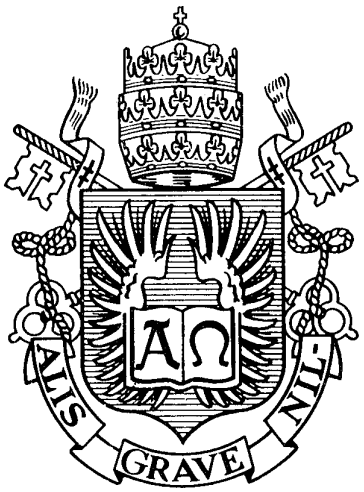


PUC

Série : Monografias em Ciência da computação
No. 5/89

UM GERADOR AUTOMÁTICO DE INTERFACES HOMEM - MÁQUINA

Bernard Lisandre
Hosannah M. Santos Filho

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 – CEP 22453

RIO DE JANEIRO – BRASIL

PUC/RJ - DEPARTAMENTO DE INFORMÁTICA

Série : Monografias em Ciência da Computação No. 5/89
Editor : Paulo A. S. Veloso Abril, 1989

UM GERADOR AUTOMÁTICO DE INTERFACES HOMEM - MÁQUINA *

Bernard Lisandre
Hosannah M. Santos Filho

* Trabalho parcialmente financiado pela SID Informática como parte do projeto ESTRA.

Para obter cópias :
Rosane T. L. Castilho
Assessoria de Biblioteca, Documentação e Informação
Rua Marquês de São Vicente, 225 - Gávea
22.453 - Rio de Janeiro, RJ.
Brasil

UM GERADOR AUTOMÁTICO DE INTERFACES HOMEM - MÁQUINA

Bernard Lisandre
Hosannah M. Santos F.
Departamento de Informática
Pontifícia Universidade Católica
Rio de Janeiro

Resumo

O trabalho apresenta um sistema gerador de interfaces para uma classe muita ampla de aplicações. O protótipo é usado como parte do ambiente PUC* (Planejador da Utilização Casual) que usa também técnicas de inteligência artificial para produzir interfaces ajustadas às características do usuário final. A ferramenta é descrita a partir de sua especificação funcional e do seu uso para a geração de uma interface para uma aplicação simples.

Palavras chaves: interface homem-máquina, ambientes de desenvolvimento, menus, janelas.

Área: metologias de desenvolvimento de sistemas.
Sub-área: CASE.

Público-alvo: projetistas de software.

* Esta pesquisa é parcialmente patrocinada pela SID Informática como parte do projeto ESTRA.

1. INTRODUÇÃO

O uso de ferramentas e ambientes de desenvolvimento vêm gradualmente consolidando a área de Engenharia de Software. A grande maioria dos ambientes existentes foi projetada para o desenvolvimento de grandes sistemas de software básico ou aplicativo. Recentemente reconheceu-se que interfaces homem-máquina são uma forma muito especial de sistemas de software. Por isso, não só é possível automatizar vários aspectos do seu desenvolvimento, através de ambientes apropriados, como talvez as dificuldades do projeto de interfaces tornem o uso deste tipo de tecnologia mais relevante neste caso do que no desenvolvimento de aplicações clássicas. Muitos projetos de pesquisa encontram-se em desenvolvimento nesta área e vários tem sido publicados na literatura recente (exemplo: [1], [2], [3]).

Como ocorre na área de ambientes de desenvolvimento de propósito geral, o uso de técnicas de inteligência artificial tem permitido a incorporação a esses ambientes de conceitos tais como geração e reconhecimento de planos (do usuário), modelamento do usuário e do próprio projetista de uma interface. O projeto PUC, Planejador da Utilização Casual [4] é um ambiente baseado em conhecimento para o projeto e implementação de interfaces homem-máquina com características inteligentes.

O presente trabalho descreve um dos dois protótipos de gerador de interfaces usados no projeto PUC, independentemente dos aspectos de inteligência artificial, os quais tratam a questão de como separar o software da interface do software da aplicação. Este tipo de problema vem motivando várias pesquisas recentes [5].

O protótipo BIG (Beta Interface Generator) permite a especificação, validação (através de simulação) e geração de código para um software de aplicação qualquer definido através de um conjunto de funções. Neste artigo ele é

apresentado através de sua especificação funcional e do seu uso para desenvolver uma interface para um editor de texto clássico.

2. CONCEITOS GERAIS

O termo interface sugere, geralmente, rotinas responsáveis pelo diálogo entre o usuário e a máquina. Para a criação deste diálogo, é necessário um conjunto de elementos básicos. A composição entre elementos básicos permite que se crie uma ampla variedade de diálogos para poder atender de forma bastante flexível às necessidades de um usuário qualquer. É neste conceito que se baseia o gerador de interfaces desenvolvido.

Os elementos básicos definidos são os seguintes :

- Menu
- Janela de Auxílio
- Janela de Diálogo
- Teclas
- Janela de Trabalho
- Operações com elementos

Cada elemento é composto por campos, cada um associado a um novo elemento. Este pode ser um dos elementos básicos citados anteriormente, ou ainda uma função específica da aplicação. No caso do protótipo, a interface controla as chamadas às funções da aplicação.

O conhecimento que o gerador precisa ter sobre a aplicação se limita a uma tabela contendo o nome e os parametros das funções por ela definidas. Portanto, exige-se apenas da aplicação que ela seja construida na forma de um conjunto de rotinas para que possa ser tratada pelo gerador. Outra condição é a de que a aplicação não contenha uma interface própria para que não haja conflito com a definida pelo BIG.

3. CONTATO COM A APLICAÇÁO

3.1 PRESSUPOSTOS BÁSICOS

- Existência de tabelas para referências à funções da aplicação.
- A aplicação não deve possuir rotinas de Menus, Janelas, ...

Sendo o BIG um sistema de interfaceamento não faz sentido que a aplicação também forneça rotinas de Menus, Janelas ... Sendo assim devemos de algum modo permitir que rotinas do BIG sejam acessadas pela aplicação.

A idéia é dar à aplicação condicões para que ela possa manipular os elementos em tempo de execução, já que o conteúdo de determinados elementos bem como outras de suas carecterísticas não podem ser conhecidas a priori. Um exemplo seria :

A aplicação fornecer um diretório, sob a forma de menu, para que o usuário, escolha um deles para efetuar, por exemplo, uma função de Load. Fica claro nesse exemplo que a aplicação não tem como conhecer de antemão os arquivos existentes no diretório especificado, já que isto só ocorre quando da execução do programa.

A solução encontrada para permitir a característica em questão é fornecer uma rotina à aplicação, através da qual ela poderá chamar um elemento. A condição necessária para isto é que o elemento chamado tenha partes a definir.

Cabe à aplicação a chamada da rotina de controle dos elementos quando for conveniente, pois a chamada dessa rotina não é incluida pelo BIG quando da geração do código, ou seja, se a chamada não existir na aplicação, não existirá no código gerado.

3.2 CONTROLE

Existem duas maneiras de encarar o problema.

A primeira pressupõe que o construtor da aplicação conheça o BIG; ou seja, construirá as suas rotinas, que necessitam de chamadas a elementos em tempo de execução, de acordo com as especificações do BIG.

A segunda pressupõe conhecimento da aplicação, sendo que a adaptaçao da aplicação ao BIG é feita pelo projetista através da construção de subrotinas de mascaramento, ou seja, é criada uma nova rotina a qual será constituida da rotina da aplicação mais a chamada da rotina de controle dos elementos. A rotina descrita na tabela da aplicação passa a ser a rotina de mascaramento.

3.3 ATRIBUTOS

Atributos são estruturas de controle utilizada para o encadeamento dos elementos. Cada campo do elemento possui um atributo, ou seja, caso o campo i seja ativado ele executará seu atributo que por sua vez é um novo elemento. O fim de cada uma das sequências de encadeamento ocorre quando é ativada uma função da aplicação pois a mesma não possui atributo.

Elemento	Atributo
Menu	1, x
Janela de Auxílio	2, x
Janela de Diálogo	3, x
Teclas	4, x
Janela de Trabalho	5, x
Operaçoes com Elementos	6, x
Função da Aplicação	11, x, y, z

x : identifica o número ou nome do elemento a ser executado.

y : identifica o número do parâmetro da função.
z : identifica se a função será ou não executada.

Cada campo de um determinado elemento possui um conjunto restrito de atributos possíveis os quais são listados abaixo

Campo i do Elemento	Atributo Possíveis
Menu	1, 2, 3, 5, 6, 11
Janela de Auxílio	-----
Janela de Diálogo	1, 2, 3, 5, 6, 11
Teclas	-----
Janela de Trabalho	1, 2, 3, 5, 6, 11
Operaçoes com Elementos	-----
Função da Aplicação	-----

3.4 ESTRUTURA DE DADOS

A estrutura de dados adotada está muito ligada a definição dos elementos, ou seja, tem poucas diferenças com relação a aquisição de dados dos elementos.

Apesar do preenchimento, dos elementos secundários, não ser obrigatório os seus valores sempre estarão definidos; ou seja, ao criarmos um novo elemento, caso seus elementos secundários não sejam definidos pelo projetista, estes serão definidos através de default do programa.

Os dados de controle são utilizidados quando da verificação de consistência, que é a fase intermediária entre a aquisição de dados e a construção do código.

O programa foi dividido nas seguintes partes lógicas:

I. Aquisição de dados e construção do Metafile.
II. Verificação de consistência da interrelação dos elementos.
III. Geração do código, a partir do Metafile gerado na parte I.

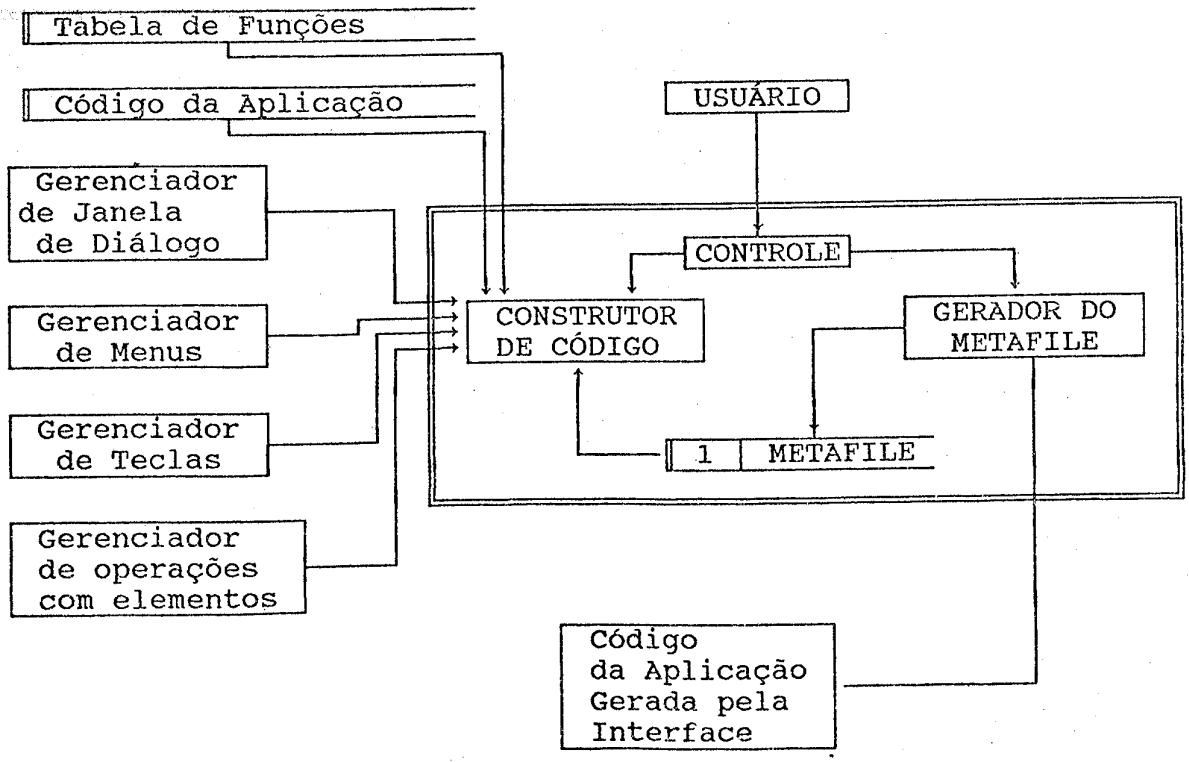
Essa divisão apresenta as seguintes vantagens :

- Relativa independência da aplicação quando do uso das Parte I e Parte II, sendo que só a tabela de funções da aplicação é necessária.

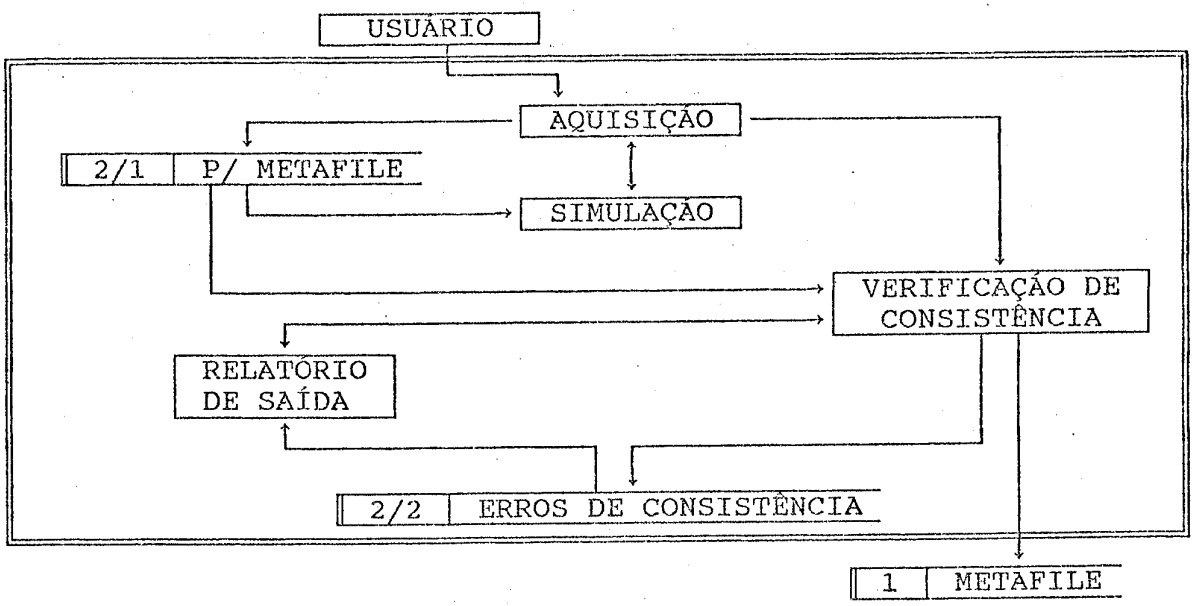
- Maior facilidade de correção (Parte I e Parte II), haja visto que não havendo geração de código se permite uma redefinição dos elementos em qualquer momento.

- Maior facilidade de simulação, possível a qualquer momento, sendo esta ligada simplesmente ao layout da tela, ou seja, sem se preocupar com a execução da função da aplicação com a qual se está trabalhando.

DIAGRAMA DE FLUXO DE DADOS (DFD)



GERADOR DE INTERFACE BETA (BIG) - NÍVEL 1



GERADOR DO METAFILE - NÍVEL 2

4. UM EXEMPLO DE INTERFACE GERADA PELO BIG :

A idéia é mostrar o processo de construção de uma interface utilizando-se o BIG. Para isto escolhemos como aplicação um editor de texto, por ser esta uma aplicação muito difundida. As funções do editor foram bastante simplificadas uma vez que o objetivo principal do trabalho não é o editor ou a sua descrição, mas sim a descrição da interface.

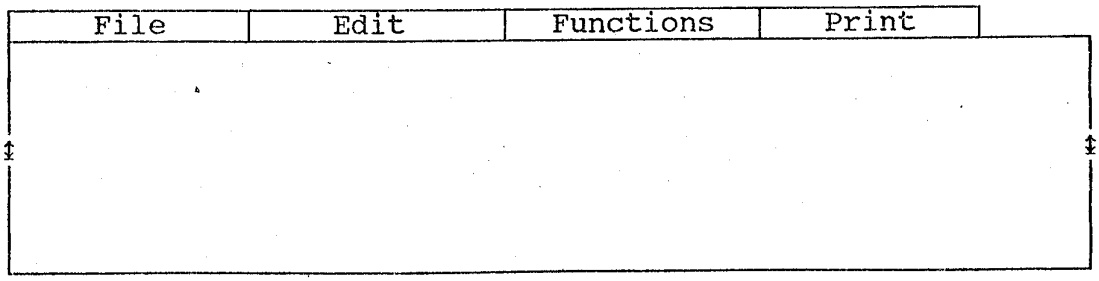
Conhecimento sobre a aplicação :

- Trata-se de um editor de texto muito simples.
- Funções (da aplicação) disponíveis e parâmetros
  - Salvar texto editado(NOME)
  - Carregar texto para edição(NOME)
  - Mudar nome do texto editado(NOME)
  - Sair do editor(sem parâmetro)
  - Inserir linha(No. da LINHA,PAGINA)
  - Deletar linha(No. da LINHA,PAGINA)
  - Deletar palavra(No. da LINHA,No. da COLUNA,PAGINA)
  - Deletar caracter(No. da LINHA,No. da COLUNA,PAGINA)

Marcar início de bloco(No. da LINHA,PAGINA)
Marcar fim de bloco(No. da LINHA,PAGINA)
Mover bloco(No. da LINHA,PAGINA)
Copiar bloco(No. da LINHA,PAGINA)
Modo INSERÇÁO(sem parâmetro)
Mover um caracter para frente(LINHA,COLUNA,PAGINA)
Mover um caracter para tras(LINHA,COLUNA,PAGINA)
Mover um caracter para cima(LINHA,COLUNA,PAGINA)
Mover um caracter para baixo(LINHA,COLUNA,PAGINA)
Mover uma página para cima(LINHA,PAGINA)
Mover uma página para baixo(LINHA,PAGINA)
Mover para início do texto(sem parâmetro)
Mover para fim do texto(sem parâmetro)
"Backspace"(LINHA,COLUNA,PAGINA)

Todas estas funções poderão ser acessadas através de um menu e/ou através de uma tecla global.

O responsável pelo projeto da interface optou por uma tela permanente (ou inicial) do seguinte tipo:

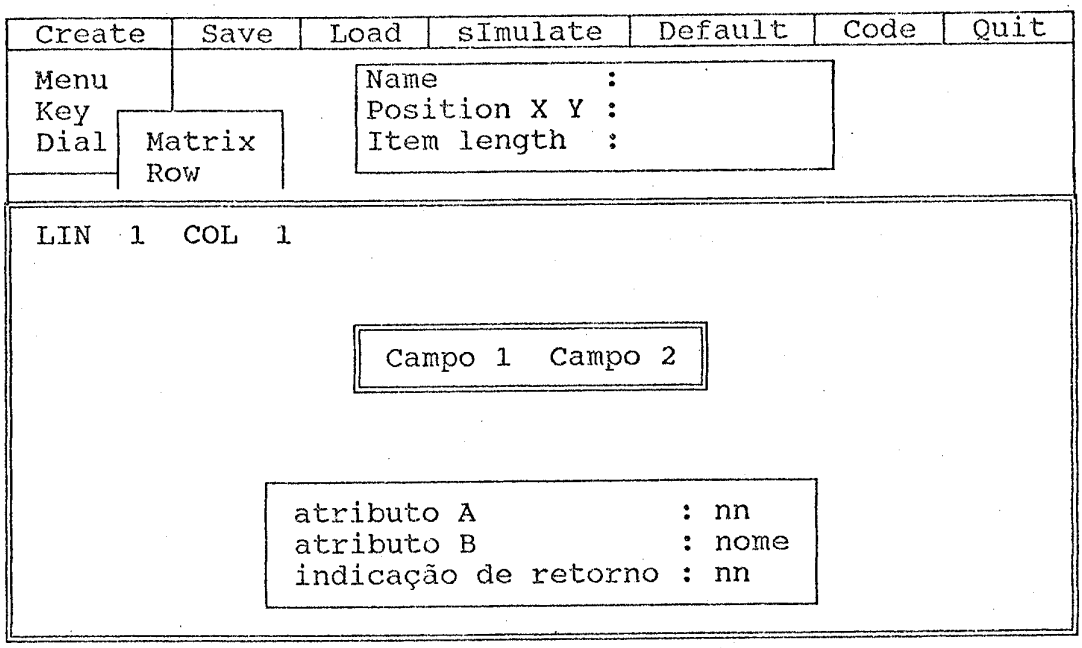


Para projetar esta tela com o BIG é preciso definir um menu linha, e associar a esse elemento um conjunto de teclas locais. Estes elementos serão definidos a seguir, assim como o elemento Janela de Diálogo.

4.1 MENUS

-tipo do menu = 2(menu linha)
-nome do menu= inicial
-dimensão dos campos = 12
-posição do menu = (1,1)

O resto das informações necessárias será adquirido dinâmicamente através da seguinte tela (do BIG):



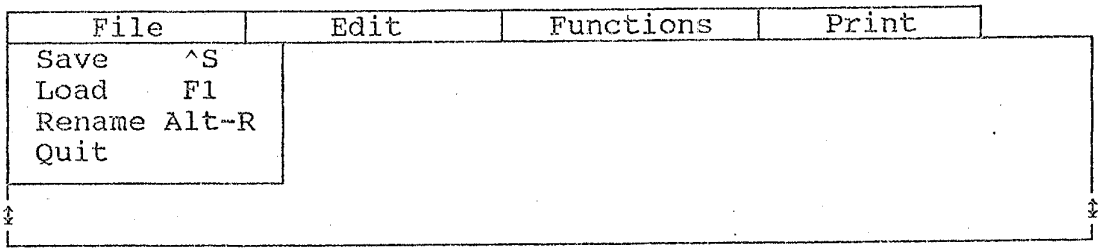
Ou seja, o projetista tem na tela o que será mais tarde o menu gerado pelo BIG. Através das teclas de função, ele pode dimensionar o menu a seu critério.

A cada campo serão associados dois atributos :

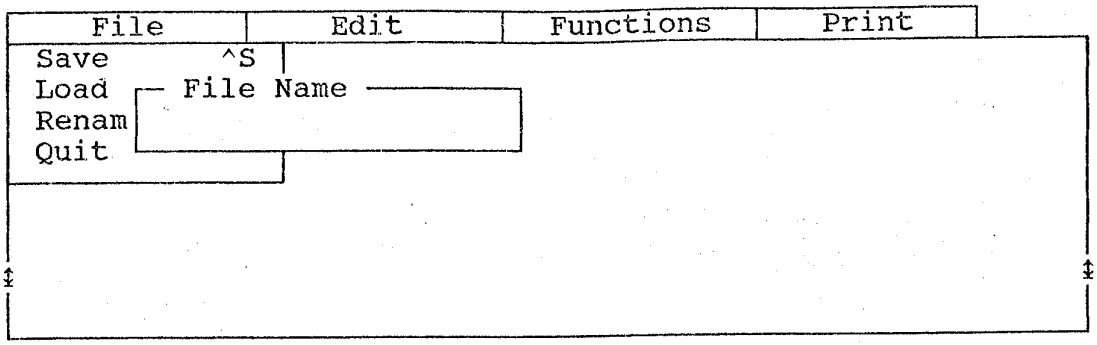
- Atributo A : referente ao tipo do elemento chamado;
- Atributo B : referente ao nome do elemento chamado;

Para que não haja erro, estes elementos chamados deverão existir na hora da geração do código; caso contrario, o gerador acusará erro e não gerará código.

O processo de aquisição para os outros menus é semelhante, como no exemplo do menu coluna associado a execução do campo 1 do menu "inicial" mostrado a seguir.



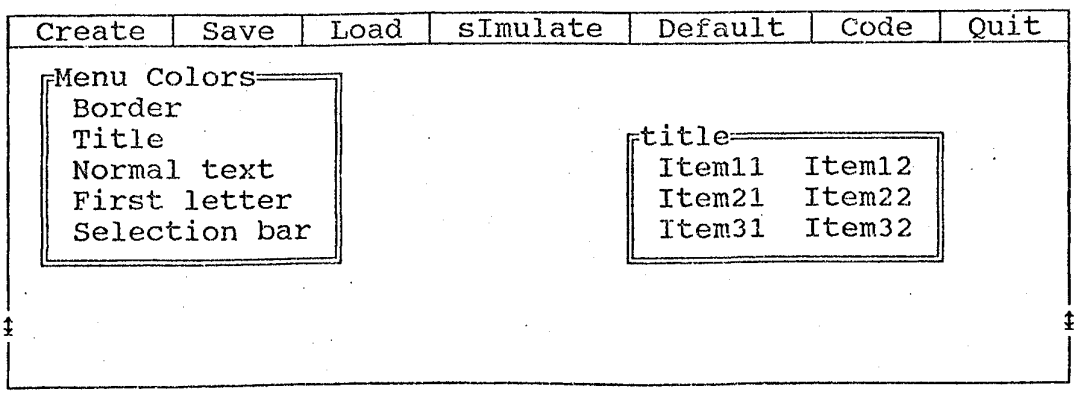
Um exemplo de janela de diálogo é mostrado a seguir, e está associado a execução do campo 1 do menu "File". De fato trata-se de uma chamada à uma função da aplicação que necessita de um parâmetro externo fornecido pelo usuário potencial da interface. O projetista precisa então definir esta janela.

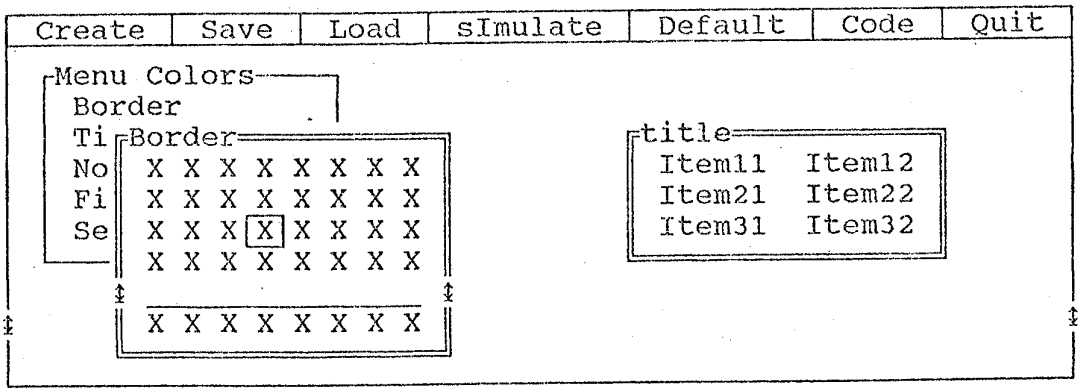


A definição das cores dos elementos pode ser feita de duas formas. Geralmente o usúario deseja manter uma certa padronização para as cores de cada elemento. Por isso, o BIG permite que se definam cores "default" para cada elemento. Se não existir um pedido explícito do projetista para alterar as cores de um determinado elemento, todos os elementos do mesmo tipo terão as mesmas cores.

A outrà forma consiste num pedido explicito quando da definição do elemento através de uma tecla de função.

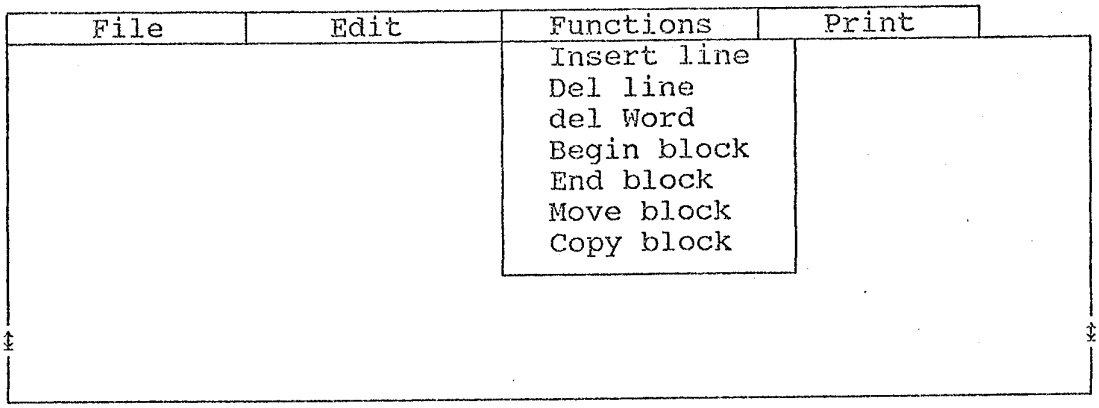
A aquisição das cores se faz através da seguinte tela :





Seleciona-se uma opção no menu(Border por exemplo), e em seguida escolhe-se a cor que se deseja para o menu, movimentando as setas até posicionar o quadrado sobre a opção desejada. Na janela de "exibição a direita pode-se observar o resultado obtido com as mudanças de cor. Em seguida, tecla-se Enter para validar a opção.

Outras funções acessíveis por um menu :



O restante das funções será somente acionado por uma tecla global ou local definida explicitamente pelo projetista. Por exemplo a tecla — associada a execução de "Backspace".

4.2 TECLAS

GLOBAIS

São teclas válidas, independentemente de que elemento esteja ativo.

Na interface acima temos como teclas globais:

- F1 desativa todo e qualquer elemento não permanente e ativa em seguida o menu File e a janela de diálogo Load.
- ^S (CONTROL-S) executa a função da aplicação Save.
- ALT-R ativa a janela de diálogo Rename sem que seja desativado nenhum dos elementos.

LOCAIS

São as teclas válidas somente enquanto um determinado elemento estiver ativo: as teclas locais têm prioridade maior que as globais.

Na interface acima temos como teclas locais:

- menu principal
  as letras F, E, P que respectivamente executam os elementos associados aos campos File, Edit e Print.
- menu File
  as letras L, S, R, Q que respectivamente executam os elementos associados aos campos Load, Save, Rename e Quit.
  ->, <- desativam o menu atual e ativam respectivamente o campo Edit e o campo Print do menu principal.

4.2.1 DEFININDO AS TECLAS

- Quando se define um menu, automaticamente fica definido um conjunto de teclas locais, composto pela primeira letra maiúscula de cada campo. Essas teclas têm como efeito a ativação do campo do menu em questão. Portanto as teclas locais F, E e P do menu principal e as teclas locais L, S, R e Q do menu File, já se encontram automaticamente definidas.

- Para definição das teclas locais ->, <-, F1, ALT-R é necessário uma estrutura de "navegação" entre os elementos, de modo a permitir que os elementos possam ser ativados ou desativados.

Uma vez que o BIG foi desenvolvido num PC utlizando-se como sistema operacional o DOS, optou-se por fazer a "navegação" entre os elementos de forma análoga a "navegação" entre diretórios, ou seja:

..\ retorna ao elemento anterior.

\ retorna ao elemento raiz ativando o campo .

{ } ativa o elemento contido dentro da chave desde que conste dentro da chave o tipo de elemento e seu nome.

yyyy ativa o campo yyyy do elemento em questão.

Vamos exemplificar a sintaxe acima com as teclas que ainda restam por definir.

->

..\Edit

fecha o menu File retornando ao menu principal, ativando o campo Edit do mesmo.

<-

..\Print

fecha o menu File retornando ao menu principal, ativando o campo Print do mesmo.

F1

\File\Load

retorna ao elemento raiz, ativa o campo File do menu principal, ativando em seguida o campo Load do menu File.

^S

{Aplic Save}

identifica que a função da aplicação Save deve ser executada.

ALT-R

{Dial Rename}

identifica que a janela de diálogo Rename deve ser aberta.

5. OUTRAS FUNÇÕES DO BIG

CREATE

Permite que se definam os elementos que irão compor a interface

SAVE

Salva em um arquivo os elementos definidos até a ativação deste comando.

LOAD

Lê um arquivo e carrega os elementos em memória descartando os elementos que estavam anteriormente em memória.

DEFAULT

Permite que se escolham os defaults para cada tipo de elemento definido pelo BIG. Esses defaults valem para todos os elementos do mesmo tipo definidos após, até que sejam novamente alterados.

CODE

Gera o código em linguagem C, dos elementos que estão em memória. A partir disto, basta compilar a sua aplicação com a interface gerada. O BIG requisita o nome do arquivo no qual se deseja gravar o código gerado.

SIMULATE

Permite fazer uma simulação da interface ainda quando da sua construção através do BIG e antes da geração de código.

6. CONCLUSÕES

O modelo de implementação adotado no sistema BIG recomenda a sua adoção no ambito mais amplo do projeto PUC. Desenvolvimentos futuros envolvem a sua re-implementação sobre X-Windows em ambiente UNIX, quando será explorada a idéia de desenvolvimento cooperativo de uma interface.

REFERÊNCIAS

[1] Hill, R.D.; "Supporting Concurrency, Communication and Synchronization in Human Computer Interaction - The Sassafras UIMS"; ACM Trans. on Graphics, 1986.

[2] Lieberman, H; "There is More to Menu Systens than Meets the Screen"; ACM Computer Graphics, 1985.

[3] Myers, B.A.; "Creating highly - interactive and graphical user interfaces by demonstration"; ACM Computer Graphics, 1986.

[4] Guaranys, P.Y.; Lucena, C.J.P.; "PUC: A knowledge Based Environment for Planned User Communication"; Monografia em Ciência da Computação, Departamento de Informática, PUC/RJ 1988.

[5] Szekely, P.; "Separating the User Interface from the Functionality of Application Programs"; Tese de Doutorado, Carnegie - Mellon University, 1988.