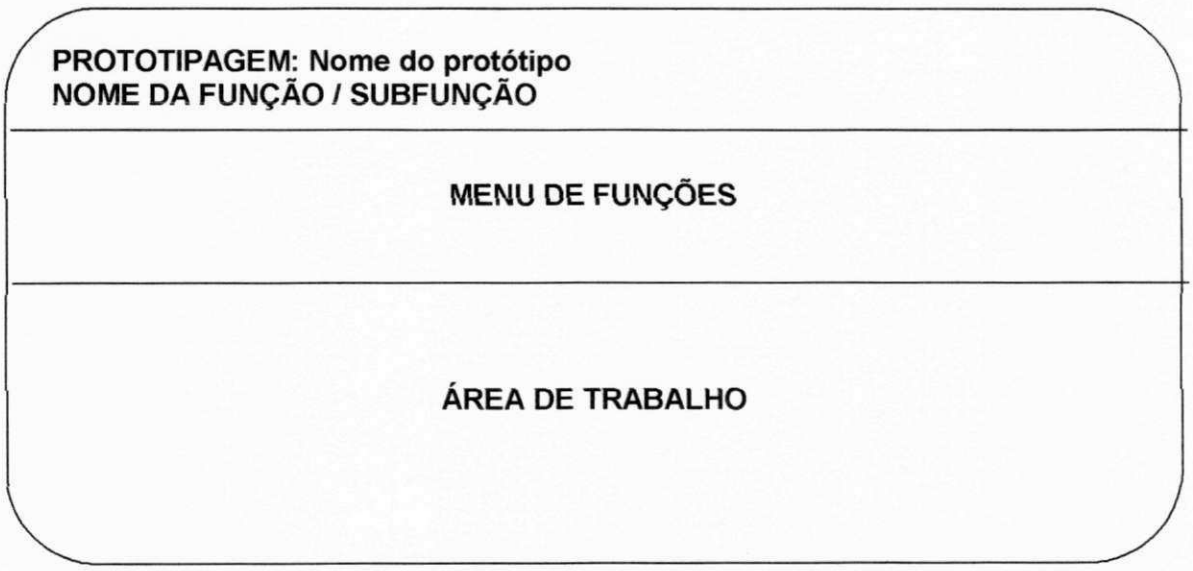

Fig. A.3 – Layout da tela no Ambiente de Prototipagem.

Diálogo Dirigido, sem ajuda

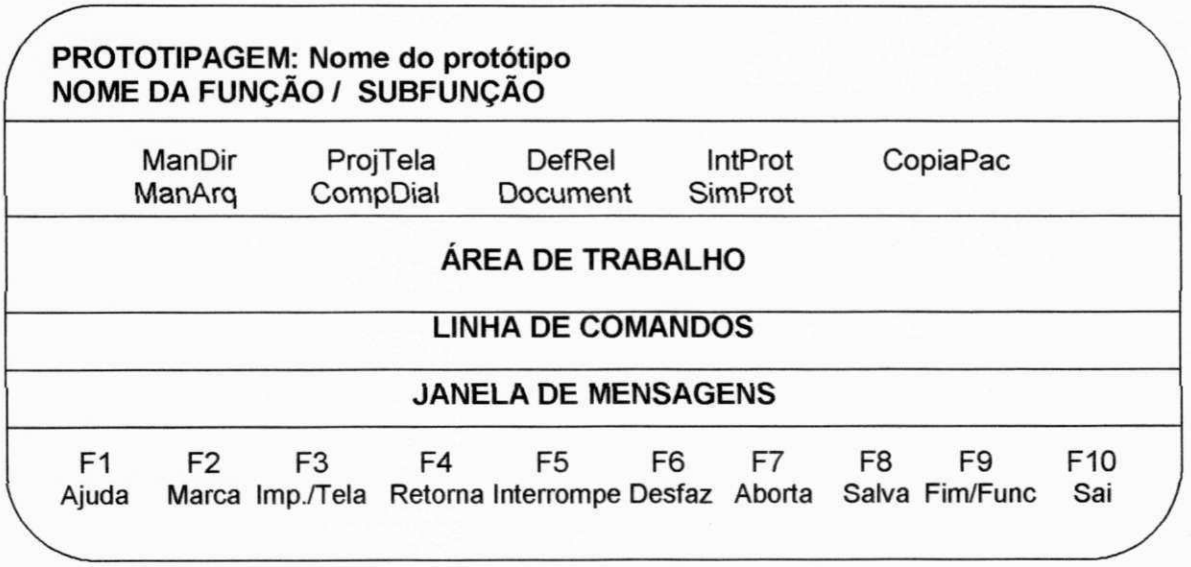

Fig. A.4 – Layout da tela no Ambiente de prototipagem.

Diálogo Direto, com ajuda

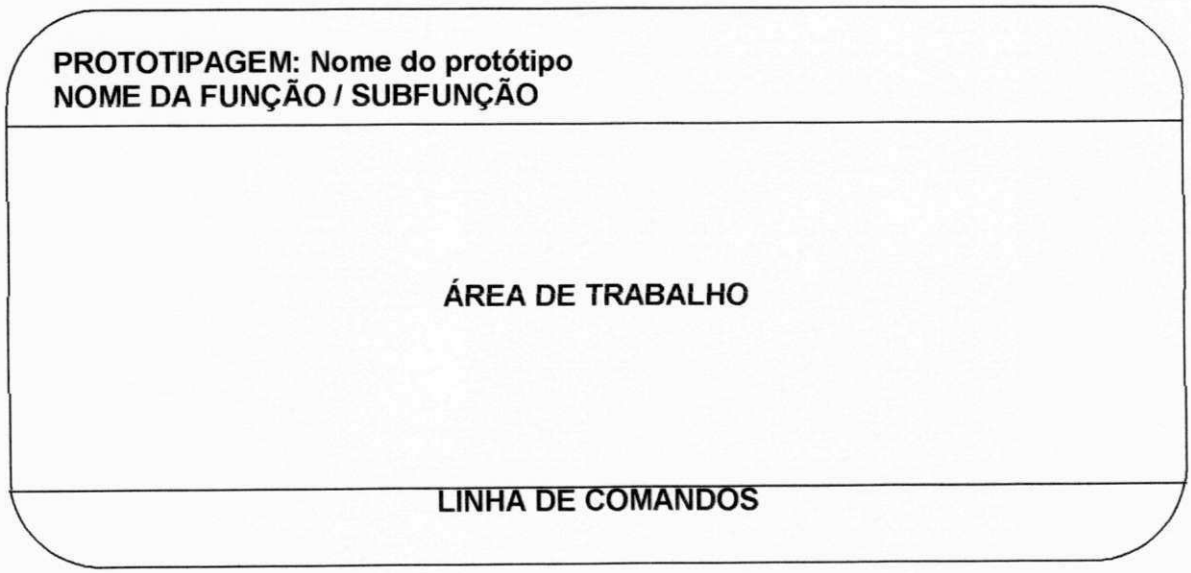

Fig. A.5 – Layout da tela no Ambiente de prototipagem.
Diálogo Direto, sem ajuda

Fig. A.6 – Layout da tela no Ambiente de Simulação, com ajuda.

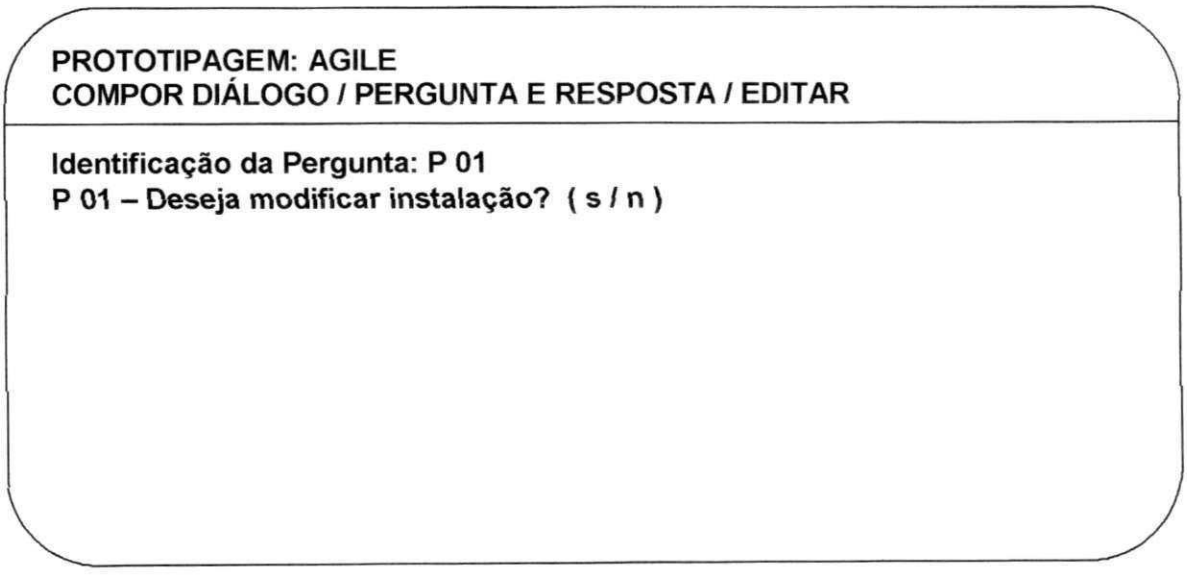

Fig. A.9 – Exemplo de tela da função **COMPOR DIÁLOGO** do tipo **PERGUNTA E RESPOSTA.**

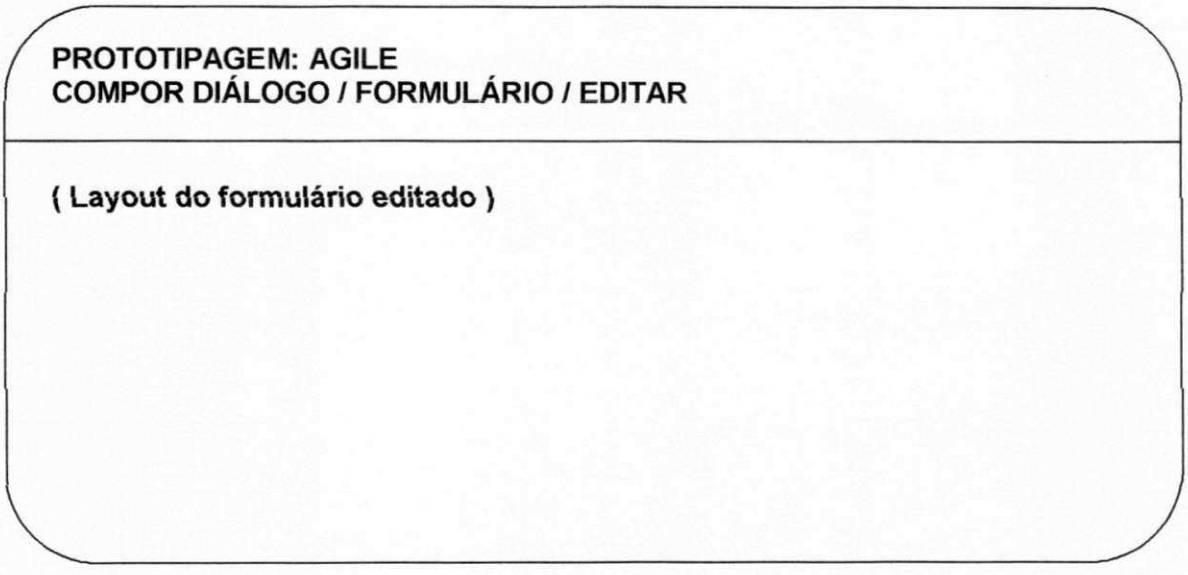

Fig. A.10 – Exemplo de tela da função **COMPOR DIÁLOGO** do tipo **FORMULÁRIO**.

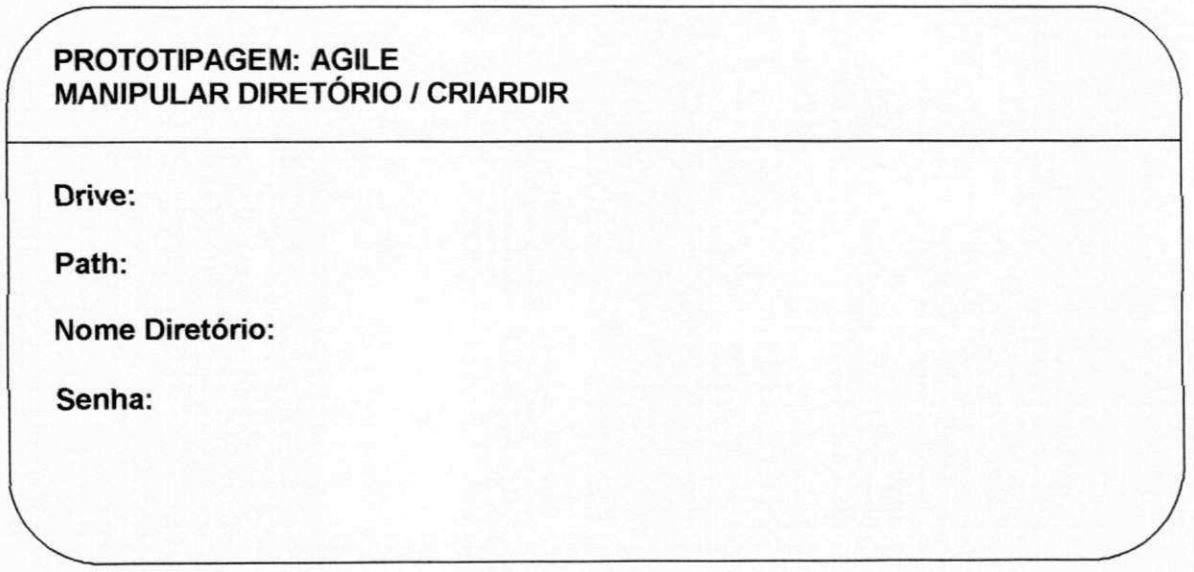

Fig. A.11 – Exemplo de tela da função **MANIPULAR DIRETÓRIO.**

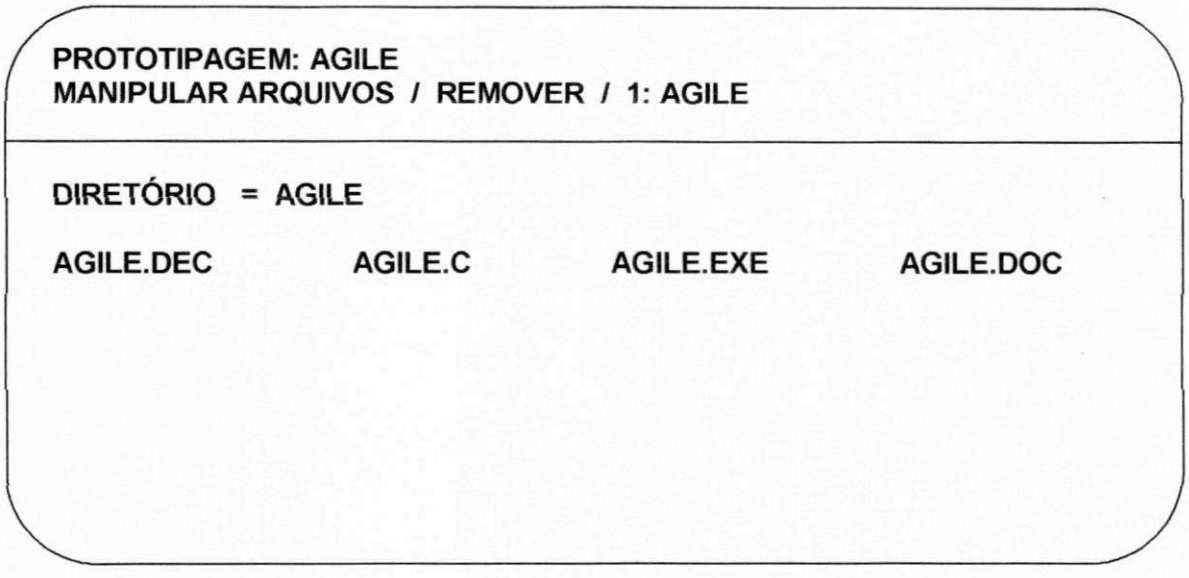

Fig. A.12 –Exemplo da tela da função **MANIPULAR ARQUIVOS.**

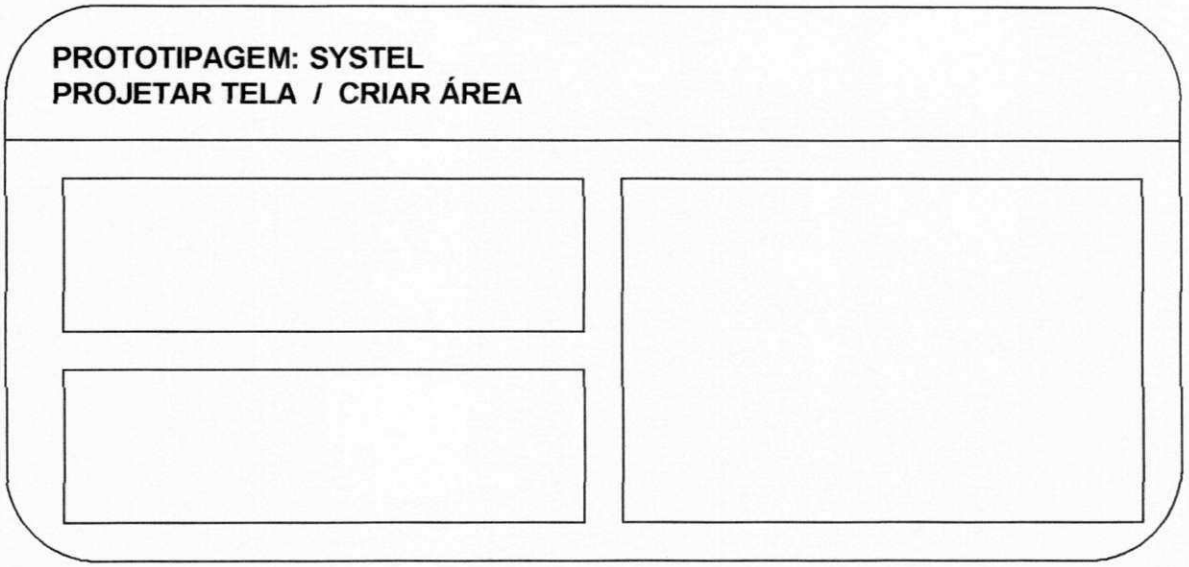

Fig. A.13 – Exemplo de tela da função **PROJETAR TELA.**

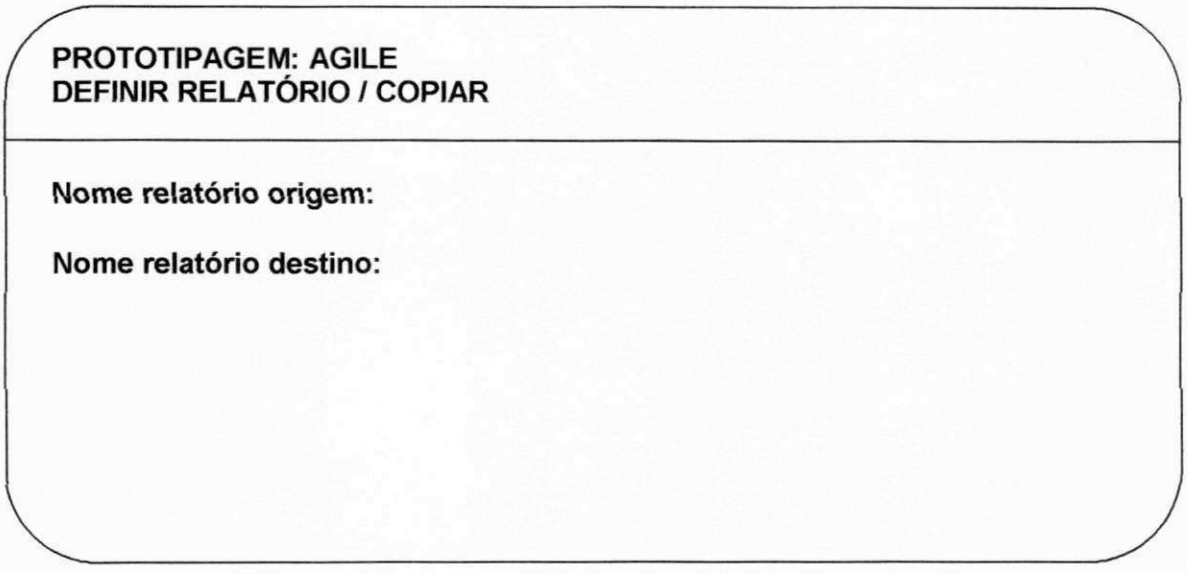

Fig. A.14 – Exemplo de tela da função **DEFINIR RELATÓRIO.**

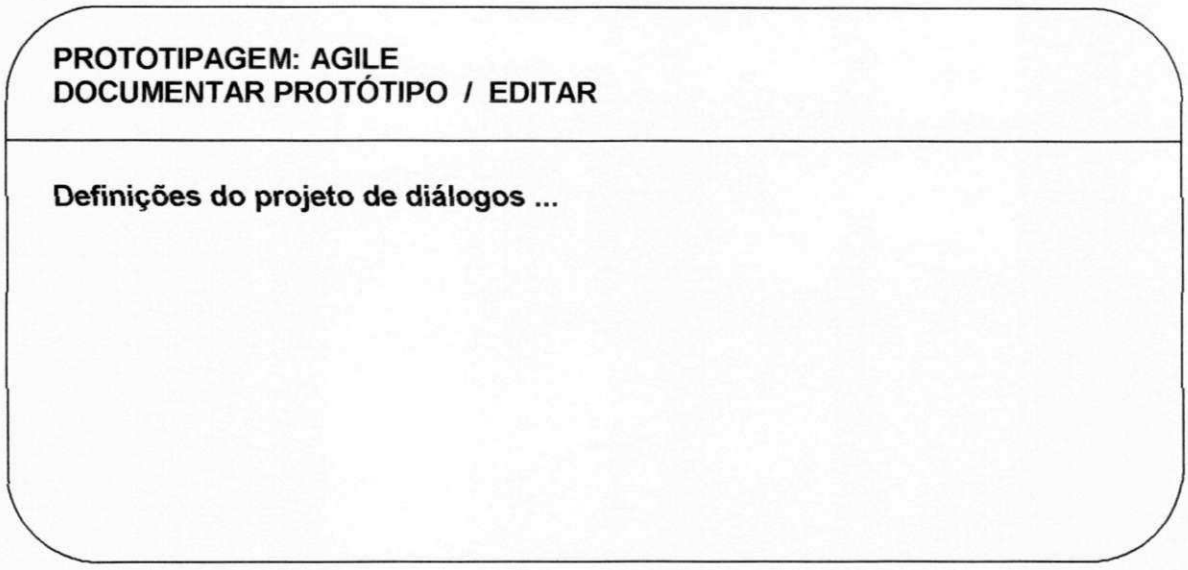

Fig. A.15 – Exemplo de tela da função **DOCUMENTAR PROTÓTIPO.**

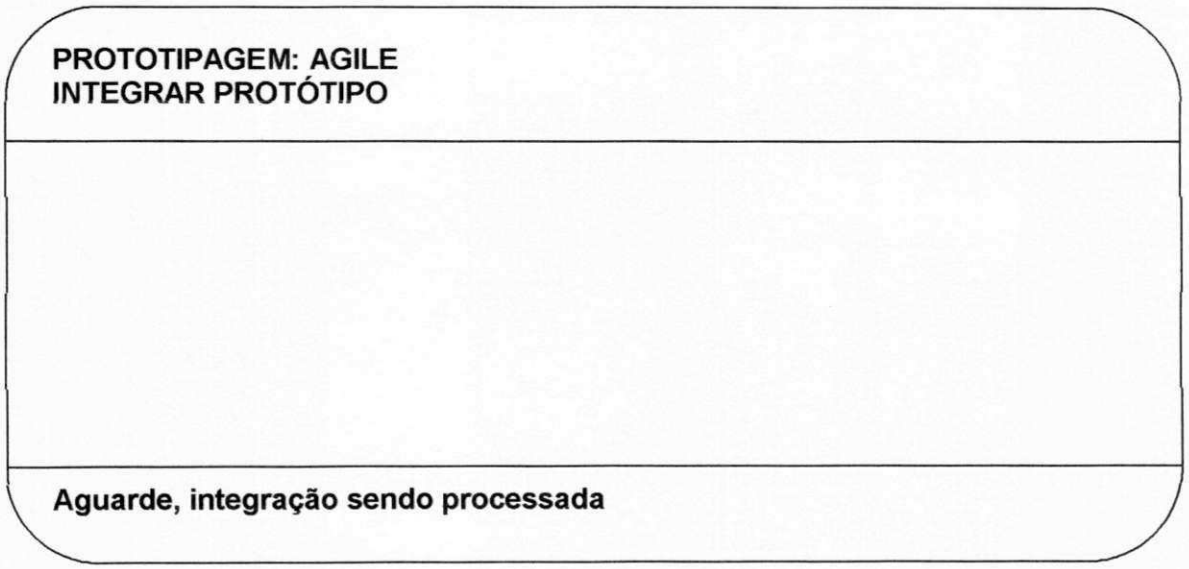

Fig. A.16 – Exemplo de tela da função **INTEGRAR PROTÓTIPO.**

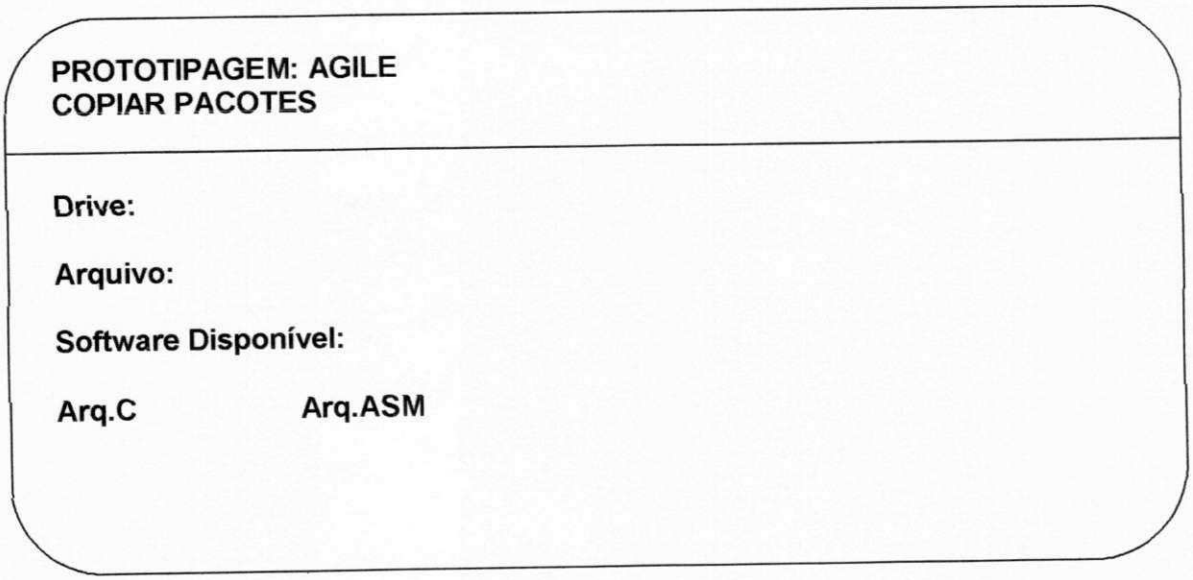

Fig. A.17 – exemplo de tela da função **COPIAR PACOTES**.

APÊNDICE B – Sumário das Funções do Sistema AGILE

- Comando para **entrada no sistema: AGILE**

- Comando para **saída do sistema: SAI**

- Função **Manipular Diretórios:**

Sintaxe: **"Subfunção (d:path\nomedir)"**

Onde, "**subfunção**" é uma das descritas abaixo, " **d** " é o nome do drive para o qual se deseja dirigir, "**path**" é o caminho por onde se deve seguir para caminhar na árvore de diretórios e "**nomedir**" é o nome do diretório desejado.

Abaixo, descrevemos as subfunções que compõem a função:

- **CriaDir:** cria diretório na árvore de diretórios;
- **RemovDir:** remove diretório na árvore de diretórios;
- **SelecDir:** seleciona diretório na árvore de diretórios;
- **RenomDir:** renomeia diretório.

- Função **Manipular Arquivos:**

Sintaxe: Subfunção (d:nomearq.tipo)

Onde, " d " é a denominação do drive onde encontra-se o arquivo, "nomearq" é o nome do arquivo desejado, "tipo" é a extensão dada ao arquivo e "subfunção" é a subfunção que se deseja, a qual pode ser escolhida dentre:

- **CriaArq:** Cria arquivo no diretório atual;
- **SelecArq:** Seleciona arquivo no diretório atual;
- **RemovArq:** Remove arquivo do diretório atual;
- **ImpmArq:** Imprime conteúdo do arquivo;
- **CopiArq:** Copia do arquivo para o arquivo de origem;
- **RenomArq:** Renomeia arquivo.

- **DetalhaDial:** é a função do tipo de diálogo, ou seja, para cada tipo de diálogo existirão as informações de detalhamento correspondentes.
- Exemplos:

  - No tipo de diálogo **MENU**, será preciso detalhamentos sobre as **opções, os comandos aplicáveis, recuperação de erros e ações;**

  - No tipo de diálogo **PERGUNTA E RESPOSTA**, será preciso detalhamento sobre **as respostas, os comandos aplicáveis, a recuperação de erros e as ações;**

  - No tipo de diálogo **COMANDO**, será preciso detalhamento sobre os parâmetros, os comados aplicáveis, a recuperação de erros e as ações;

  - No tipo de diálogo **FORMULÁRIO**, será preciso detalhamento sobre **os campos, os comandos aplicáveis, a recuperação de erros e as ações;**

  - No tipo de diálogo **TEXTO**, será preciso detalhar **os comandos aplicáveis;**

  - No tipo de diálogo **MENSAGENS**, será preciso detalhar os **comandos aplicáveis e as ações;**

- Função **Definir Relatório**

Sintaxe: **DefRel (nomerel, subfunção)**

Onde, "**nomerel**" é o nome do relatório e "**subfunção**" é uma das subfunções descritas abaixo:

- **Editar:** possibilita editar o texto do relatório;
- **Imprimir:** causa a impressão do relatório especificado;
- **Remover:** apaga do arquivo do protótipo o relatório especificado;
- **Copiar:** copia a definição de um relatório origem para um relatório destino.

- Função **Documentar Protótipo**:

Sintaxe: "**Document (d:\path\nomearq.tipo(*), subfunção)**"

Onde, " **d** " é o drive no qual encontra-se o protótipo, "**path**" é o caminho da árvore de diretórios através do qual se atinge o protótipo, "**nomearq**" é o nome do arquivo correspondente ao protótipo, "**tipo**" é o tipo de extensão do arquivo, "**(*)**" correspondente ao default do arquivo do projeto que está sendo prototipado e

- Função **Projetar Telas:**

Sintaxe: "**ProjTela (identela, subfunção)**"

Onde, "**identela** é a identificação da tela e "**subfunção**" pode ser uma das abaixo:

- **DimenTela:** dimensiona a tela do protótipo em linhas e colunas;
- **CriaArea:** cria uma área na tela (identificando-a, dimensionando-a em linhas e colunas e classificando-a em texto ou gráfica);
- **RemovArea:** remove uma área da tela;
- **LocArea:** posiciona uma área em um determinada localização da tela (através de coordenadas especificadas);
- **DetalhArea:** possibilita o detalhamento das informações sobre uma área especificada.

Sintaxe: "**DetalhArea (identarea, subfunção)**"

Onde, "**identarea**" é a identificação da área e "**subfunção**" é uma das descritas abaixo:

- **CursorArea:** permite a entrada de informações sobre o cursor de uma área especificada. Por exemplo, a identificação do símbolo que representará o cursor e o mecanismo de posicionamento do mesmo (teclas, mouse, etc.);
- **DistCor:** possibilita a distribuição de cores em uma área especificada;
- **Movarea:** possibilita a definição da forma de movimentação da informação dentro de uma área. Por exemplo, qual será o dispositivo usado para movimentar a área, como será a movimentação (rolamento de tela ou paginação), etc.

- Função **Compor Diálogo**:

Sintaxe: "**CompDial (tipodial, identdial, subfunção)**"

Onde, "**tipodial**" é o tipo de diálogo que será definido, "**identdial**" é a identificação do diálogo e as subfunções são as descritas abaixo:

- **Editar:** possibilita editar o texto do diálogo;
- **Copiar:** possibilita copiar o texto de um diálogo especificado para um outro diálogo, também especificado;
- **Remover:** remove a definição de um diálogo especificado;

"**subfunção**" é a subfunção que se deseja ativar, a qual deverá ser escolhida entre uma das abaixo:

- **Editar:** possibilita editar o texto do arquivo de definições do protótipo;
- **Imprimir:** possibilita imprimir o arquivo de definições do protótipo.

- Função **Integrar Protótipos:**

Sintaxe: "**IntProt (d:\path\nomearq.tipo (*))**"

Onde, " **d** " é o drive para o qual se deseja dirigir, "**path**" é o caminho a ser seguido na árvore de diretórios para se atingir o arquivo desejado, "**nomearq.tipo (*)**" é o nome do arquivo seguido da extensão, sendo que o default do mesmo é o arquivo que está sendo prototipado no momento.

- Função **Simular Protótipo:**

Sintaxe: "**SimProt (d:\path\nomearq.tipo (*))**"

Onde, as definições dos parâmetros, correspondem às mesmas que foram feitas pra a função Integrar Protótipo.

- Função **Copiar Pacotes**: permite a cópia integral ou de partes de um determinado software.

APÊNDICE C – Teclas de Função (comandos do sistema)

(F1) AJUDA – Mostra o menu de ajuda, permitindo a ativação/desativação de facilidades e mudanças de níveis de ajuda. O nome da função (modo direto) ou tecla (modo dirigido) é usado para chavear a função de Ajuda, isto é, ativar/desativar esta função.

(F2) MARCA – Utilizada para selecionar múltiplas opções em menu. O procedimento consiste em apontar para cada uma das opções desejadas e teclar **MARCA**. Uma vez tendo escolhido todas as opções, tecla-se <**ENTER**>. O nome da função (modo direto) ou tecla (modo dirigido) é usado para chavear entre marcar/desmarcar um item.

(F3) IMPRIME – imprime cópia da tela em exibição.

(F4) RETORNA – retorna um passo na "função que estiver sendo executada sem no entanto, descartar as decisões já tomadas.

(F5) IMTERROMPE – Gera a interrupção temporária de uma "função" em execução para que se execute uma outra. Uma vez esta outra função tendo sido encerrada (**FIMFUNC**), o sistema retorna ao ponto onde foi interrompido na função original.

(F6) DESFAZ – Anula os efeitos da última "subfunção" executada/em execução e volta para o nível imediatamente acima desta.

(F7) ABORTA – Anula os efeitos da "função" em execução e volta para a raiz do menu de funções.

(F8) SALVA – Grava no arquivo do protótipo, tudo que foi executado até então e volta ao ponto onde foi chamado.

(F9) FIMFUNC – Encerra a execução de uma "função" e retorna à raiz do menu de funções.

(F10) SAI – Retorna o controle ao Sistema Operacional, gravando tudo que foi executado até então.

APÊNDICE D – Arquivos do Sistema

Os tipos de arquivos serão identificados pelas suas extensões, portanto, indicaremos a seguir quais são elas e quais as suas descrições:

.DEC Arquivo de descrições e dados de um protótipo, criado pelo usuário ao iniciar o mesmo. Contém informações detalhadas sobre as telas, diálogos, periféricos, dados para simulação, dentre outros, de modo a poder ser interpretado e transformado no arquivo executável do protótipo.

.DOC Arquivo de documentação de um protótipo, criado pelo sistema e atualizado ao longo da prototipagem. Pode ser editado pelo usuário. Contém informações detalhadas de modo a permitir a transferência dos resultados da prototipagem para o ambiente de desenvolvimento da aplicação.

.C Programa fonte, em C, gerado pelo sistema na primeira fase de Integração do protótipo. Pode ser modificado fora do ambiente do sistema e depois importado antes de ser compilado.

.EXE Código executável correspondente à interface prototipada. É gerado pelo sistema na segunda fase de integração do protótipo a partir da compilação e "linkagem" do programa fonte em C representando as características do protótipo com os recursos do sistema que são necessários durante a simulação.

A seguir, sumariamos, para cada tipo de arquivo, as operações que podem ser realizadas sobre eles, tanto pelo sistema quanto pelo projetista.

TIPO DO ARQUIVO	OPERAÇÕES(usuário)	OPERAÇÕES(sistema)
.DOC	Editar, Copiar, Imprimir, Apagar, Renomear	Criar, Atualizar
.EXE	Copiar, Apagar, Renomear	Criar
.DEC	Criar, Copiar, Apagar, Editar Selecionar, Renomear	Atualizar
.C	Remover, Imprimir, Copiar, Renomear	Criar, Atualizar