Uma Ferramenta Baseada em Tabelas *UAN* para Apoio à Modelagem de Interfaces

Elton José da Silva*
Beatriz Mascia Daltrini **
José Geraldo Veloso Moreira*
Igor Prata Soares*

* *{elton, jgvm, igor}@iceb.ufop.br*
Departamento de Computação
Universidade Federal de Ouro Preto (UFOP)
Campus Morro do Cruzeiro, Ouro Preto - MG, CEP 35400-000

** *beatriz@dca.fee.unicamp.br*
Departamento de Computação e Automação Industrial
Faculdade de Engenharia Elétrica e de Computação
Universidade Estadual de Campinas (UNICAMP)
Caixa Postal 6101, Campinas - SP, CEP 13083-970

Resumo

A crescente complexidade dos sistemas interativos tem chamado a atenção dos projetistas para a importância da tarefa de especificação da interface. Dentre as principais técnicas de especificação de interfaces destaca-se a User Action Notation (UAN), que consiste numa linguagem simples, concisa, centrada nas tarefas e ações do usuário e apropriada para descrever interfaces de manipulação direta. Neste trabalho, apresenta-se uma ferramenta CASE, baseada no formalismo UAN, para apoiar a especificação de interfaces. Com esta ferramenta é permitida a definição de tarefas do usuário, tabelas que relacionam ações do usuário às respostas da interface, diagramas de transição de estados, diagramas de transição de tarefas e dicionário de dados. O sistema proposto possui uma estrutura modular para implementar cada uma dessas tarefas, onde o projetista, utilizando uma abordagem top-down, define a interface a partir de níveis mais altos de abstração até o nível de ações do usuário.

Palavras-chave: projeto de interfaces, técnicas e ferramentas de especificação, *User Action Notation (UAN).*

1. INTRODUÇÃO

Quando o conceito de *interface homem-computador* começou a surgir, associava-se *interface* ao hardware e software através dos quais as pessoas e os computadores podiam se comunicar. Naquela época, uma boa interface poderia ser, por exemplo, aquela que levasse o usuário, os cientistas da época, a manipular o menor número de válvulas, alavancas e botões. Daquela época até os dias atuais houve uma grande evolução, sendo que no espaço de menos de dez anos a interação entre o usuário e o computador mudou de um simples diálogo baseado numa troca de caracteres alfanuméricos para as conhecidas interfaces que utilizam janelas, ícones, botões, menus e síntese de voz.

Atualmente, uma grande parte dos recursos computacionais é dedicada exclusivamente para tornar mais fácil a interação com o usuário. A *comunicação* com o sistema se tornou pelo menos tão importante quanto a *computação* feita pelo mesmo. Hoje, considera-se a interface não só o meio físico que separa duas entidades distintas -- usuário e computador -- em seu processo de comunicação; ela funciona como um mediador que suprime as deficiências de cada entidade ao longo da interação [17].

À medida que as interfaces ficam mais fáceis de aprender e usar, elas se tornam mais difíceis de especificar e programar de forma clara. Projetistas de interfaces, geralmente, usam técnicas informais ou *ad hoc* na fase de especificação. Essas técnicas, que são incompletas ou ambíguas, fazem com que os desenvolvedores interpretem as

especificações de forma diferente. Isto pode levar a várias inconsistências e mal-entendidos que se propagam para os estágios finais de integração do sistema. Com o objetivo de reduzir esses problemas, existe hoje uma grande preocupação em se produzir especificações mais precisas e consistentes para comunicar a aparência e funcionalidade da interface entre os membros de uma equipe de desenvolvimento. Hix, em [9], comenta a necessidade de técnicas efetivas de especificação no desenvolvimento que enfatiza refinamentos sucessivos e envolve diferentes grupos de trabalho para a produção de uma interface.

Em [11, 12], é enfatizada a importância da especificação da interface antes da sua construção, pois o projetista poderá assim descrever e estudar uma variedade de interfaces, sem ter o trabalho de codificá-las primeiramente. Com uma boa técnica de especificação pode-se descrever de forma precisa uma interação usuário-computador, livre das restrições relacionadas com a sua implementação. Para se fazer um bom projeto de interface é interessante ter em mãos uma boa notação para armazenar e discutir possibilidades durante o seu desenvolvimento.

O restante deste trabalho está organizado da seguinte forma: a próxima seção apresenta algumas técnicas de especificação de interfaces, destacando vantagens e problemas com a utilização de cada uma delas. Na seção 3 apresenta-se o formalismo *UAN*, uma notação orientada a ações e tarefas do usuário, para descrever interfaces. Na seção 4 apresentamos a arquitetura de uma ferramenta CASE, baseada em *UAN*, para dar suporte à tarefa de especificação de interfaces. Na seção 5 mostramos alguns aspectos da implementação desse modelo em uma plataforma específica. Na seção 6 apresentam-se as conclusões relativas à experiência de utilização do modelo aqui apresentado e perspectivas para trabalhos futuros.

2. ALGUMAS TÉCNICAS DE ESPECIFICAÇÃO DE INTERFACES

No projeto de software convencional não existe uma técnica de especificação para todo e qualquer tipo de aplicação. A situação não é diferente para o caso de software interativo. Infelizmente, nenhuma das técnicas atualmente existentes engloba todas as características desejáveis numa notação, mesmo porque muitas dessas características são conflitantes entre si. Por exemplo, se a geração automática do código da interface é contemplada pela ferramenta que implementa uma técnica, provavelmente os resultados da especificação não serão de fácil entendimento por toda a equipe que está desenvolvendo a interface. Nenhuma notação é apresentada como a solução para os problemas de especificação de interfaces, bem como ainda não existem regras de ouro para a aplicação desta ou daquela notação.

As principais técnicas de especificação de interfaces, principalmente aquelas para projeto de diálogo de baixo nível, são apresentadas na tabela 1. Para cada uma delas, destacamos as principais vantagens e problemas com a sua utilização. Um estudo mais detalhado dessas técnicas, com exemplos de aplicação, pode ser encontrado em [24].

3. USER ACTION NOTATION (UAN)

O formalismo *UAN* consiste em uma notação orientada a tarefas e ações do usuário, desenvolvida para descrever o projeto de interfaces de manipulação direta[1] [8, 25, 26]. Esse formalismo foi utilizado inicialmente para auxiliar projetistas e programadores do *Dialogue Management Project*[2]. O idealizador da *UAN*, Antonio C. Siochi, propôs a notação, devido aos problemas constantes de grandes volumes de especificação de projetos de interface. Desde a sua criação, muitas equipes de projeto têm utilizado a *UAN* na descrição de diversos tipos de interfaces, contribuindo assim para a sua extensão e formalização [2, 13, 24]. Em particular, algumas das instituições que têm utilizado a *UAN* de forma experimental são:

[1] As abordagens de especificação baseadas em gramáticas e diagramas não expressam convenientemente a variedade de ações permissíveis e o *feedback* visual que as interfaces de manipulação direta fornecem.

[2] Desenvolvido no Departamento de Ciência da Computação *do Virginia Polytechnic Institute and State University, USA.*

Técnica	Vantagem	Problemas
Gramática [20, 21, 22]	- Especifica bem interfaces baseadas em linguagens de comando. - *TAG (Task Action Grammar)* enfoca a noção de consistência da gramática. - *CLG (Command Language Grammar)* cria uma estrutura para descrever vários aspectos da interface em diferentes níveis.	- É pobre para a descrição de aspectos dinâmicos de interfaces de manipulação direta.
Árvore de Menus [23]	- Especifica bem estilos de seleção. - Mostra de forma sucinta a estrutura completa e detalhada do sistema. - É fácil conferir a consistência, completude, ambigüidade e redundância.	- Mais voltada para o domínio de construção do software, e não da interação. - Gera basicamente diálogos seqüenciais. - Não apresenta a estrutura completa de ações possíveis do usuário.
DTE (Diagrama de Transição de Estados) [3, 5, 11]	- Apresenta de forma explícita o conceito de estado, regras de transição associadas e seqüencialidade de ações. - Representa graficamente o fluxo de controle do diálogo.	- Não há como fatorar transições semelhantes (ocasionadas por um mesmo evento) de um grupo de estados. - Não fornecem uma representação natural para expressar intercalação e paralelismo entre tarefas numa interface gráfica.
Estadograma [7, 18]	- Aumenta o poder de expressão dos DTEs. - É um modelo gráfico. - Especifica hierarquia e concorrência entre estados. - Permite especificar transições condicionais entre estados.	- A sua compreensão não é tão simples como a dos DTEs. - Mais voltado para o domínio de construção da interface.
Modelo de Eventos [5]	- Possui alto poder de descrição (tanto diálogos seqüenciais como diálogos assíncronos).	- Dificuldade de se criar código correto devido ao fluxo de controle não ser localizado. - O entendimento da especificação é dificultado à medida que esta cresce, pois não existe uma hierarquia bem definida.
GOMS (Goals, Operators, Methods and Selection rules) [1, 4, 6, 10, 15, 16]	- É o mais estudados dos modelos analíticos, é o que tem recebido mais testes empíricos. - Algumas extensões permitem tratar de imprecisão nas regras de seleção. - Especifica a interface do ponto de vista das tarefas do usuário.	- Notação é difícil de usar. - É falho para a especificação de tarefas que envolvam processamento paralelo, as que não possuem uma hierarquia bem definida e as sujeitas a erros freqüentes de usuários.

Tabela 1: *Comparando algumas técnicas de especificação de interfaces*

- Laboratório de Interfaces do *Texas Instruments* (Dallas, Texas), em projetos de interfaces para telefonia.
- *Bureau of Land Management* (Denver, Colorado), no desenvolvimento de sistemas de informação geográfica.
- *NCR Corporation*, na representação de interfaces multimídia.
- *Laboratoire de Génie Informatique, Institut IMAG* (Grenoble, França), na descrição de uma interface multimodal para um sistema de informação sobre viagens aéreas.
- Laboratório de Computação e Automação, da FEEC/Unicamp (Campinas, Brasil), na descrição de um modelador geométrico para o ambiente *ProSIm* (Prototipação e Síntese de Imagens Fotorealistas e Animação).

Conforme comenta Shneiderman, em [23], a *UAN é uma notação compacta e poderosa; uma abordagem de alto nível para especificar o comportamento de sistemas interativos e descrever ações do usuário*. As principais

características que justificam o uso da notação como base para um modelo de ferramenta de especificação de interfaces são:

- produz especificações concisas, precisas, não ambíguas e fáceis de entender após pouco tempo de uso, facilitando a comunicação entre os membros da equipe de desenvolvimento de interfaces;
- embora se destaque como um modelo de síntese de interfaces (suporta a criação e documentação) a *UAN* pode ser utilizada para analisar a estrutura e detectar possíveis ambigüidades e inconsistências no projeto;
- é uma técnica mais voltada para especificar o projeto da interação, e não o projeto do software da interação;
- apresenta, de forma simples, suporte para especificação de relacionamentos temporais entre tarefas, como por exemplo: concorrência, intercalação e interrupção;
- utiliza a natureza gráfica dos *DTEs* para uma representação completa dos estados da interface;
- oferece suporte para a especificação do *feedback* da interface para as ações do usuário.

Em *UAN*, uma interface é representada como uma estrutura hierárquica de *tarefas assíncronas*. As ações do usuário, o *feedback* da interface e as informações de mudança de estado são representadas num nível de abstração mais baixo, denominado *nível articulatório*. Níveis de abstração mais altos, ou níveis semânticos, escondem esses detalhes e são necessários para construir a estrutura das tarefas na interface. Em todos os níveis, as ações do usuário e as tarefas são combinadas com relações temporais. As descrições de tarefas a nível de ações do usuário são apresentadas em forma de tabela, como no exemplo apresentado na figura 1. O sentido de leitura de uma tabela de especificação deve ser de cima para baixo e da esquerda para a direita. Neste exemplo é descrita uma interação para apagar um arquivo num aplicativo hipotético denominado *FManager*[3]. A descrição prosaica dessa interação é dada a seguir:

- Mover o cursor para o ícone do arquivo a ser apagado.
- Pressionar o botão esquerdo do mouse sobre o ícone do arquivo, que é destacado com uma mudança de cor.
- Mantendo o botão do mouse pressionado, movimentar o cursor para arrastar a representação do ícone do arquivo para o contexto do ícone de cesta de lixo.
- Liberar o botão do mouse; o ícone do arquivo desaparece. Algum tipo de destaque no ícone cesta de lixo deve ser dado para indicar que o arquivo ainda pode ser recuperado.

Uma possível descrição dessa interação em *UAN* pode ser feita da seguinte forma:

Tarefa: Apagar um arquivo				#
Ações do usuário	**Feedback da Interface**	**Exceções**	**Estados da Interface**	**Conexões**
~[Arquivo']; M*l*∨	Arquivo'-! ? Arquivo'!		Arquivo Selecionado	
~[CestaLixo];	Contorno(Arquivo')>~ CestaLixo!			
M*l*∧	Apaga(Arquivo') Apaga(Contorno(Arquivo')) CestaLixo!	@x',y'∉[CestaLixo] ? Voltar(Arquivo)	Arquivo Apagado	// Marcar o arquivo como apagado

Figura 1: *Um Exemplo de Especificação de uma Tarefa em UAN*

3.1. Onomatopéias Visuais da *UAN*

Os símbolos da notação *UAN* foram escolhidos, em sua maior parte, com o propósito de serem visualmente onomatopéicos. Por exemplo, o sinal gráfico de til ("~") sugere movimento, "[X]" sugere uma caixa em volta de X representando o contexto do objeto, o sinal de exclamação ("**!**") sugere algo que chama atenção, apropriado ao *feedback* da interface e o sinal de maior (">") reflete a noção de seguir, acompanhar. No exemplo da figura 1, para representar que um contorno do arquivo se move pela tela acompanhando o movimento do cursor para a cesta de lixo, foi usado o símbolo ">~". Os símbolos para especificar as ações do usuário são escolhidos de forma que representem uma *mímica* das ações, tais como os símbolos "∨" e "∧" para as ações de pressionar e liberar um dispositivo, como por exemplo, o botão esquerdo do mouse, denotado por "M*l*". As tabelas a seguir resumem os principais símbolos utilizados na *UAN* [8, 9, 24].

[3] similar ao utilitário *FileManager*, do *OpenWindows 3.0*

O que é representado	Símbolo UAN	Significado
Agrupamento	( )	Mecanismo de agrupamento de tarefas.
Seqüência	A B	Tarefas A e B devem ser executadas na ordem da esquerda para a direita ou de cima para baixo ($^{A}{}_{B}$).
Escolha	A | B	Escolha de tarefas (útil para mostrar tarefas alternativas).
Escolhas repetidas	$(A \mid B)^*$	Escolha de A ou B, nenhuma vez ou várias vezes sucessivas.
Independência na ordem de execução	A & B	Tarefas A e B são independentes quanto à ordem de execução.
Interrupção	$A \rightarrow B$	Tarefa A pode interromper a tarefa B.
Não interrupção	<A>	Tarefa A não pode ser interrompida.
Intercalação	$A \leftrightarrow B$	Tarefas A e B podem ser intercaladas no tempo.
Concorrência	A || B	Tarefas A e B podem ser executadas simultaneamente.
Espera	A (t > n) B	Tarefa B é executada depois de uma espera de mais de n unidades de tempo após a ocorrência da tarefa A.
Mudança de Intenção	A ; B	A tarefa B não precisa necessariamente ser executada.
Repetição de tarefas	A^*	Tarefa A é executada zero ou mais vezes.
	A^+	Tarefa A é executada uma ou mais vezes.
Condição	A ? B	Se a condição A for satisfeita então executar a tarefa B.
	A ? B : C	Se A então B, senão C.

Tabela 2: *Símbolos UAN para representar relacionamento entre tarefas do usuário*

O que é representado	Símbolo UAN	Significado
Objetos e Contextos	X’	Objeto X específico.
	[X]	Contexto do objeto X, através do qual ele é manipulado.
	[X.região]	Contexto do objeto X, descrevendo a região através do qual ele é manipulado numa determinada operação.
Movimento do cursor	~	Move o cursor.
	~[X]	Move o cursor para o contexto do objeto X.
	[X]~	Move o cursor para fora do contexto de X.
	~[x, y]	Move o cursor para uma posição arbitrária (x, y).
	~[x, y] in A	Move o cursor para um ponto arbitrário dentro do objeto A.
	~[X in Y]	Move o objeto X para dentro do objeto Y.
Dispositivos e ações primitivas do usuário	Xv	Aperte o botão, chave ou interruptor X.
	X^	Libere o botão, chave ou interruptor X.
	K(“abc”)	Entrada de um literal, “abc”, via teclado.
Apagar objeto	apaga(X)	Apaga o objeto X da tela.
Exibir objeto	exibe(X)	Mostra o objeto X na tela.
Contorno de um objeto	contorno(X)	Apresenta o contorno do objeto X na tela.
Arrastar objeto	X >~	O objeto X é arrastado pelo cursor.
Rubberbanding	X >>~	O objeto X muda de tamanho enquanto segue o cursor.
Destaque	X!	Destaca o objeto X.
	X!!	Destaque especial para o objeto X.
	(X!-!)*	Objeto X piscando na tela.
Localização	@ x’, y’	numa posição específica (x’, y’).
	@ X	na posição onde se encontra o objeto X.
Feedback num conjunto de objetos	$\forall, =, \subset, \not\subset, \in, \notin, \neq$	possuem o mesmo significado matemático.

Tabela 3: *Símbolos UAN para representar ações do usuário e feedback da interface*

4. UM MODELO DE FERRAMENTA

A utilização de qualquer metodologia de desenvolvimento de software pode ser bastante melhorada através da utilização de ferramentas automáticas que apoiem essa tarefa. Para o caso de software interativo, a falta de ferramentas de suporte ao projetista durante a fase de especificação da interface tem levado a sistemas muitas vezes difíceis de usar. Na figura 2 apresentamos uma estrutura modular, baseada no modelo básico de tabelas da notação *UAN*, para dar suporte ao projetista durante a fase de especificação da interface. Este modelo, permite ao projetista

utilizar uma abordagem top-down de desenvolvimento, ou seja, descrever a interface a partir do nível de tarefas até o nível de ações do usuário.

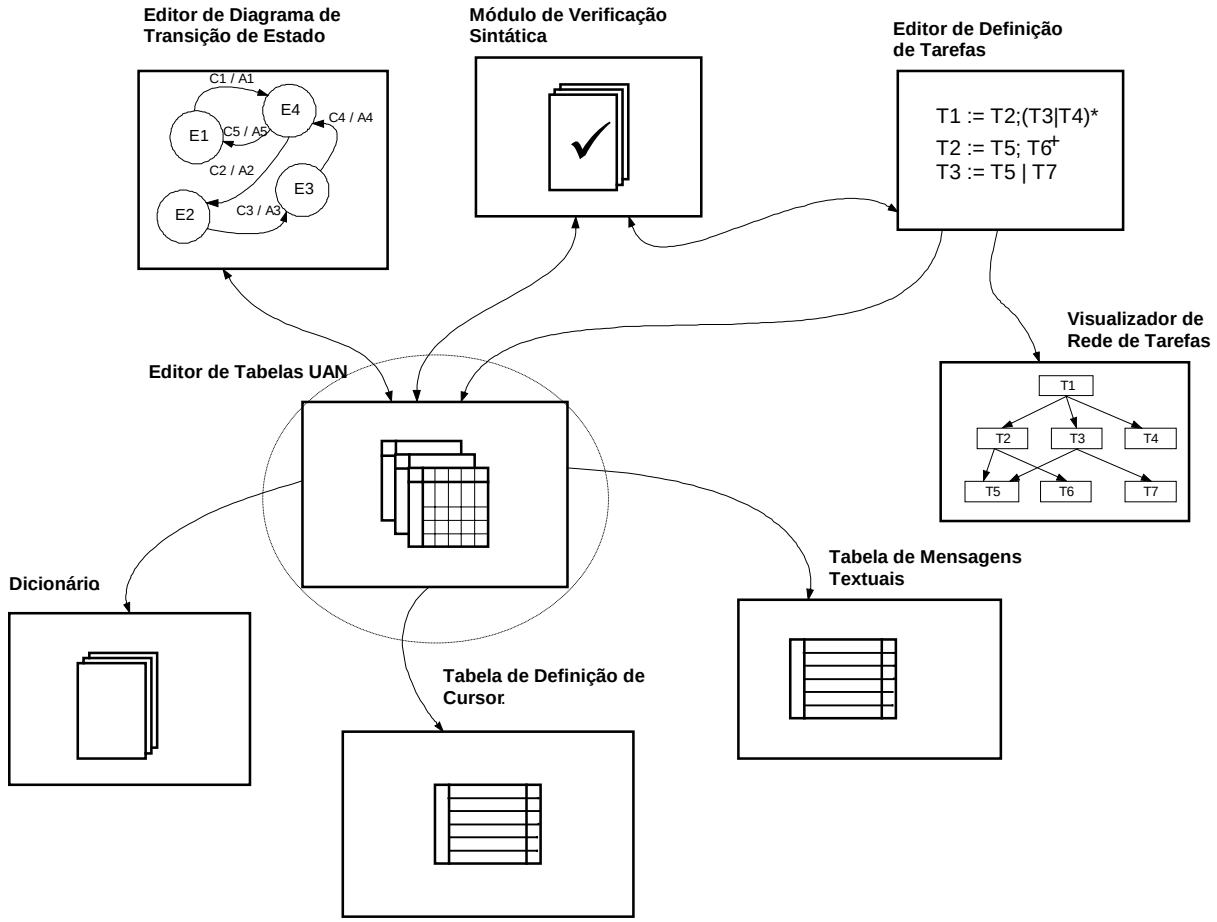

Figura 2: *Um modelo de ferramenta para especificação de interfaces*

Os componentes deste modelo são descritos a seguir:

- **Editor de Definição de Tarefas**: responsável pela definição das tarefas e o relacionamento entre elas em níveis mais altos de abstração, utilizando os operadores temporais descritos na tabela 2. A preocupação em especificar as relações temporais entre as tarefas nas fases inicias de projeto força os projetistas a pensarem sobre os problemas de temporização bem antes da implementação da interface.
- **Editor de Tabelas *UAN***: é o item central de uma especificação. Possibilita ao projetista descrever as tarefas no nível articulatório, que envolve as ações físicas do usuário. Este editor se assemelha a uma planilha eletrônica de edição de hipertexto em cada célula. A proposta original do formalismo *UAN* consiste basicamente em especificar a interface nessas tabelas. Os novos módulos incorporados à ferramenta e as extensões propostas na notação vêm aumentar o seu poder de descrição, complementando as informações dessas tabelas.
- **Editor de Diagramas de Transição de Estados**: responsável pela definição de estados da interface e as transições entre eles.
- **Visualizador de Rede de Tarefas**: possibilita uma visualização da hierarquia existente entre as diversas tarefas na interface.
- **Dicionário**: permite ao usuário registar variáveis, funções e nomes de tarefas utilizadas nas descrições de tabelas *UAN*.
- **Tabela de Definição de Cursores**: Um tipo de *feedback* que deve ser explicitado durante a especificação da interface é aquele referente ao formato do cursor para as diferentes ações do usuário. Nesta tabela, os cursores utilizados nas diversas interações podem ser especificados com o seu significado e formato.

- **Tabela de Mensagens Textuais**: Os identificadores das mensagens de erro ou ajuda utilizados na interação podem ser organizados em tabelas juntamente com o texto de cada mensagem. O sucesso de algumas interfaces deve-se em grande parte à clareza e consistência desse tipo de *feedback* fornecido ao usuário.
- **Módulo de Verificação Sintática**: Em [19] é proposta uma descrição mais rigorosa para a linguagem *UAN*, baseada em três gramáticas *G1*, *G2* e *G3*, associadas, respectivamente, à descrição de ações do usuário, *feedback* da interface e relacionamento temporal entre tarefas. Nesse trabalho, também são feitas verificações sintáticas em descrições *UAN*, construídas a partir dessas três gramáticas. Este módulo, acoplado ao modelo de ferramenta apresentado, auxilia o projetista na obtenção de especificações corretas.

5. IMPLEMENTAÇÃO DO MODELO

Atualmente, encontra-se em desenvolvimento no Departamento de Computação da UFOP, o software *FERA (Ferramenta para Especificação Rápida de Aplicações Interativas)*[4], baseado no modelo proposto na seção 4. Até o momento, foi realizada a análise de requisitos, envolvendo análise de usuário e análise de tarefas e funções, e o projeto conceitual e detalhado, envolvendo projeto de layout de janelas e projeto de mensagens e ícones. A ferramenta está sendo implementada com a utilização da linguagem orientada a objetos *Object Pascal*. A janela principal do aplicativo é composta basicamente por um conjunto de paletas de edição dos diversos componentes de uma especificação, conforme apresentado na figura 3.

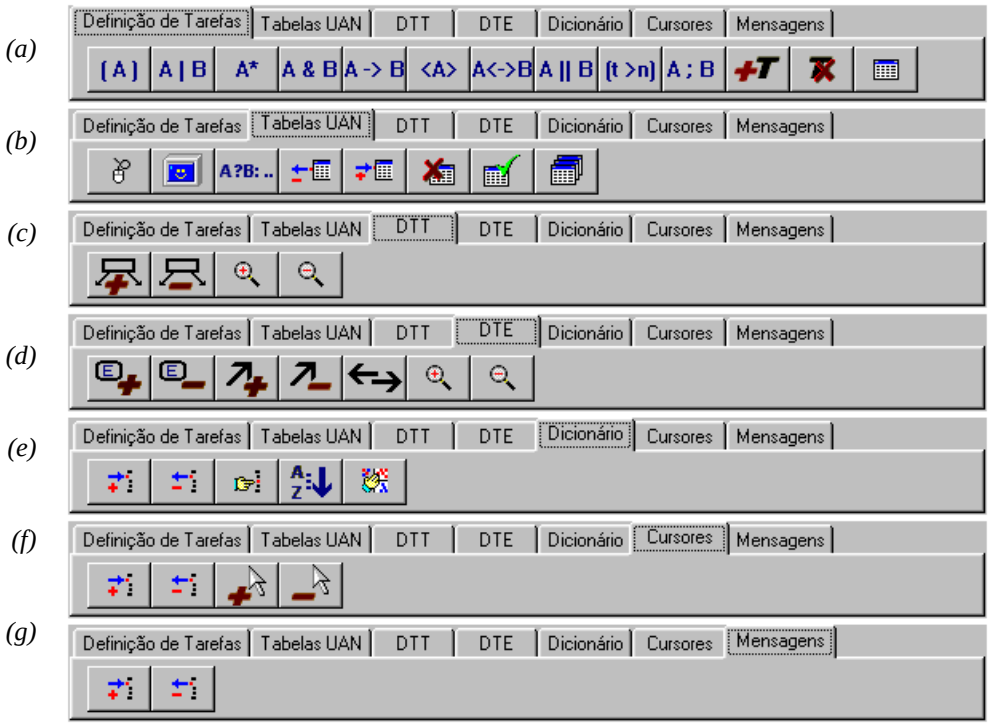

Figura 3: Paletas de Edição para *(a)* Definição de Tarefas, *(b)* Tabelas *UAN*, *(c)* Diagrama de Transição de Tarefas, *(d)* Diagrama de Transição de Estados, *(e)* Dicionário de Variáveis e Funções, *(f)* Tabela de Cursores, *(g)* Mensagens Textuais.

A partir da definição da hierarquia de tarefas de uma determinada aplicação, o projetista pode selecionar tarefas de níveis mais baixos de abstração - que não precisam ser decompostas em outras sub-tarefas - para fazer a descrição da interface no nível articulatório. É apresentada uma tabela de descrição de tarefas, como na figura 4, onde o projetista descreve a interação em termos de ações do usuário, *feedback* da interface, tratamento de exceções, estados da interface e conexões com o componente computacional associado. O Editor de Tabelas *UAN* apresenta itens da notação, habilitados conforme a coluna atual em edição, que auxiliam o projetista durante a especificação de uma tarefa. Durante a especificação no nível articulatório o projetista pode dicionarizar variáveis, funções, mensagens textuais (de erro ou ajuda) e cursores, não necessariamente nessa ordem.

[4] Projeto TEC 81714/95, financiado pela FAPEMIG (Fundação de Amparo à Pesquisa no Estado de Minas Gerais).

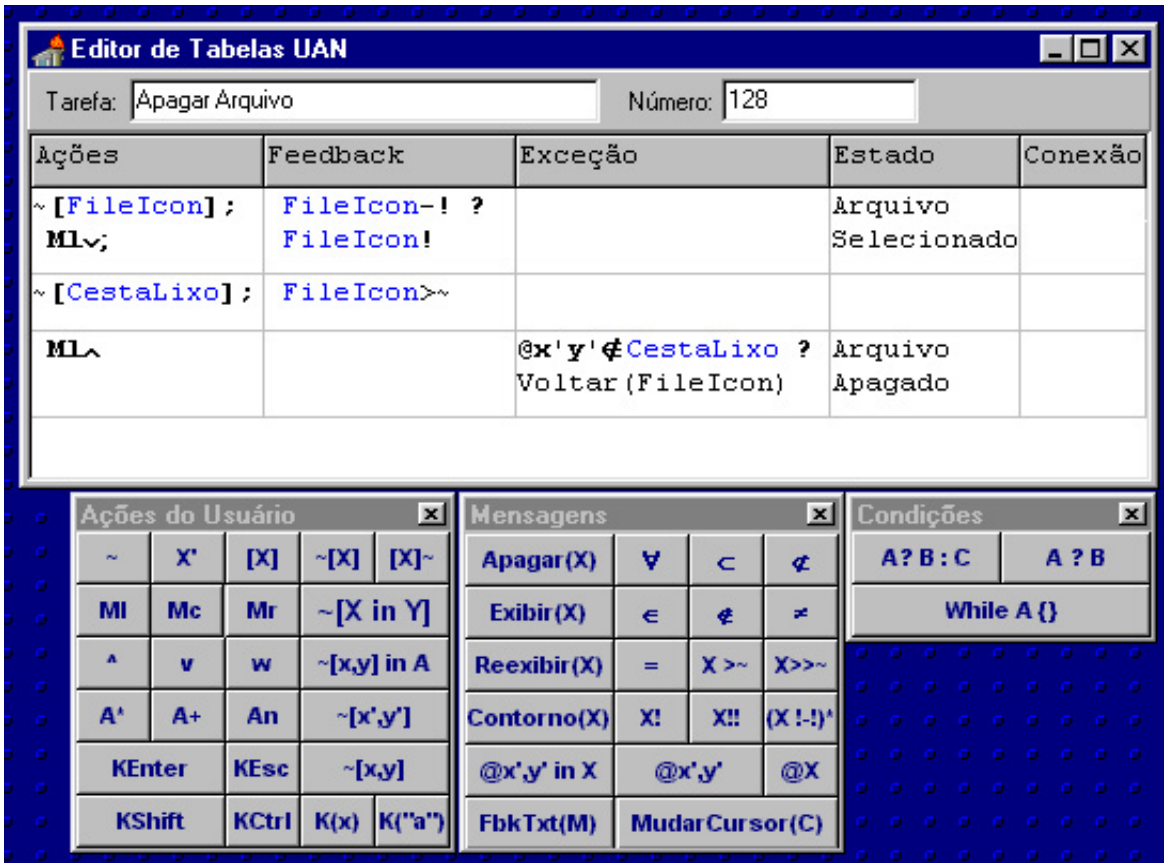

Figura 4: *Editor de Tabela UAN, com um trecho de especificação*

Algumas das principais características da ferramenta são:

- Facilidade de criação e manutenção de vários projetos de interface.
- Especificação visual da hierarquia de tarefas do usuário.
- Fácil manipulação (inserção, eliminação e movimentação) na estrutura da rede de tarefas.
- Manutenção de tabela com sintaxe de descrições básicas em *UAN*, facilitando a especificação de tarefas.
- Manutenção de tabela de referência entre sintaxe e semântica, com exemplos de alguns símbolos *UAN*.
- Edição fácil nas tabelas de especificação *UAN*.
- Verificação sintática das especificações da tabela *UAN* e das definições de tarefas.

No estágio atual de desenvolvimento, em seu processo de avaliação, a ferramenta está sendo utilizada na disciplina *Sistemas Interativos*[5], oferecida pelo Departamento de Computação da UFOP, para a especificação de interfaces de manipulação direta.

6. CONCLUSÕES E PERSPECTIVAS

Neste artigo foi apresentada uma discussão sobre algumas técnicas de especificação de interfaces, principalmente aquelas mais voltadas para a modelagem de diálogos de baixo nível, destacando as principais vantagens e problemas com a utilização de cada uma delas. Também foi apresentada uma ferramenta que permite aos projetistas de interfaces a criação e manipulação de especificações de projeto baseadas no formalismo *UAN* (*User Action Notation*). Essa ferramenta, que incorpora módulos que complementam o modelo básico de descrição em tabelas *UAN*, tem se mostrado eficiente no processo de especificação de diálogos. Os resultados obtidos por equipes que utilizam esse formalismo têm incentivado muitos projetistas de interface. A nossa experiência tem mostrado que a *UAN* é

[5] o programa completo dessa disciplina encontra-se em *http://iceb.ufop.br/~elton/ihc.*

apropriada para especificar o projeto de interação de forma satisfatória, tanto no nível de tarefas quanto no nível de ações físicas do usuário.

As expectativas para o prosseguimento deste trabalho apontam em duas direções principais: a geração automática de código para uma linguagem específica e o suporte ao trabalho cooperativo[6] e ao *design rationale*[7], características importantes em ferramentas que apoiam o trabalho em equipes multidisciplinares. Também como continuação deste trabalho estão previstas sessões de teste e avaliação sobre as dificuldades ou facilidades de uso da ferramenta em situações reais de design.

A atividade de projeto, embora devesse englobar grande parte de um ciclo de desenvolvimento, muitas vezes é deixada para trás. Especificar o projeto de uma interface é uma tarefa difícil, cara e até mesmo enfadonha; mas pode ajudar a equipe de desenvolvimento na obtenção de uma interface final mais próxima do modelo do usuário. A utilização da ferramenta proposta neste artigo vem ajudar a cobrir essa lacuna, criando uma expectativa positiva com relação à produtividade no desenvolvimento, manutenção e evolução de sistemas interativos.

Agradecimentos

Este trabalho foi realizado no Departamento de Computação da UFOP e no Departamento de Computação e Automação da Faculdade de Engenharia Elétrica e de Computação da Unicamp. Agradecimentos especiais ao grande amigo Jesus Cristo, à CAPES, por financiar parte deste trabalho, com bolsa ao primeiro autor, e à FAPEMIG, com bolsas de iniciação científica ao penúltimo e último autores.

REFERÊNCIAS

[1] Card, S. K., Moran, T. P., Newell. A. *The psychology of human-computer interaction*. LEA, 1983.

[2] Coutaz, J. *et al. MATIS: a UAN description and Lesson Learned*. Amodeus Project Document: System Modelling/WP14, Jun. 1993.

[3] Foley, J. D. et al. *Computer Graphics: principles and practice*. Addison-Wesley, 2nd. ed., Jan. 1989.

[4] Gong, R., Kieras D. A validation of the GOMS model methodology in the development of a specialized, commercial, software application. In CHI'94 Conference Proceedings: *Human factors in computing systems*, pp. 351-357, ACM, 1994. Austin.

[5] Green, M. A survey of three dialogue models. *ACM Trans. Graph*., v. 5, n. 3, pp. 244-275, Jul. 1986.

[6] Gugerty, L. The use of analytical models in human-computer-interface design. *Int. J. Man-Machine Studies*, pp. 625-660, 1993.

[7] Harel, D. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, v. 8, n. 3, pp. 231-274, Jun. 1987.

[8] Hartson, H. R., Siochi, A. C., Hix, D. The UAN: A User-Oriented Representation for Direct Manipulation Interface Designs. *ACM Trans. On Information Systems*, v. 8, n. 3, pp. 181-203, Jul. 1990.

[9] Hix, D., Hartson H. R. *Developing user interfaces: ensuring usability through product and process*. John Wiley and Sons, 1993.

[10] Irving, P. P. S., Irving, J. E. A GOMS analysis of the advanced automated cockpit. In CHI'94 Conference Proceedings: *Human factors in computing systems*, pp. 344-350. ACM, 1994. Austin.

[11] Jacob, R. J. K. Using formal specifications in the design of a human-computer interface. In N. Gehani e A.

[6] envolve grupos de pessoas engajadas numa tarefa comum, utilizando uma interface neste ambiente compartilhado.

[7] Em termos gerais, *design rationale* é a descrição do desenvolvimento de um artefato que vai além do registro da sua especificação e teste. Inclui o porquê de determinadas decisões terem sido tomadas durante o projeto, análise de compromisso envolvida entre tomar uma decisão ou outra, motivos que levaram a equipe a incluir determinadas caracterísicas e a abandonar outras, ou quais os erros e enganos cometidos durante o projeto.

D. McGettrik, eds., *Software Specification Techniques*, pp. 209-222. Addison-Wesley, 1986.

[12] Jacob, R. J. K. A specification language for direct-manipulation interfaces. *ACM Trans. Graph*., v. 5, n. 4, pp. 283-317, Oct. 1986.

[13] Jambon, F., Karsenty, L. *Formalisation des Interfaces et Travail Coopératif: Quelles Conséquences?* IHM'94: Sixièmes journées sur l'ingénierie des Interfaces Homme-Machine, Lille, Déc. 1994.

[14] Karat, J., Dayton, T. Taking design seriously: exploring techniques useful in HCI design. *ACM SIGCHI Bulletin*, v. 22, n. 2, pp. 26-33, Oct. 1990.

[15] Kieras, D. E. Towards a pratical GOMS model methodology for user interface design. In M. Helander, ed., *Handbook of Human Computer Interaction*, pp. 135-157. Elsevier, 1988.

[16] Kieras, D. E. GOMS modeling of user interfaces using NGOMSL. In CHI'94 Conference Proceedings: *Human factors in computing systems*, pp. 371-372. ACM, 1994. Austin.

[17] Laurel, B. *The art of human-computer interface design*. Addison-Wesley, 1990.

[18] Lucena, F. N. *Construção de interfaces homem-computador: o uso de estadogramas na especificação e implementação de controle de interface*. Tese de Mestrado, IMECC, Universidade Estadual de Campinas, Mar. 1992. Mestrado em Ciência da Computação.

[19] Mafuz, R. G., Silva, E. J. *Um compilador para a linguagem UAN*. Relatório Técnico 013/96. Departamento de Computação, Universidade Federal de Ouro Preto, Dez. 1996.

[20] Moran, T. The command language grammar: a representation for the user interface of interactive computer systems. *Int. J. Man-Machine Systems*, v. 15, n. 1, pp. 3-51, Jul. 1981.

[21] Payne, S. J. Task-action grammars: a model of the mental representation of task languages. *Human Computer Interaction*, v. 2, pp. 93-133, 1986.

[22] Reisner, P. Formal grammar and human factors design of an interactive graphics system. *IEEE Trans. Soft. Eng*., v. SE-7, n. 2, pp.229-240, Mar. 1981.

[23] Shneiderman, B. *Designing the user interface: strategies for effective human-computer interaction*. Addison-Wesley, 2nd. ed., 1992.

[24] Silva, E. J. *Representação em projeto de interfaces homem-computador: estudo, aplicação e propostas de extensão no formalismo UAN*. Tese de Mestrado. Faculdade de Engenharia Elétrica, Universidade Estadual de Campinas, Jul. 1995.

[25] Siochi, A. C., Hartson, H. R. Task-oriented representation of asynchronous user interfaces. In CHI´89 Conference Proceedings: *Human factors in computers systems*, pp. 183-188. ACM, May 1989, Austin.

[26] Siochi, A. C., Hix, D., Hartson, H. R. The UAN: a notation to support user-centered design of direct manipulation interfaces. In John Karat, Ed., *Taking Software Design Seriously*, pp. 157-194. Academic Press, 1991.