Isa Haro Martins

Um Instrumento de Análise Semiótica para Linguagens Visuais de Interfaces

Tese de Doutorado

Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
Rio de Janeiro, 24 de abril de 1998

Isa Haro Martins

Um Instrumento de Análise Semiótica para Linguagens Visuais de Interfaces

Tese apresentada no Departamento de Informática da Pontifícia Universidade Católica do Rio de Janeiro como parte dos requisitos para obtenção do Título de Doutor em Informática

Orientadora: Clarisse Sieckenius de Souza

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Rio de Janeiro, 24 de abril de 1998

"... A vida é uma flor dourada,
tem raiz na minha mão. ... "
Cora Coralina

Vida Dura ... ou Não

Para defender minha tese tive que ralar, afinal
Tremendo estresse até passei mal
Pesadelo na véspera acordei assustada
A maldita da tese estava toda errada.

Defendi a maldita com a cara e a coragem
Banca de cara feia, mas que sacanagem
Pensei tô perdida o que vou fazer
Se me encherem aviso eu pago prá ver

Logo fui me acalmando
Dominei a platéia e nessa odisséia botei prá quebrar
Mas vejam a minha sina
Uma Pergunta cretina a mim se destina
P'ra me derrubar

Sua tese tá boa cheia de detalhes
Mas defina melhor linguagem visual
Aproveitei respondi também pro Sir Charles
Nesta vida nem tudo tem de ser formal

Agora já sou doutora
Não preciso estudar, posso cozinhar e pegar na vassoura
Esta é minha ventura
Vejam o que me espera a vida é vera uma grande loucura

Mas não vou me dobrar sou intelectual
Se alguém me quiser é de igual para igual
Pode se acostumar e mudar a postura
Acabou a moleza, vai ser vida dura...., ou não.

Este samba foi presente de fim de tese,
dessas pessoa tão queridas.
Autores: Luiz Fernando, Rogério e João Célio

Agradecimentos

- Ao Professor Marcelo Gattass que nos apoiou, fornecendo subsídios para o desenvolvimento de pesquisas envolvendo sistemas gráficos.
- Mais uma vez, à "tribo da Clarisse" pelas discussões, pela participação, pela presença. Em particular à Marisa, uma convidada especial da tribo, autora de uma das perguntas fundamentais, sobre as quais repousa este trabalho. Vocês são ótimos.
- Com carinho especial ao meu parceiro na vida, Luiz Fernando, que com paciência e compreensão sempre deu força, acenando com a luz no fim do túnel todas as vezes em que desanimei.
- Uma lembrança carinhosa ao Rogério e ao João Célio, pela parceria com o LF no samba “Vida Dura … ou Não”, ao pessoal do Telemídia e ao nosso amigo querido, hoje Magnífico, Professor Antonio Mauro, que, a despeito da distância geográfica, esteve presente à defesa deste trabalho. A convivência com vocês é parte do que faz a vida boa.
- Aos colegas, professores e funcionários do Departamento de Informática, pelas boas horas de convivência e pela troca de experiências.
- À Fundação CAPES, pela ajuda financeira recebida durante o curso.
- Meu agradecimento especial à Professora **Clarisse Sieckenius de Souza** pelos anos de trabalho conjunto, pelo rigor teórico que exigiu de nosso trabalho, garantindo a boa formação que hoje nos ajuda a orientar outros alunos, pelo rigor teórico que exige de si mesma, oferecendo a seus alunos a segurança de "estar em boas mãos". Também pelos anos de convivência humana, feitos de alegrias, incentivos, participações e algumas caras feias de parte a parte, de vez em quando. Junto à bagagem de conhecimentos que acumulei nestes anos de trabalho, levo, vaidosa, o nome da minha orientadora como um distintivo de qualidade.

Resumo

Os Sistemas de Modelagem Geométrica migraram de entradas de dados textuais em *batch*, para entradas de dados através de interfaces gráficas por manipulação direta, como consequência da evolução da tecnologia de hardware. Essa migração trouxe vantagens para o usuário e um novo eixo de problemas, relativos à utilização dos recursos visuais como recursos expressivos. Parte dos problemas de desempenho das interfaces por manipulação direta deve-se ao uso assistemático de recursos expressivos na linguagem de interação. Nos Sistemas de Modelagem Geométrica essa questão é particularmente importante, em função de sua complexidade e das restrições expressivas impostas pelo recurso visual. Tais sistemas são sistemas de representação/abstração de objetos do mundo físico no mundo computacional, logo, são sistemas de natureza intrinsecamente semiótica. Visto que são sistemas criadores de signos, eles devem apoiar os usuários nas ações fundamentais relativas a essa tarefa: (a) selecionar e instituir formas significativas (primárias e secundárias) e (b) prover mecanismos sistemáticos para a interpretação de tais formas quando usadas em atos de comunicação/representação.

Este trabalho aborda o problema das linguagens visuais por manipulação direta para Sistemas de Modelagem Geométrica, a partir do paradigma semiótico. Propõe uma análise sistemática sobre a utilização do recurso expressivo visual e demonstra o efeito de sua aplicação em ambientes de especificação de interfaces. Para esse fim são analisados os limites do recurso visual como meio expressivo e sua estruturação em uma linguagem. Com base nas diretrizes teóricas apresentadas, este trabalho propõe uma extensão da TAG (Task-Action Grammars), envolvendo aspectos semióticos da linguagem, denominada Semiotic TAG (STAG). A STAG reflete a sistematização das escolhas expressivas, servindo de apoio aos projetistas na especificação de suas linguagens visuais particulares. Ela opera em conjunto com diretivas de orientação para codificação da linguagem visual (DOCLV) e heurísticas de avaliação da linguagem visual (HALV). É proposto um ambiente para implementação da STAG em conjunto com as DOCLV e as HALV.

Abstract

Geometric Modeling Systems have changed from textual batch data entry styles to direct manipulation graphical interface data entry, as a consequence of advances in hardware technology. This has brought out several advantages to users. However, it has also raised several issues that remain to be resolved, with respect to the use of visual resources as expressive tools. One of the weakest points is the non-systematic use of the expressive resources in interactive language specification. This problem is particularly important in Geometric Modeling Systems, due to the complexity of such systems and the language constraints imposed by visual expression. These systems are representation/abstraction of physical world objects, so, they are intrinsically semiotic systems. Because they are signs generating systems, they have to support users in the fundamental actions related to this task: (a) to select and to establish significant shapes (primary and secondary) and (b) to supply systematic procedures for the interpretation of such shapes when used in communication/representation acts.

This work discusses the use of direct manipulation visual languages in Geometric Modeling Systems through the use of a semiotic paradigm. It proposes a systematic analysis to the use of visual expressive resources and shows the effect of this approach in interface specification environments. The work analyses the limits of the visual resource as an expressive medium and its structuring as a language. Based on the theoretical directives presented, it proposes an extension of TAG (Task-Action Grammars), including semiotic aspects of the language, named Semiotic TAG (STAG). The STAG reflects the systematization of expressive choices, in order to support designers in the specification of these particular visual languages. It acts together with orientation directives for coding the visual language (ODCVL) and with evaluation heuristics of the visual language (EHVL). Finally, an environment for the implementation of STAG together with the ODCVL and EHVL is proposed.

Sumário

Lista de Abreviaturas

Lista de Exemplos

Lista de Figuras

Lista de Resumos

Lista de Tabelas

Este trabalho tem como objeto de análise as linguagens visuais (LV) de interfaces gráficas para Sistemas de Modelagem Geométrica (SMG) que fazem captura de dados através da representação visual do objeto modelado. Nesses sistemas, a linguagem de interação utiliza recursos visuais, para expressão do sistema, e de manipulação direta (MD), para expressão do usuário. Sua especificação pede cuidados específicos, não só pela complexidade inerente ao uso dos recursos visuais, como pelo processo de captura de dados, cujos requisitos permeiam a linguagem de interação. Neste capítulo é apresentada uma visão geral do problema de LVs para interfaces por MD para SMG. É apresentada também a proposta deste trabalho, referente a um tratamento sistemático na especificação dessas linguagens, baseado em uma abordagem semiótica.

1.1. O Problema

Os Sistemas de Modelagem Geométrica (SMG) evoluíram de entradas textuais, em "batch", para entradas de dados através de interfaces gráficas por manipulação direta (MD), como consequência da evolução da tecnologia de hardware. A entrada de dados em "batch" era realizada a partir da edição de um longo arquivo de dados alfanuméricos. A operação de edição, extremamente trabalhosa e sujeita a erros, deveria ser repetida tantas vezes quantas fosse necessário para a criação, correção e ajuste dos dados das diferentes simulações. Esta entrada de dados foi substituída, passando a ser feita através da representação visual do objeto tratado pelo sistema e de operações sobre tal representação. A representação gráfica que o usuário percebe na tela é compatível com o modelo de dados do sistema, apresentando resultantes de vários cálculos simbólicos feitos automaticamente pelo sistema. Esta representação torna menor a distância entre a forma dos dados e aquilo que ela representa, oferecendo ao usuário uma melhor aproximação do objeto real, conforme sugerido pela Figura 1.1.

Segundo Norman (86), a distância a ser ultrapassada na comunicação do usuário com o sistema divide-se em duas etapas: distância semântica e articulatória. Na primeira etapa o usuário traduz sua intenção em uma operação conceitual do sistema, intenção esta formulada sobre a própria interface. Na segunda etapa ele traduz a operação conceitual em uma articulação correspondente a uma ativação do sistema. Portanto, a natureza da representação adotada na interface exerce um papel importante no entendimento do usuário sobre o processo computacional em execução e, consequentemente, na sua interação com o sistema.

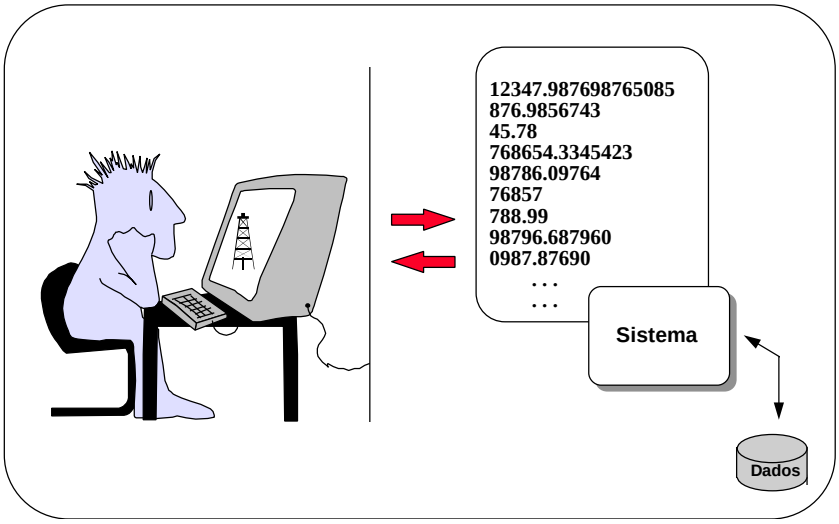


Figura 1.1. Interação usuário-sistema através da representação visual do objeto

Apesar das vantagens deste processo, a interação por MD gera alguns conflitos, cuja origem está ligada, de forma direta ou indireta, à capacidade expressiva dos recursos visuais e, principalmente, à sua utilização de forma sistemática. De modo geral, observamos que grande parte da utilização dos recursos visuais como recurso expressivo é feita de forma assistemática, faltam critérios sólidos para selecionar bons projetos de linguagens visuais no universo dos projetos possíveis, falta instrumental teórico e prático para apoiar o processo de criar uma nova linguagem. A especificação da linguagem visual (LV) é feita a partir de decisões ocasionais, dificultando o processo de compreensão do usuário sobre o conteúdo do sistema.

A utilização sistemática dos recursos de expressão é fundamental para a compreensão de um código entre os agentes da comunicação. No caso dos recursos visuais, a sistematização também é importante para fim de integração e interação com outros recursos expressivos, como por exemplo os textuais. Considerando-se que os recursos visuais são limitados e geralmente insuficientes para a comunicação total de um sistema computacional típico [Foley et alii 93], seu uso quase sempre ocorre num contexto de interfaces multimodais, com a utilização de pelo menos o modo texto e o modo gráfico [Cohen 91, Maybury 93]. Segundo Cohen (91), a integração entre os diferentes modos de interação é fundamental para uma síntese produtiva. Embora limitados, os recursos visuais mostram-se bastante adequados em situações específicas. A integração do código visual com outros códigos permite a exploração adequada de cada um deles como recurso expressivo [Stenning & Inder 95].

Lembramos que os SMG são sistemas de captura de dados, portanto, através deles o usuário estará (a) desenhando um objeto (de forma controlada ou não) e (b) modelando o objeto (de forma controlada), sendo absolutamente necessária a consistência no mapeamento desenho/modelo. O sistema pode permitir ao usuário desenhar, modelar e desenhar modelando. Como será detalhado no tópico 1.1.2, o projetista pode optar por um paradigma de desenho não controlado, estabelecendo um controle posterior, quando então o desenho será validado com relação ao modelo do domínio do sistema. O projetista também pode optar por um paradigma de desenho controlado, no qual o desenho será validado durante sua execução.

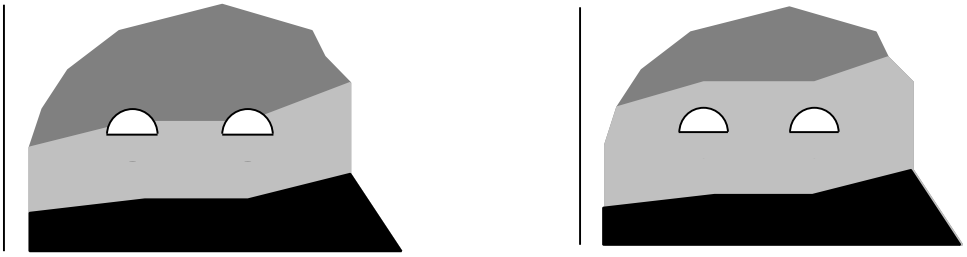
O foco deste trabalho é, portanto, a utilização de LVs para interfaces gráficas por MD em SMG. As LVs devem atender as etapas de desenho e modelagem sucessivas do objeto. O objetivo é subsidiar decisões de projeto sobre LVs destinadas a possibilitar interação usuário-sistema através de MD.

1.1.1. Aspectos da Percepção Visual do Usuário

A seguir, ilustramos uma situação de conflito entre a percepção visual do usuário e a captura de dados do sistema através de um exemplo bastante simples. No exemplo, o usuário deseja executar uma operação mentalmente formulada a partir de sua percepção sobre a interface; mas, a operação é bloqueada pelo sistema, devido ao risco de perda da integridade dos dados correspondentes à representação visual.

Suponha que um engenheiro está usando um pré-processador de elementos finitos para analisar o comportamento de dois túneis a serem construídos em uma rodovia. Na Figura 1.2 apresentamos um desenho, em corte transversal simplificado, de dois túneis passando por um morro com diferentes camadas de solo. A imagem representa esquematicamente a geometria e a topologia do artefato e do seu contorno. Não foi feita neste cenário a geração automática de malha [1]. Para o sistema, a imagem será formalmente descrita em termos de vértices, arestas e faces, a partir das quais o gerador de malhas irá atuar [Gallagher 75, George 91].

[1] Sobre o traçado geométrico do artefato os PPEF geram automaticamente a malha de elementos finitos, que fornecerá elementos para os cálculos de análise e simulação.



(a) (b)

Figura 1.2. Representação visual de dois túneis atravessando um morro

Vamos abstrair as dificuldades iniciais do usuário na construção do primeiro desenho (até porque ele poderia ter sido gerado automaticamente ou importado de outra aplicação) e concentrar-nos na tarefa de edição proposta a seguir. Suponha que o engenheiro perceba que ocorreu um erro, e que a altura da camada do meio é na realidade maior do que aquela representada na Figura 1.2 (a). A representação mais correta da esquematização geral do artefato seria a que aparece na Figura 1.2 (b).

A alteração de 1.2 (a) para 1.2 (b) poderia ser descrita como um ajuste no tamanho da camada do meio, a qual na maioria das interfaces por manipulação direta poderia ser atingida através de uma operação de pressão-e-arrasto (press-and-drag). Esta suposição parte do princípio que os pré-processadores capitalizam o conhecimento já adquirido pelos usuários, permitindo que os mesmos façam uma transferência de conhecimento, resolvendo problemas a partir de experiência adquirida no passado, em situações similares [Booth 89], no caso, a operação de arrasto. O problema, no entanto, é que diferente de muitos desenhos comuns que são tratados apenas como formas compostas e sobrepostas, os desenhos dos pré-processadores são interpretados por analisadores e simuladores. Isto significa que cada vértice, linha e polígono tem um significado implícito, válido em um contexto de modelagem pelo método dos elementos finitos. Assim, no exemplo acima, a região a ser ajustada não é composta por uma linha imaginária que cruza os dois túneis (como na Figura 1.3 (a)), mas por uma linha imaginária que circunda os dois túneis (como na Figura 1.3 (b)), como se tivesse dois dentes na fronteira inferior. Por consequência, o ajuste por pressão-e-arrasto da camada do meio poderia causar uma deformação não desejada nos túneis, como mostra a Figura 1.3 (c).

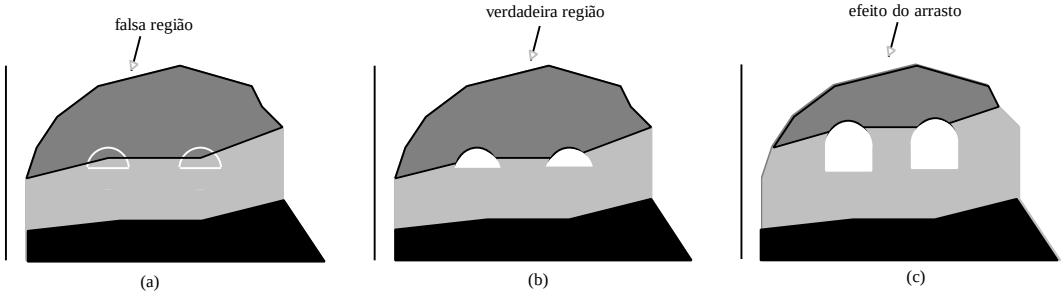


Figura 1.3. Efeitos de manipulação da camada do meio

A consistência da percepção visual do usuário é atingida quando ele pode manipular "intuitivamente" a representação visual. Ela significa a consistência entre o significado pretendido (pelo projetista) e o significado percebido (pelo usuário). Na manipulação intuitiva as inferências e expectativas sobre transformações na forma/estado da imagem correspondem às consequências semânticas operadas pelo sistema em sequência a tais transformações efetivas [Jackendoff 83/87; Souza 92/93b; Ware 93]. De acordo com Norman (86), a intenção do usuário é formulada sobre a própria interface do sistema. Portanto, se a interface desperta uma certa intenção no usuário, mas bloqueia uma manipulação intuitiva de um objeto da interface, dá-se uma quebra de consistência da percepção visual.

Nos SMG, a absoluta necessidade de se manter a consistência semântica para a série de cálculos a serem feitos nas várias simulações gera restrições críticas de manipulação ao nível da interface. Nem sempre similaridades perceptuais do artefato correspondem a similaridades estruturais (menos ainda a similaridades semânticas). Ao tentar manipular a representação visual intuitivamente, transferindo conhecimento adquirido em situação similar (como, por exemplo, a edição gráfica em modo WYSIWYG), o usuário frequentemente se defronta com um conflito.

Em resumo, a representação gráfica que o usuário percebe e interpreta é a resultante de dois processos automáticos: (a) interpretação das variáveis de entrada, (b) geração de modelos gráficos segundo regras específicas. Modificações sobre a representação gráfica, segundo os modelos adotados, devem significar modificações sobre as variáveis de entrada (e não sobre as regras de geração de modelos). Em alguns casos, para não quebrar a consistência da percepção visual do usuário, a interface deveria ser capaz de interpretar alterações referentes às variáveis de entrada e às regras de geração dos modelos gráficos. Por exemplo, a possibilidade de aumentar a altura de um objeto através do mero arrastar no sentido vertical pode ser uma interpretação do usuário sem correspondência com previsões/interpretações do projetista. Se o sistema se restringe a aceitar alterações referentes às variáveis de entrada, este limite deve ser adequadamente sinalizado para o usuário.

1.1.2. Aspectos Característicos dos SMG

No que se refere às LVs em questão, o primeiro aspecto de interesse nos SMG ocorre no contexto da negociação entre a flexibilidade de manipulação e a consistência semântica dos dados. Na interação com estes sistemas, o usuário pode atuar de duas formas: (a) através de uma manipulação de dados orientada à REPRESENTAÇÃO, ou (b) através de uma manipulação de dados orientada à CONSTRUÇÃO.

A MD orientada à representação é aquela que favorece e prioriza operações que imprimem modificações à forma (ou desenho) do objeto, com autonomia (ainda que temporária) em relação àquilo que a forma quer dizer. Um exemplo claro e simples pode ser obtido com um editor gráfico no qual o usuário constrói o modelo (em corte frontal) de um galpão, ver Figura 1.4 (a). Uma orientação que autonomiza forma de conteúdo permite que a forma seja trabalhada independentemente do conteúdo até estabilizar-se a contento do usuário que a cria. Assim, o galpão da Figura 1.4 (a) poderia ser construído pelo caminho 1.4 (b) em que a cobertura do galpão é representada ("colocada") no desenho antes das vigas que a sustentam.

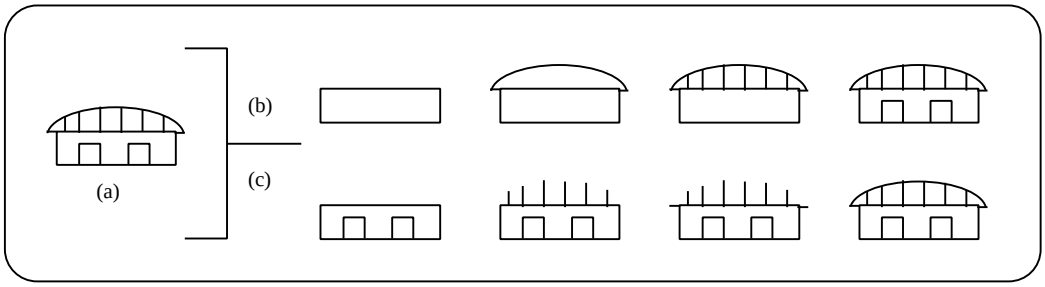


Figura 1.4. Sequência de estágios no processo de edição da representação gráfica de um objeto

A MD orientada à construção é aquela que favorece e prioriza a viabilidade física do processo de construção, tomando operações de "desenho" como operações de "criação" do objeto. Assim, no exemplo acima, seria impossível sustentar a cobertura sem a prévia colocação das vigas. Neste caso, o caminho correto de modelagem seria 1.4 (c).

Em um paralelo com aspectos sintáticos e semânticos de uma linguagem, chamamos ao primeiro caso de MD de base sintática - as operações de arranjo e estruturação são mais fáceis e flexíveis para o usuário. Ao segundo caso chamamos de MD de base semântica - as operações de verificação de consistência na interpretação do modelo que o desenho representa estão sempre ativas, imperando sobre (e inibindo) pequenos arranjos práticos de forma desconectados de uma intenção fisica.

Na manipulação orientada à representação, porque não existe interpretação semântica simultânea ao traçado, a ocorrência de inconsistências nas etapas intermediárias do traçado não é detectada pelo sistema e sua correção depende do usuário. Em uma manipulação orientada à construção a edição é menos flexível, mas a consistência dos dados é garantida pelo sistema, não dependendo do usuário.

Em alguns SMG é adotado o paradigma retratado no primeiro caso. Eventualmente os usuários encontram dificuldades em validar representações visuais já executadas. Em outros é adotado o segundo paradigma, com o risco de uma maior inflexibilidade na manipulação da representação visual. Se a facilidade de desenho é o objetivo principal, a interface deve permitir ao usuário editar livremente os contornos visuais do artefato, com o risco de o gerador de malhas não conseguir operar subsequentemente em função de alguma inconsistência introduzida nos dados originais. Se a consistência semântica é o objetivo mais importante e todo desenho deve ser interpretável em termos de um modelo de malha, simples correções nos componentes da representação visual podem se tornar extremamente difíceis de serem atingidas. Alguns projetistas tentam flexibilizar esta manipulação através de soluções localizadas, resultando em soluções assistemáticas, nas quais o sistema oferece comportamentos diferentes para situações similares, causando uma ruptura no processo de compreensão do usuário. Em Martins et alii (95) é apresentada uma proposta de flexibilização da manipulação com base em heurísticas, buscando uma solução sistemática.

O segundo aspecto de interesse nos SMG refere-se ao próprio modelo semântico ativo. Por exemplo, os pré-processadores de elementos finitos têm como característica bastante peculiar a coexistência de dois modelos semânticos atuando sobre a mesma representação visual. Isto significa que uma mesma representação visual está sujeita a diferentes interpretações e que a resposta do sistema às manipulações do usuário estarão de acordo com o modelo de interpretação ativo. Durante a definição da geometria do artefato o traçado é interpretado de acordo com um modelo de engenharia, que busca capturar a geometria e topologia do objeto. Sobre o traçado geométrico o sistema traça a malha de elementos finitos, que pode ser editada pelo usuário. Durante a manipulação da malha o traçado é interpretado de acordo com um modelo de elementos finitos. Observe-se que mesmo após o traçado da malha o usuário pode alternar de um modelo semântico para outro, voltando a editar a geometria. Algumas edições sobre a geometria do objeto podem invalidar a malha traçada, neste caso uma nova malha será traçada pelo sistema.

A Figura 1.5 apresenta um exemplo gerado através do aplicativo Mtool [Mtool 97]. Esta figura representa o dimensionamento para um poço, que passa através de diferentes camadas de solo. Na Figura 1.5(a) a representação

está sendo tratada de acordo com o modelo de engenharia, que permite editar a geometria do objeto. Na Figura 1.5(b) a mesma representação está sendo tratada de acordo com o modelo de elementos finitos, que permite editar a malha de elementos finitos traçada sobre a geometria do objeto.
A interface deveria permitir ao usuário alternar sua atuação entre os diferentes tipos de interação com uma LV consistente [Martins, Souza & Gattasss 94, 95], ou seja, expressando claramente as alternações e o comportamento do sistema em cada caso. Esta não é uma meta fácil de ser atingida. O conhecimento da capacidade expressiva da linguagem e sua adequada utilização são fundamentais na execução deste objetivo.

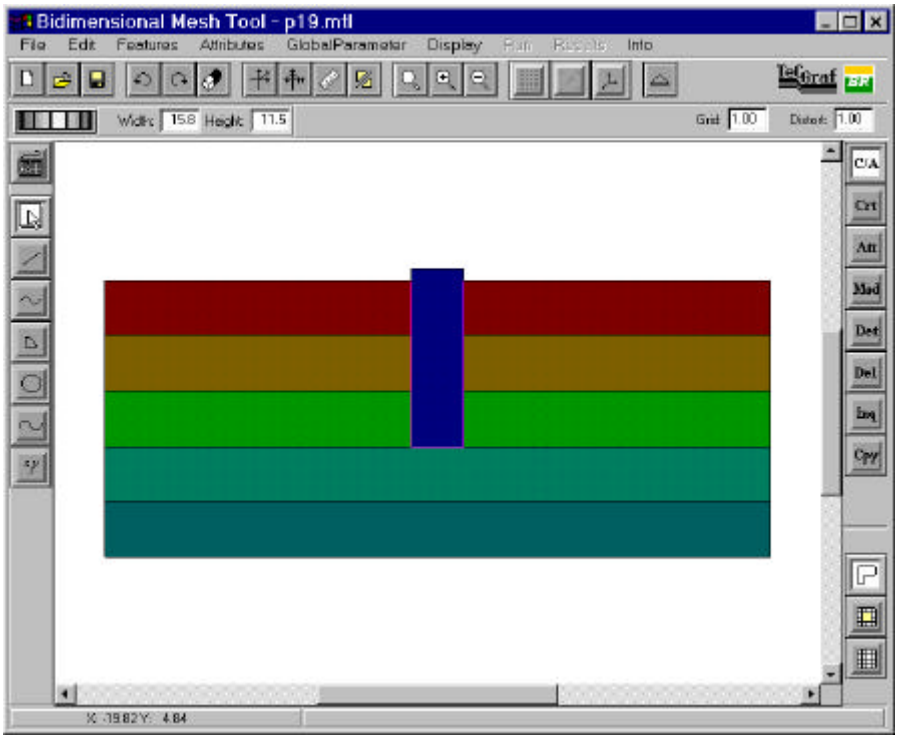


Figura 1.5 (a) Representação gráfica do objeto, com base em um modelo de engenharia

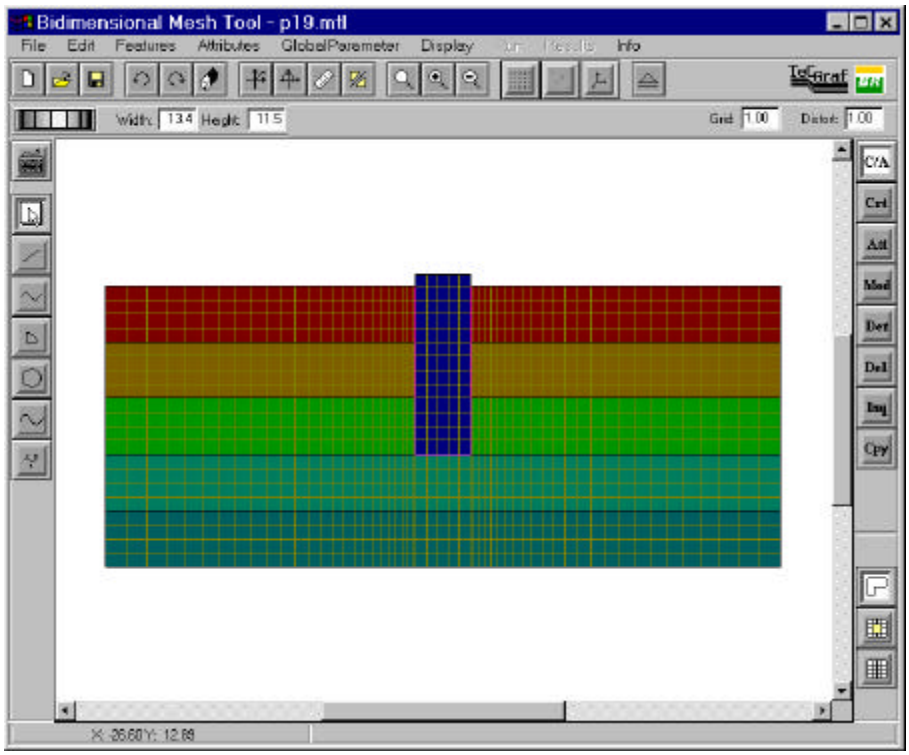


Figura 1.5 (b) Representação gráfica do objeto, com base em um modelo de elementos finitos

1.2. Proposta

1.2.1. Justificativa do Trabalho

Em se mantendo a tendência evolutiva da tecnologia de hardware, as linguagens visuais tendem a se tornar cada vez mais complexas e cada vez mais utilizadas, devido à evolução dos dispositivos de entrada (recursos de manipulação) e dos dispositivos de saída (recursos visuais). Por exemplo, mouses com novas funções têm sido colocados no mercado comercial. Em uma análise de campo, junto a equipes de desenvolvimento de aplicativos gráficos (ITS - Instituto de Tecnologia de Software, Puc-Rio), foi constatada a dificuldade dos projetistas em atender aos objetivos impostos pela complexidade da linguagem. Diante destes fatos, e diante da constatação de uso desordenado dos recursos visuais nos projetos de interfaces, diretrizes de orientação tornam-se cada vez mais necessárias.

Em termos de futuro tecnológico, um pequeno parênteses deve ser feito para um comentário sobre a provável evolução dos sistemas gráficos no sentido da realidade virtual. A realidade virtual tem tipicamente tentado mimetizar sensações físicas reais (daí o nome realidade virtual). Os SMG referem-se à realidade, mas efetuam computações de natureza abstrata e hipotética. A eventual aplicação da realidade virtual a estes sistemas trará novas formas de interação, mas não alterará sua natureza computacional. Isto significa que a realidade virtual não prescindirá da LV, ao contrário, ela envolverá a LV, acrescentando a ela novos aspectos de interação. Portanto, mesmo a evolução dos SMG no sentido da realidade virtual não invalida a busca de tratamento sistemático das LVs proposta neste trabalho, ao contrário reforça essa necessidade.

A disponibilidade de uma ferramenta que permita ao projetista avaliar a qualidade do sistema de comunicação que ele está criando para dialogar com o usuário oferece vantagens em vários aspectos, de método, de qualidade e de custo. A consciência dos problemas inerentes à LV, durante o projeto, permite que todo potencial do recurso seja explorado e que suas restrições sejam devidamente complementadas através do uso de outros recursos. Uma ferramenta de auxílio ao projeto da LV pode permitir que a medida de qualidade da interface venha à tona sem um esforço extra do projetista.

A metodologia auxilia ainda que as interfaces não sejam projetadas apenas para o consumo imediato. Tendo conhecimento claro dos recursos disponíveis, o projetista consegue prever melhor os problemas e projetar soluções mais seguras e mais abrangentes. Interfaces projetadas de forma sistemática, oferecem suporte a expansões futuras e quando necessário permitem correções controladas, o que não se verifica em projetos com base em soluções ocasionais.

O desenvolvimento de projetos sistemáticos tende a criar uma nova cultura entre os projetistas, substituindo gradativamente a utilização ocasional dos recursos visuais por uma utilização mais sólida. Frequentemente ocorre que soluções inadequadas adotadas em aplicativos comerciais de grande divulgação no mercado se perpetuam,

transformando-se em cultura na área de aplicativos gráficos. Existe um custo implícito para que uma solução inadequada em termos de linguagem seja absorvida pelo usuário. Mas, uma vez que uma solução tenha se estabelecido, os projetistas podem encontrar resistência de mercado em alterá-las. Pode ocorrer de os projetistas encontrarem sérias dificuldades em criar um projeto novo, consistente como um todo, acabando por definí-lo sobre os vícios de projetos anteriores. O estímulo a uma mudança de postura dos projetistas de LVs pode render bons frutos. A sistematização do processo de desenvolvimento de LVs é fundamental em termos de trabalhos futuros. Ela é o primeiro passo para qualquer tratamento mais complexo em ambiente computacional. Quanto antes um erro aparece, menos ele custa. Portanto, ferramentas analíticas como a aqui proposta podem auxiliar o desenvolvimento de projetos mais baratos, uma vez que, com base nas diretrizes de orientação, o projetista trabalha de forma mais segura, com a possibilidade de um resultado melhor e de facilidade em manutenções futuras [Berry 97].

1.2.2. Os Objetivos do Trabalho

Os SMG são sistemas de representação/abstração de objetos do mundo físico no mundo computacional, logo, são genuinamente sistemas de natureza semiótica[2]. Visto que são sistemas de representação/abstração, isto é, criadores de signos, tais sistemas devem apoiar os usuários nas ações fundamentais aí envolvidas: (a) selecionar e instituir formas significativas (primárias e secundárias) e (b) prover mecanismos sistemáticos para a interpretação de tais formas quando usadas em atos de comunicação/representação. Isto significa que os SMG devem oferecer aos usuários meios de interação de forma que a tarefa de representação/abstração a ser cumprida no mundo computacional possa ser realizada a bom termo.
Este trabalho aborda o problema das linguagens visuais por manipulação direta para Sistemas de Modelagem Geométrica, a partir do paradigma semiótico. Um de seus pilares de apoio é uma análise dos recursos expressivos no ambiente computacional, segmentando-os e estruturando-os com características de código. Nesta análise são discutidos os limites do recurso visual como meio expressivo e sua estruturação em uma linguagem. O objetivo do trabalho é propor um tratamento sistemático na utilização do recurso visual e demonstrar o efeito deste tratamento na especificação de LVs de interfaces.

1.3. Hipótese de Trabalho

O desempenho dos usuários na utilização de um aplicativo é um elemento de avaliação de sua usabilidade[3]. Em Booth (89) observamos que o conceito de usabilidade abrange diferentes aspectos da interação. Em uma abordagem genérica, alguns pontos podem ser destacados como importantes para a usabilidade. São eles: a capacidade de o usuário aprender sobre o sistema, a capacidade de o usuário atingir seus objetivos de forma eficiente, a capacidade de o usuário transferir para o aplicativo em uso o conhecimento adquirido em situações similares [Booth 89]. O conceito de eficiência envolve não apenas o conhecimento da tarefa, mas a capacidade de ativar o sistema para executá-la, com um mínimo de erros. De acordo com esta abordagem o projetista lança mão de avaliações sobre o desempenho do usuário, buscando produzir interfaces que garantam a eficiência.
Em Adler & Winograd (92) é apresentada uma nova abordagem de usabilidade, na qual o usuário é tratado como um componente ativo capaz de entender o sistema, aprender sobre ele e utilizá-lo de formas criativas. "*The key criterion of a system's usability is the extent to which it supports the potential for people who work with it to understand it, to learn, and to make changes.*" [Adler & Winograd 92, p. 7]. Nesta abordagem, ao invés de o projeto de interfaces estar orientado na busca de elementos de interface capazes de favorecer um melhor desempenho do usuário, ele estará orientado na busca da melhor forma possível de expressar o modelo conceitual do sistema, potencializando ao usuário utilizá-lo plenamente. O foco de preocupação desloca-se de "como o usuário vai usar os recursos do sistema" para "como o projetista pode expressar melhor ao usuário o que o sistema faz". Este novo enfoque desloca o projeto de um nível mais superficial de uso dos recursos de interface, para um nível mais profundo de uso de uma linguagem expressiva, que possibilite a comunicação do potencial de operação do sistema. Obviamente o nível mais profundo de análise não elimina o mais superficial, apenas vai exigir que os elementos de interface sejam tratados com uma estrutura mais profunda, com características de linguagem.

---

[2]Semiótica é a disciplina que estuda os signos e os sistemas sígnicos. Signo é qualquer coisa que representa (isto é, que é interpretável como) algo para alguém [Peirce 31, Eco 76, Sebook 95]. No campo de interfaces de sistemas, tudo são signos, e, portanto se presta a um tratamento semiótico.

[3]Estaremos usando a formação "usabilidade" a partir da palavra "usável", como tradução do termo inglês 'usability".

1.3.1. A Interface como um Artefato de Metacomunicação

Observando a interface sob a perspectiva de *media* apresentada por Kammersgaard (88), podemos avaliá-la como um elemento dentro um processo de comunicação. Neste processo, o usuário não está se comunicando com um artefato computacional, mas com outro agente humano, no caso o projetista, através do sistema. De acordo com os fundamentos da Engenharia Semiótica postulados por Souza (93b), dois níveis de comunicação podem ser identificados neste processo. No nível 1, a interface é um meio de expressão unidirecional através do qual são enviadas mensagens do projetista para o usuário. " *The mediation is such that the user cannot send back a response to the designer, except perhaps in test, maintenance or evaluation circunstances...*" [Souza 93b, p. 756]. No nível 2, ocorre a troca de mensagens entre o usuário e a própria interface. Este nível caracteriza a interface como um artefato de metacomunicação [Souza 93b], uma vez que ela própria será um emissor e receptor de mensagens. A Figura 1.6 ilustra esta situação.

Por exemplo, comparando as interfaces dos editores gráficos CorelDraw [CorelDraw 5.0] e Paint [Windows 95], Figuras 1.7 (a) e (b) respectivamente, observamos diferenças substanciais nas mensagens do projetista para o usuário - nível 1 de comunicação. Por exemplo, no CorelDraw a posição do cursor é ininterruptamente assinalada através de duas réguas, uma vertical e uma horizontal, além da identificação textual das coordenadas correntes, apresentada na barra de mensagens. Também são assinaladas na barra de mensagens as medidas em polegadas das primitivas traçadas, como por exemplo um polígono, sempre que esta primitiva estiver selecionada. A mensagem do projetista, através da interface, é de que este editor gráfico dirige-se a aplicações que necessitam precisão. Ele oferece recursos de medição dos objetos traçados. No Paint não observamos nenhum recurso de medição, mas no menu de ícones (à esquerda da tela) observamos pelo menos três recursos diferentes de traçado: o traço fino (lápis), o traço grosso (pincel) e o traço difuso (aerosol). Através destes recursos a mensagem do projetista para o usuário é de que este editor dirige-se a aplicações voltadas, não para a precisão, mas para a criatividade. A mensagem do projetista informa o propósito geral do sistema, não impedindo que o usuário faça usos alternativos do mesmo.

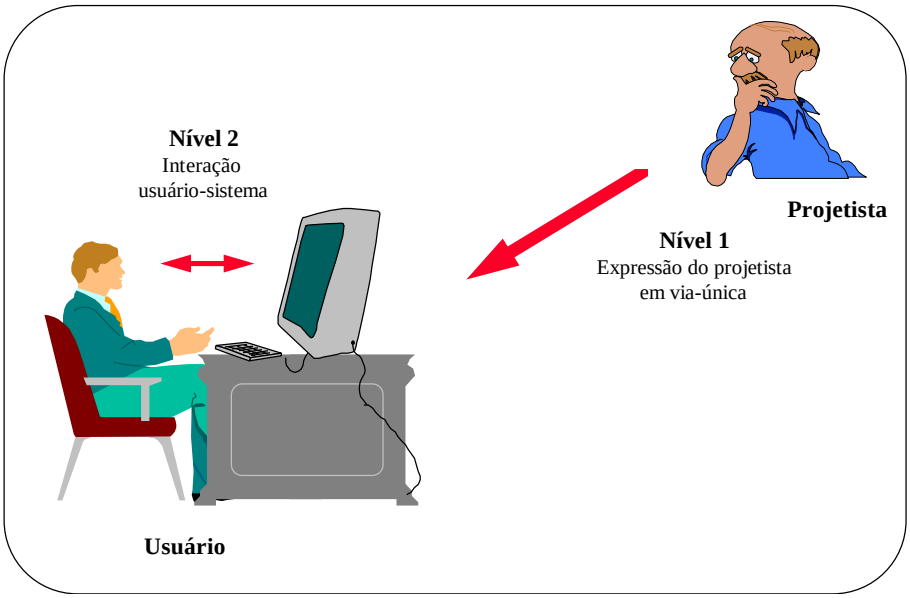


Figura 1.6. Interface como meio de comunicação entre o projetista e o usuário

Um segundo exemplo pode ser observado, envolvendo a especificação de objetos para manipulação. Por exemplo, no CorelDraw se o usuário deseja selecionar um conjunto de primitivas para manipulação ele traça uma janela delimitando um espaço na tela. As primitivas que estiverem dentro da janela são selecionadas, as que estiverem fora ou forem seccionadas pela janela não serão selecionadas. O sistema expressa a seleção através de formas auxiliares (■), indicando pontos de manipulação em torno do conjunto de primitivas (Figura 1.8 (a)). As manipulações executadas sobre esses pontos são aplicadas para todo o grupo. No Paint, o mesmo procedimento oferece um resultado diferente. A janela não se apaga e as primitivas seccionadas são recortadas. A área delimitada pela janela pode ser manipulada conforme ilustrado na Figura 1.8 (b).

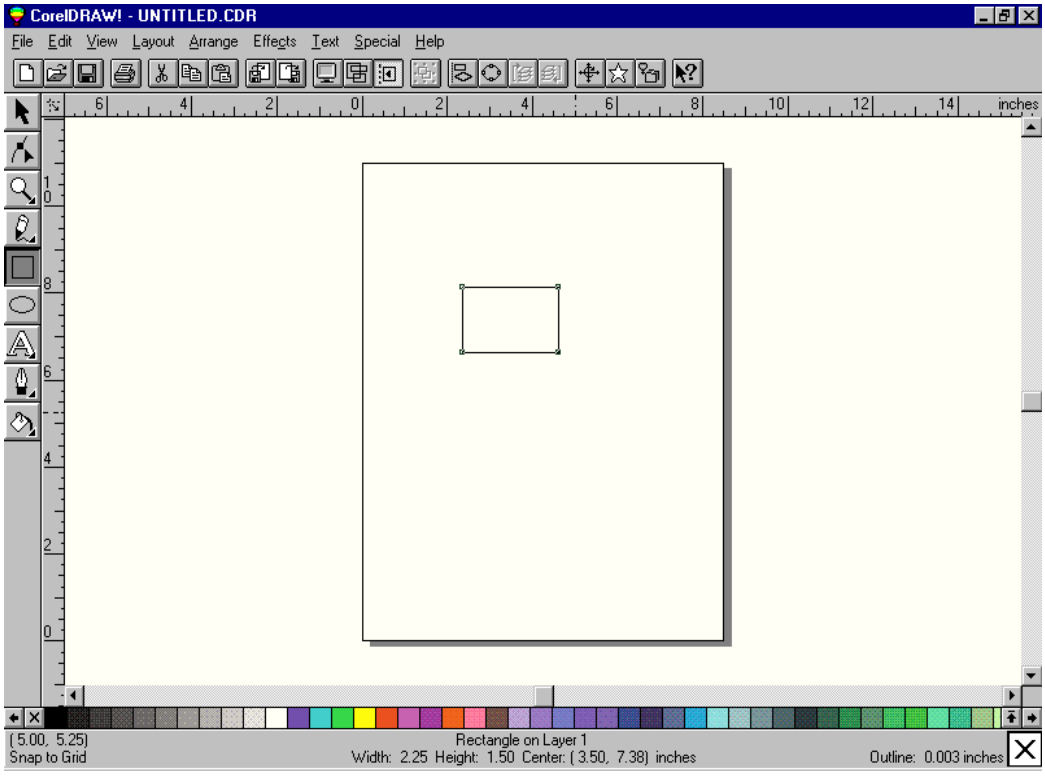


Figura 1.7 (a) Interface do CorelDraw

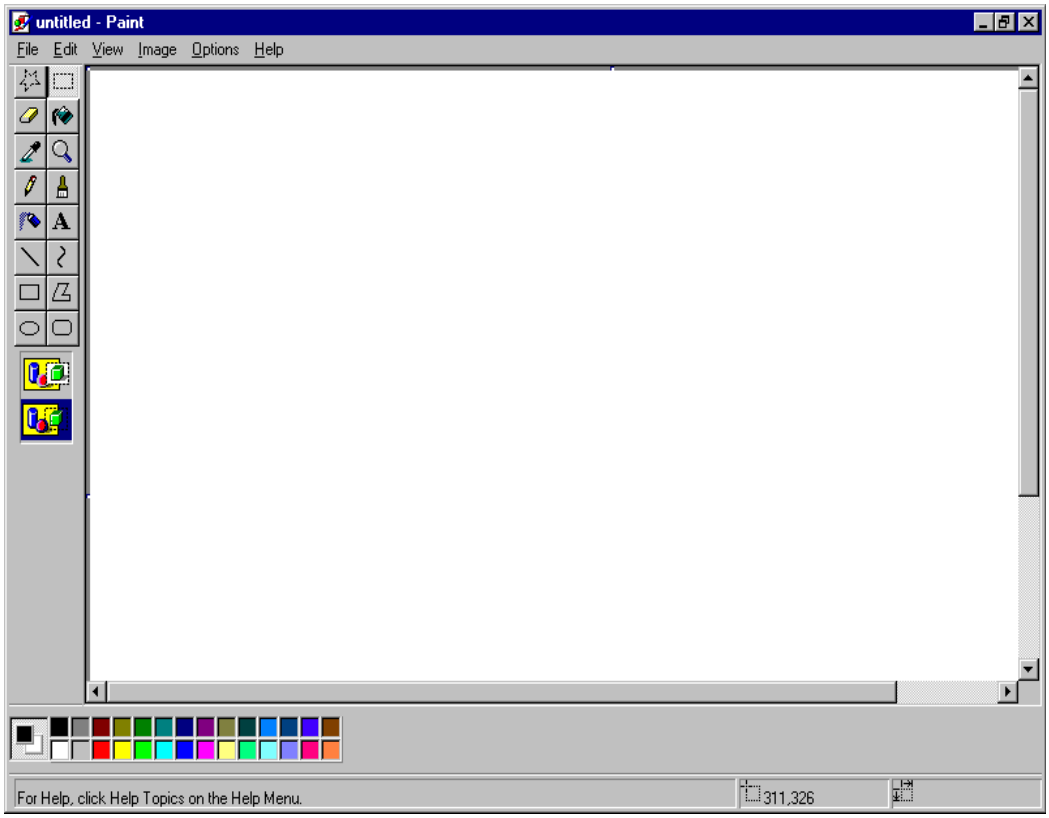


Figura 1.7 (b) Interface do Paint

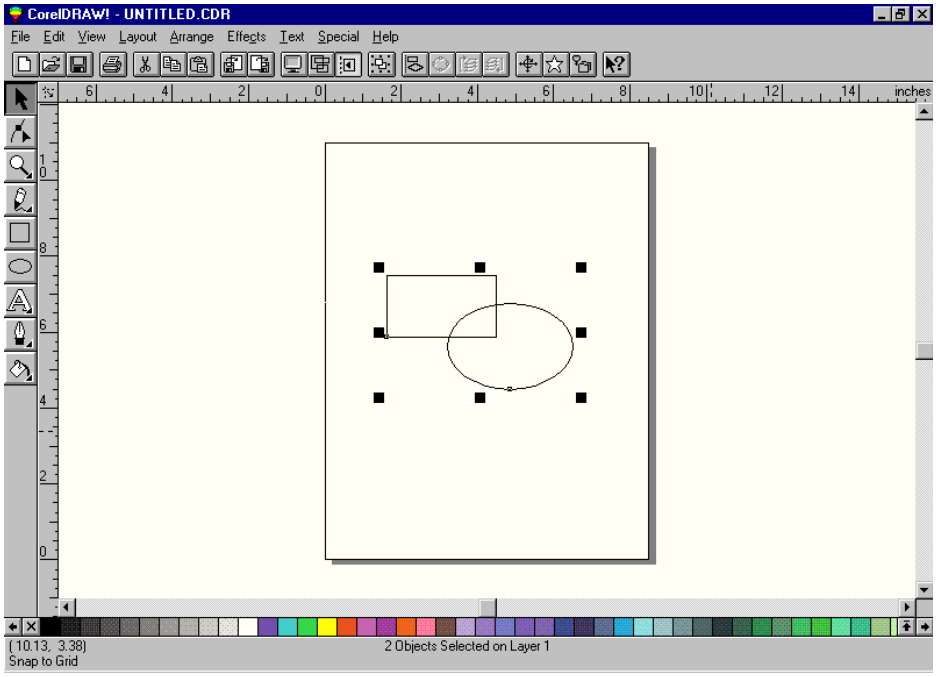


Figura 1.8. (a) Seleção de objetos no CorelDraw

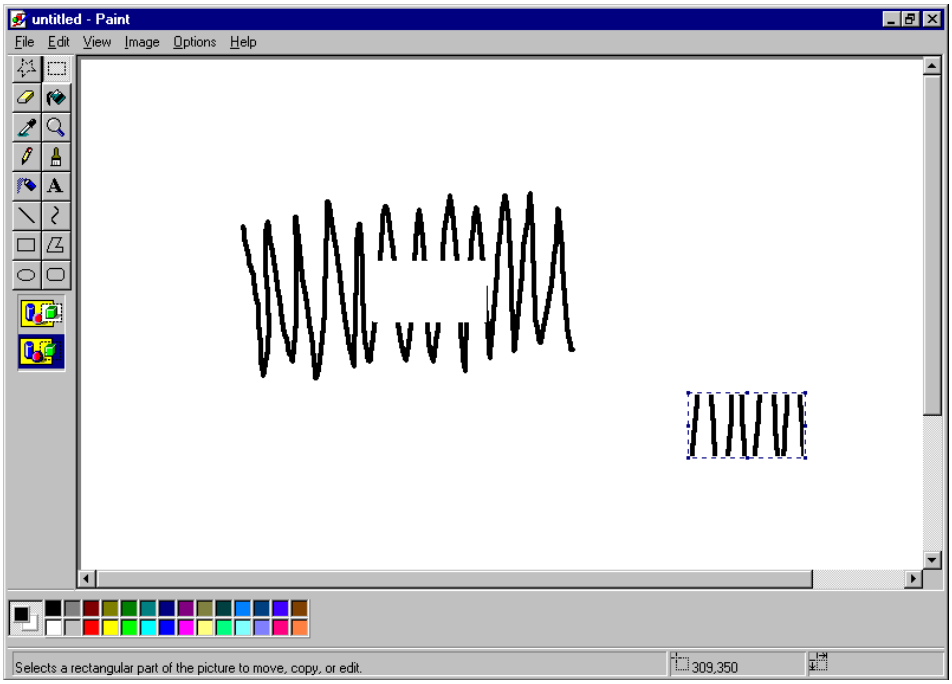


Figura 1.8. (b) Seleção de objetos no Paint

Através deste segundo exemplo, podemos facilmente observar os dois níveis de comunicação. No nível 1, a mensagem do projetista informando o modelo conceitual do sistema. No caso do CorelDraw, o sistema mantém pontos de manipulação em torno do conjunto de primitivas, informando que elas serão tratadas como um grupo. No caso do Paint, a janela permanece, indicando que os elementos traçados não serão tratados individualmente, o espaço delimitado pela janela é tratado como uma área, como se fosse parte de uma tela ou fotografia. No nível 2, a mensagem do usuário especificando um objeto para manipulação e o retorno do sistema sinalizando o objeto especificado (pontos de manipulação no CorelDraw e janela tracejada no Paint). Em ambos os casos, a expressividade da linguagem de interface é fundamental para a codificação adequada da mensagem.

1.3.2. A Abordagem Semiótica ao Projeto Linguístico

"*Compared to more analytical theories, Eco's TSP (Theory of Sign Production) grants us a synthesis-oriented perspective which is paramount to all design activities, and allows us to sketch the basis of a theoretic approach to user interface language design (UILD). ... We show that semiotic principles do not contradict cognitive ones; quite contrarily, both indicate the same direction to be followed. Insights and contributions, however, are of a different nature; whereas Cognitive Science presents designers with the appropriate targets to be hit, Semiotics presents valuable guidance for making successful shots.*" [Souza 93, p. 754].

A semiótica apresenta uma revisão do conceito de significação, no qual o processo de significação é dividido em significado semântico e pragmático. Semântica, para certas correntes, é o tratamento de relações de significado em um mundo abstrato ideal, associando-se elementos de representação a conceitos abstratos ou estados do mundo [Lyons 77]. Pragmática, para outras correntes, é o tratamento das condições intencionais ou circunstanciais outras, que dão sentido prático real e situado às idealizações determinadas pela semântica [Austin 62, Searle 69/75a/75b].

O significado pragmático modula o significado semântico. A divisão é confortável do ponto de vista didático, porque em certas circunstâncias permite a redução do problema aos aspectos semânticos ignorando os aspectos pragmáticos. Em ambiente computacional, o elenco de variações intencionais e circunstanciais aplicável a apenas um dos parceiros da interação direta (o humano) cai no vazio da falta de paralelo para estas mesmas variações no outro (o computador) [Martins 90]. O comportamento deste último é sempre determinado por regras claras e inexoráveis que são insuficientes para tratar com adequação o universo em aberto de situações inéditas a que um ser humano está exposto. Porém, no processo de interação usuário-sistema, os aspectos pragmáticos não podem ser eliminados embora sejam reduzidos.

Através do conceito de signo, a semiótica apresenta uma abordagem específica para o processo de significação, na qual o processo de significação é dinâmico. "*A sign is everything which can be taken as significantly substituting for something else*" [Eco 76, p. 7]. Aparentemente este conceito é bastante semelhante ao conceito de significado semântico. Porém a instância de significação não se restringe apenas à ocorrência do elemento significante e do referenciado, ela exige a presença do elemento humano. Desde Peirce (31) o processo de significação vem sendo apontado como uma tríade. Segundo ele, o signo coenvolve uma cooperação de três sujeitos: o elemento de conteúdo, o elemento de expressão e o interpretante, ocorrendo uma influência tri-relativa não passível de resolução em uma ação de duplas [Peirce 31]. Existe uma correlação entre os elementos da expressão e os elementos do conteúdo definida por um código, mas o signo não é apenas o produto desta correlação, ele é o produto dinâmico desta correlação na presença de um intérprete.

Segundo Eco (76), desta posição decorrem algumas consequências: (a) o signo não é uma entidade física, a entidade física é no máximo o elemento da expressão; (b) como o signo só ocorre na presença do intérprete humano, sem ele não existe um processo de significação; e (c) o signo não é uma entidade semiótica fixa, a idéia a que o signo dá origem é chamada seu interpretante.

Segundo Peirce, o interpretante é aquilo que o signo produz na mente do intérprete. O interpretante pode ser concebido como uma outra representação referida ao mesmo objeto. Ele pode assumir várias formas, por exemplo, uma associação emotiva que adquire o valor de conotação fixa, como cão por fidelidade, e vice- versa, é um interpretante. Ou ainda, uma definição científica em termos do próprio sistema semiótico, como cloreto de sódio por sal, também é um interpretante. "... *in order to establish what the interpretant of a sign is, it is necessary to name it by means of another sign which in turn has another interpretant to be named by another sign and so on.*", num processo de semiose ilimitada [Eco 76, p. 68].

Na língua em uso, ou seja, no processo de comunicação, cada agente tem seu próprio interpretante sobre o que está sendo comunicado. Ao dialogar em função do seu próprio interpretante, cada um dos agentes estará dando origem a um novo interpretante, num processo de semiose ilimitada. Os incontáveis interpretantes possíveis de um determinado conteúdo podem ser tidos como representações referidas ao mesmo objeto. Em um diálogo não existem garantias de que o interpretante do agente receptor será o mesmo do agente emissor. Um objetivo a ser perseguido por uma boa linguagem de comunicação é que a distância entre os interpretantes de cada agente seja a menor possível. Quanto mais próximo o interpretante do agente receptor estiver do interpretante do agente emissor, maior será a chance de que a mensagem seja bem compreendida. O processo de comunicação é uma negociação constante entre os interpretantes dos agentes.

Na LV a correlação entre elementos do conteúdo e elementos da expressão é estabelecida pelo projetista na especificação da sua linguagem particular. Por exemplo, quando o projetista associa determinadas propriedades a texturas, ele está estabelecendo uma correlação entre um tipo de conteúdo e um recurso expressivo. A forma de expressão adotada é o seu interpretante pessoal e circunstancial. Pelo lado do agente computacional, a semiose é interrrompida quando o interpretante do projetista é fixado na implementação do sistema. A LV de interface corresponde ao signo visual do projetista.

Pelo lado do agente humano ocorre a semiose ilimitada. Cada usuário tem seus próprios interpretantes sobre o conteúdo da interface. O signo visual estabelecido no momento da interação não é necessariamente o mesmo para todos os usuários, como não é necessariamente o mesmo do projetista. A LV deve buscar uma aproximação, de modo que o signo do projetista seja reconhecido pelo maior número possível de usuários, no maior número possível de circunstâncias. Discutir as diretrizes de orientação para uma linguagem visual de interface equivale a discutir qual é o maior nível de interseção possível entre os vários interpretantes. Se o projetista não dispõe de nenhuma orientação quanto ao uso dos recursos expressivos, ele estará frequentemente sujeito a soluções mais ou menos acertadas, dependendo praticamente de sua habilidade pessoal.

Conforme nossa hipótese de trabalho, a codificação sistemática da linguagem e a incorporação da cultura do usuário são requisitos importantes para se chegar a um bom nível de interseção entre os interpretantes. Neste contexto o projeto é fundamental. "*Design is not something that can be applied after the fact, when the fundamental organization of the product has already been determined*" [Mullet & Sano 95, p. 7]. Para atingir um bom resultado o projetista deveria definir a LV antes de especificar a interface. Por exemplo, o projetista definiria a utilização do recurso cor para expressar as propriedades de um objeto. A definição prévia permitiria estruturar o meio de expressão com características mínimas de código, correlacionando elementos da expressão e do conteúdo de forma sistemática. Na medida em que esta correlação pudesse incorporar um certo padrão cognitivo, o signo visual do projetista teria melhores possibilidades de ser reconhecido e o usuário teria mais facilidade em expressar suas intenções. Assim os objetivos de usabilidade do sistema seriam mais facilmente atingidos.

1.4. Metodologia de Trabalho e Resultados

O trabalho foi desenvolvido através de uma metodologia analítico-construtiva, de acordo com as seguintes etapas. A partir da constatação das dificuldades de geração de LVs foi eleito um conjunto de elementos teóricos capazes de oferecer suporte adequado ao problema específico das LVs. Neste conjunto a teoria semiótica se destaca oferecendo um arcabouço teórico capaz de acomodar, dentro dos mesmos pressupostos de codificação linguística, o tratamento de linguagens textuais e visuais, assim como linguagens em ambiente humano e ambiente computacional. Esta questão é importante porque nos permite tratar o uso dos recursos visuais integrados com recursos textuais, sob uma mesma estrutura teórica. Parte dos elementos teóricos considerados foram rapidamente apresentados no tópico anterior, serão discutidos na íntegra no Capítulo 2.

À luz dos elementos teóricos foram avaliadas algumas ocorrências de fenômenos linguísticos de interesse. Foi feito um estudo de caso entre editores gráficos, incluindo dois desenhadores e dois pré-processadores. Entre outras coisas, este estudo objetivou capturar informações sobre a cultura estabelecida, buscando elementos orientadores da formação do signo visual. Foi construído um quadro de avaliação teórica destes fenômenos, sobre o qual foi proposto um ferramental técnico capaz de auxiliar os projetistas. A partir desse ferramental foram re-editados os fenômenos linguísticos, avaliando-se as diferenças entre a situação original e a situação apoiada pela ferramenta.

Os resultados desta dissertação são, portanto:

- um quadro analítico para LV de interfaces gráficas por MD para SMG
- uma notação para a sistematização de escolhas expressivas
- uma proposta para implementação da notação, em um ambiente de geração de signos visuais.

1.5. Resumo da Estrutura do Trabalho

A apresentação do trabalho se divide da seguinte forma.

- O Capítulo 2 discute trabalhos correlatos e apresenta as bases teóricas que sustentam o desenvolvimento do trabalho e os resultados obtidos.
- O Capítulo 3 discute especificida des da LV e aspectos da abordagem semiótica à LV.
- O Capítulo 4 apresenta a análise dos recursos expressivos (visuais e de manipulação) disponíveis no ambiente computacional, e sua estruturação em código linguístico.
- O Capítulo 5 apresenta um estudo de caso averiguando o uso de recursos expressivos visuais em aplicativos disponíveis no mercado comercial.
- O Capítulo 6 apresenta a notação para sistematizaçã o das escolhas expressivas (STAG), como uma ferramenta de apoio para especificação e análise de LVs.
- O Capítulo 7 apresenta uma proposta para implementação da notação, através do Ambiente para Geração de Signos Visuais (AGSV). Este ambiente implementa as diretivas de orientação para codificação da linguagem visual (DOCLV) e as heurísticas de avaliação da linguagem visual (HALV).
- O Capítulo 8 apresenta as conclusões finais, incluindo a ava liação dos resultados deste trabalho e propostas de continuidade.

É de domínio comum o pressuposto de que a regularidade da linguagem é um elemento importante no que se refere à sua capacidade expressiva. A especificação de LVs de forma sistemática necessita uma abordagem especial, que ofereça um tratamento adequado às especificidades dos recursos visuais. Neste capítulo é apresentado um detalhamento dos pressupostos teóricos sobre os quais este trabalho foi construído, atendendo a esta necessidade. Dois grandes eixos teóricos foram explorados, a abordagem semiótica, no que se refere à produção sígnica, a abordagem cognitiva, no que se refere às questões de percepção e compreensão. Será apresentada também uma visão geral de diferentes abordagens utilizadas no projeto de interfaces, avaliando a potencial contribuição de cada uma, com relação ao nosso propósito.

2.1. Considerações Gerais

Estudar o processo de significação na comunicação humana é uma tarefa difícil pela amplitude do universo no qual ela se realiza. Em ambiente computacional esta tarefa torna-se mais acessível, uma vez que a comunicação está restrita ao domínio da aplicação e o ambiente do sistema é um ambiente de codificação formal, não havendo interpretação sobre o código linguístico semelhante à interpretação na comunicação humana. No ambiente computacional ocorrem traduções sucessivas entre as várias camadas linguísticas do sistema, desde a ativação da interface até o ambiente de execução da máquina [Winograd & Flores 86]. Em cada camada os códigos sintático e semântico são prévia e rigidamente estabelecidos, sempre executando os mesmos mapeamentos entre os elementos de ativação da interface e o ambiente de execução da máquina. Por exemplo, a ativação de um comando Save na interface sempre será mapeada em um mesmo procedimento, de acordo com o modelo conceitual do sistema. Este procedimento terá um objeto padrão, como por exemplo o texto corrente ou o conjunto de textos em desenvolvimento, e uma forma de operação padrão, como por exemplo a gravação do texto corrente e a geração de um backup com o texto original, ou apenas a gravação do texto corrente. A cada ativação do comando Save o mesmo procedimento será executado, de forma regular. Em outras palavras, a semântica da linguagem de interface é determinada, sob a perspectiva dos interpretadores computacionais.

Embora no ambiente computacional o processo de significação seja controlável, na fronteira entre o ambiente computacional e o ambiente humano, ou seja, na interface, existem dificuldades. Pelo lado do usuário o conteúdo comunicado pela linguagem está sujeito a variações de interpretação, de forma semelhante à comunicação humana.

Enfoques Cognitivo e Semiótico da Interface

Neste trabalho a interface é observada sob duas óticas. A primeira é aquela em que ela é um artefato de troca de mensagens entre usuário e sistema. A engenharia cognitiva [Norman 86] esclarece o processo de contrução da mensagem usuário-sistema. Norman chama de golfo de comunicação a distância a ser vencida na comunicação usuário-sistema. Ele divide o golfo de comunicação em golfo de execução e golfo de avaliação, segmentando o processo de travessia em etapas de tradução, conforme Figura 2.1. No golfo de comunicação: o usuário formula uma intenção, traduz esta intenção em uma operação do sistema e ativa o sistema. As duas primeiras etapas são mentais e a última etapa é física (de manipulação). No golfo de avaliação é feito o caminho inverso: o usuário interpreta o comportamento do sistema; percebendo a informação do sistema, compreendendo a informação percebida e avaliando o resultado com relação ao seu objetivo inicial. Neste caso a primeira etapa é física (de percepção) e as duas últimas são mentais.

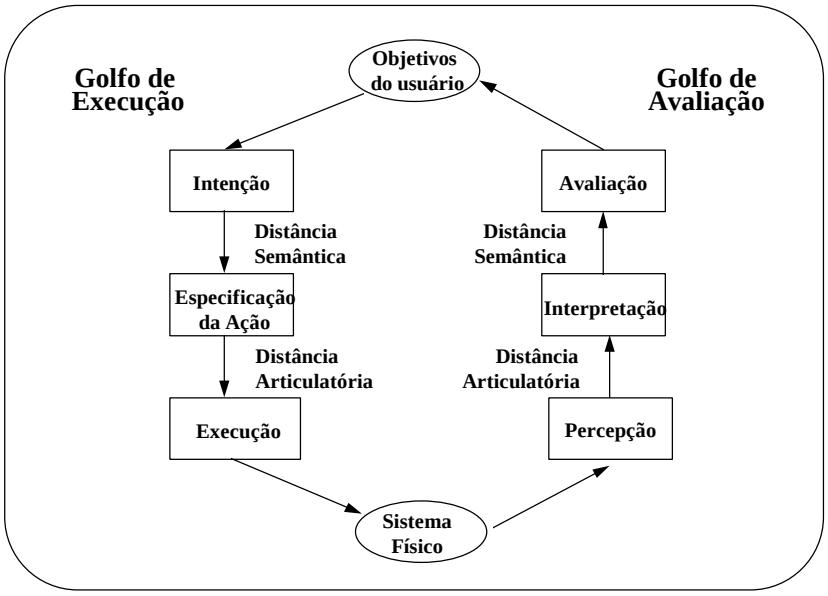


Figura 2.1. Golfo de comunicação

A relevância da proposta de Norman está na distinção entre aspectos mentais de formulação da solução e aspectos físicos de ativação do sistema. A formulação da solução será tão melhor sucedida quanto melhor o projetista puder comunicar o modelo funcional do sistema. A ativação física do sistema será tão melhor sucedida quanto melhor ele puder comunicar o modelo de interação.

A segunda ótica de análise para interfaces é a da engenharia semiótica [Souza 93b], que complementa a proposta da engenharia cognitiva. Conforme apresentado na Seção 1.3.1, ela trata a interface como um artefato de metacomunicação, distinguindo dois níveis de mensagens veiculadas. De acordo com a engenharia semiótica, a interface é uma via unidirecional através da qual trafegam mensagens do projetista para o usuário, ao mesmo tempo em que ela própria emite e recebe mensagens no processo interativo com o usuário. "*The one-shot messages designers send to users systems are of a peculiar type. Firstly, they are* ***performing messages*** *(unlike books, for instance); and secondly, what they perform is itself a communicative act, in which the message plays the role of sender and receiver of other messages. Consequently, systems can be rightfully taken as* ***metacommunication artifacts***." [Souza 93b, p. 756]. Esta visão distingue dois níveis de comunicação: um do projetista para o usuário e outro entre usuário e sistema. Esta proposta está ilustrada na Figura 2.2.

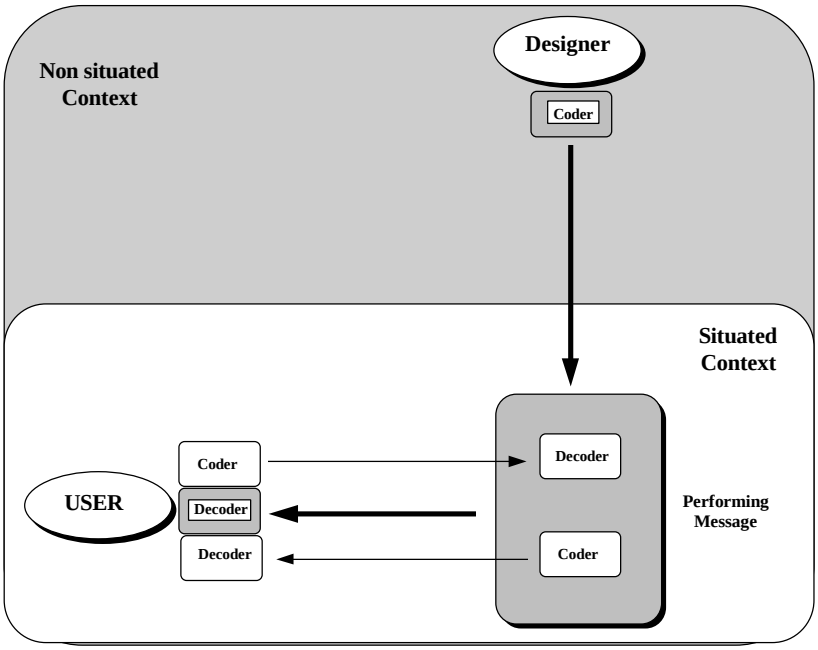
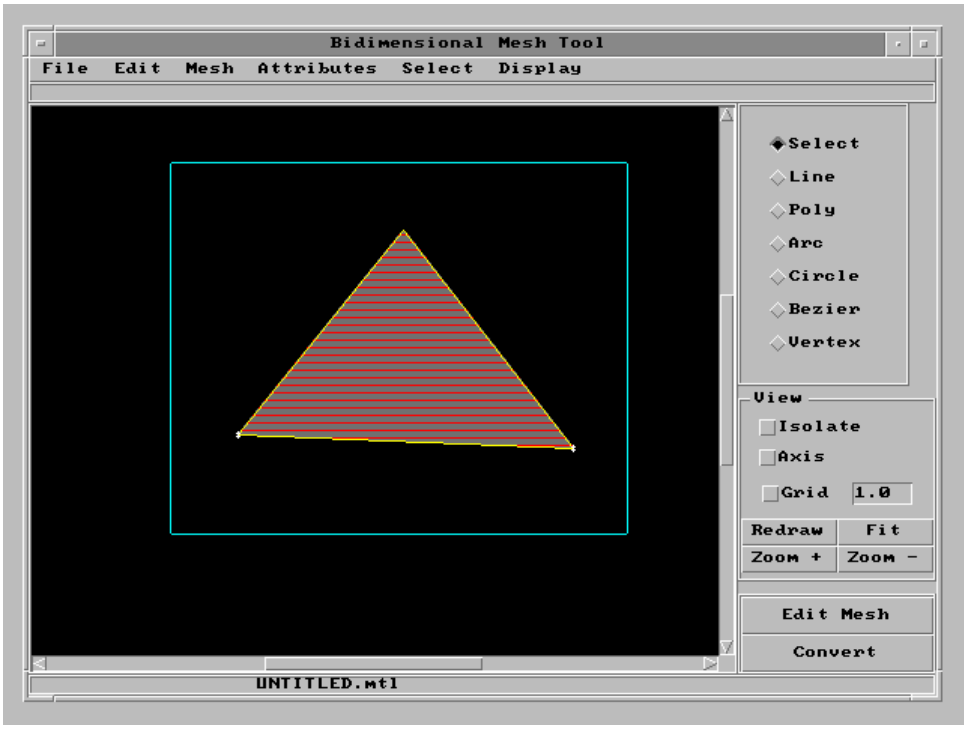


Figura 2.2. "*Two levels of communication in HCI*" [Souza 93b]

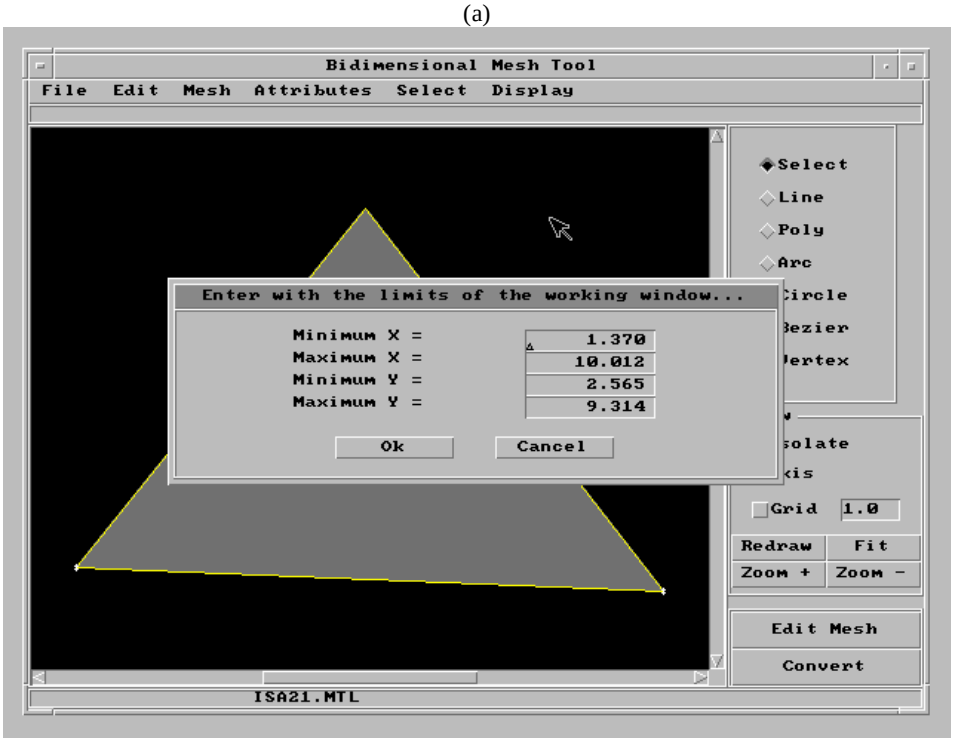
A proposta da engenharia semiótica é relevante em dois aspectos. Um deles refere-se à própria abordagem semiótica feita ao processo de comunicação, a qual não distingue pelo aspecto estrutural as linguagens formais das linguagens naturais. A diferença entre elas é que em ambientes de linguagens formais, não existe a noção de signo (Seção 1.3.2), porque não existe o elemento humano interpretador. No ambiente formal o interpretador é o processador da máquina, portanto a interpretação é formal, não gerando interpretantes diferentes daqueles previamente fixados. Portanto, a linguagem de interface, que atua na fronteira entre os ambientes humano e computacional, pode ser tratada através de um arcabouço teórico único.
Outro aspecto de importância verificado na proposta da engenharia semiótica é a distinção entre os dois níveis de mensagem (Seção 1.3.1). A linguagem de comunicação pode ser observada em termos de uma informação passiva (do projetista para o usuário), com o objetivo de comunicar o modelo conceitual do sistema, e um diálogo (entre usuário e sistema), no qual os aspectos de ativação/resposta e os elementos de discurso podem ser observados. O modelo conceitual do sistema compreende o modelo funcional (referente ao domínio do sistema) e o modelo de interação (referente à interação usuário-sistema).
Uma diferença fundamental entre a engenharia cognitiva e a engenharia semiótica é que na primeira a distância a ser vencida pelo usuário na comunicação é uma distância ideal, é uma distância para uma semântica fixa, estática. Nessa abordagem da comunicação o problema do projetista é tornar a compreensão do usuário possível. Na engenharia semiótica, com a introdução do conceito de interpretante, emerge a dinâmica deste processo. Agora, a distância a ser vencida pelo usuário na comunicação é uma distância entre interpretantes. Este enfoque não caracteriza uma semântica fixa, ele caracteriza o interpretante do projetista (implementado através do sistema) e os enumeráveis interpretantes dos usuários. Neste caso o problema não é mais tornar a interpretação possível, uma vez que não existem garantias de uma determinada interpretação. O problema agora é aproximar ao máximo os interpretantes, em torno de uma interpretação preferencial. A engenharia semiótica revela a ilusão de alvos fixos.
Sob o enfoque da engenharia semiótica, tanto os processos de interação como os de interpretação são dinâmicos. Ela permite usar a dinâmica da interação para atuar na dinâmica da interpretação. Através de diferentes formas de interação o projetista pode buscar aproximar o usuário de uma interpretação preferencial.
De acordo com os enfoques acima, este trabalho será desenvolvido com base em dois grandes eixos teóricos. Um deles referente à abordagem semiótica [Eco 76], através da qual será estudado o processo de produção sígnica, fundamental para a composição das mensagens. O outro, referente aos aspectos cognitivos do usuário, que envolvem questões de percepção e compreensão [Jackendoff 83], bem como o uso de metáforas [Lakoff & Johnson 80].
A seguir faremos uma análise mais detalhada sobre o uso de LVs na comunicação usuário-sistema. Serão apresentados, na sequência: uma caracterização do tipo de MD sobre a qual estamos tratando, aspectos de modelagem em sistemas de captura de dados através da representação visual, diferentes abordagens no projeto de interfaces gráficas e uma análise de aspectos semióticos e cognitivos ao projeto de LVs para interfaces usuário-sistema. Uma vez que nenhuma teoria cobre todos os aspectos do problema, usaremos as visões de diferentes autores, mostrando de que forma elas se complementam.

2.2. Aspectos de Distinção da Manipulação Direta

Nos SMG o usuário pode expressar dois tipos básicos de interação com o sistema: direta e indireta. Chamamos interação direta àquela executada diretamente sobre o objeto gráfico, ativada por manipulação do mesmo. Chamamos interação indireta àquela ativada através de menus, ícones, linguagens de comando ou equivalentes. A diferença entre ambas é que na MD o usuário atua sobre o objeto, na indireta ele faz um discurso sobre o objeto, atuando no mesmo através de recursos linguísticos.
A MD caracteriza-se por aspectos como: (a) representação contínua do objeto de interesse, (b) ação física ao invés de uma atuação indireta através de um discurso linguístico e (c) retorno visual imediato relativo às alterações sobre o objeto [Shneiderman 82/83, Hutchins et alii 86, Foley et alii 93, Ziegler & Fähnrich 88]. Como consequência destas características podemos citar ainda: (d) o componente de saída é o mesmo componente de entrada e (e) ambos agentes da comunicação compartilham uma representação comum. A mais forte característica da MD é a manipulação e o comportamento do objeto computacional de forma análoga ao mundo real, ou seja, de forma análoga àquilo que o objeto computacional representa.
O objeto gráfico não é um objeto físico, é representacional. Porém, da forma como é tratado por sistema e usuário, em entrada e saída, ele tem uma natureza indicial. Peirce (31), introduz os conceitos de ícone, índice e símbolo, diferenciando-os. Segundo ele, o ícone tem similaridade com o objeto representado, o índice é uma causa influenciada pelo objeto ou marca contiguamente relacionada ao objeto, e símbolo é convencional, arbitrado. O objeto gráfico manipulado não é um ícone na medida em que ele não é uma imagem do objeto real, mas uma representação do objeto computacional. " *...the visible informaton becomes a part of a model of the conceptual world in the computer.*" [Ziegler & Fähnrich 88, p. 125]. Embora seja simbólico, ele também não é um símbolo totalmente arbitrado. O objeto gráfico é uma representação do objeto computacional, com uma natureza indicial no que se refere ao objeto real. Por sua natureza indicial, na MD o objeto se comporta de forma análoga ao objeto no mundo real. Segundo Brennan "*In direct manipulation, the elements on the screen behave as if they are the objects that they represent.*" [Brennan 90, p. 393]. A analogia permite ao usuário manipular a representação visual em consonância com a forma como raciocina sobre o problema, "*...in a form that matches the way one thinks about the problem.*" [Hutchins et alii 86, p. 90].



(a)



(b)

Figura 2.3. Definição de uma janela por interação direta e indireta

Vamos exemplificar ambos os casos de manipulação através da definição de uma janela. Na interação direta o usuário traça a janela com um movimento de pressão-e-arrasto do mouse, no sentido diagonal. A janela vai sendo desenhada simultaneamente ao movimento do mouse, o retorno do sistema é imediato à manipulação. A Figura 2.3 (a) ilustra o exemplo através do aplicativo MTool [MTool 92]. Na interação indireta o usuário ativa um procedimento para traçado da janela, fornecendo os valores das coordenadas que a definem. A Figura 2.3 (b) ilustra o exemplo através do mesmo aplicativo, onde a operação foi ativada através do menu Display/Limits.

A interação indireta também pode ser feita através de LV, utilizando-se por exemplo uma barra de ferramentas. Observe-se que embora a utilização de ícones caracterize uma LV, trata-se de um tipo de linguagem equivalente à linguagem textual, no sentido de que o ícone representa um discurso sobre o objeto. A LV por ícones não deve ser

confundida com a LV por MD com a qual estamos tratando. Embora os SMG utilizem os dois tipos de interação, direta e indireta, este trabalho trata especificamente da primeira.

2.3. O Modelo Visual para Captura de Dados

A modelagem de um sistema computacional é o processo básico de transposição do problema do ambiente real de ocorrência para o ambiente computacional [Abelson & Sussman 85]. No processo de modelagem de sistemas gráficos [Foley et alii 93] o problema do mundo real é ajustado a um modelo matemático, o qual capta as propriedades do objeto, relevantes ao estudo em foco. A partir daí é definido o modelo de representação, que descreve as estruturas simbólicas que permitem a representação e a manipulação das informações em meio digital. Finalmente o sistema é codificado. O processo de modelagem está ilustrado na parte superior da Figura 2.4.
Quando os SMG utilizavam uma entrada de dados a partir de arquivos texto em "batch", extensos arquivos de dados alfanuméricos eram editados e os dados eram fornecidos em um formato mais próximo do ambiente no qual seriam processados. Neste caso os dados eram de natureza totalmente simbólica (não icônica, não indicial). A migração da entrada de dados em "batch" para uma entrada de dados através da representação visual teve por objetivo facilitar a interação do usuário com o sistema. A representação visual oferece ao usuário uma representação dos dados mais próxima da realidade perceptível; ela oferece uma melhor aproximação do objeto real. Porém, a representação na tela deve ser coerente com o modelo de dados implementado pelo sistema que, por sua vez, é uma abstração sobre a realidade.

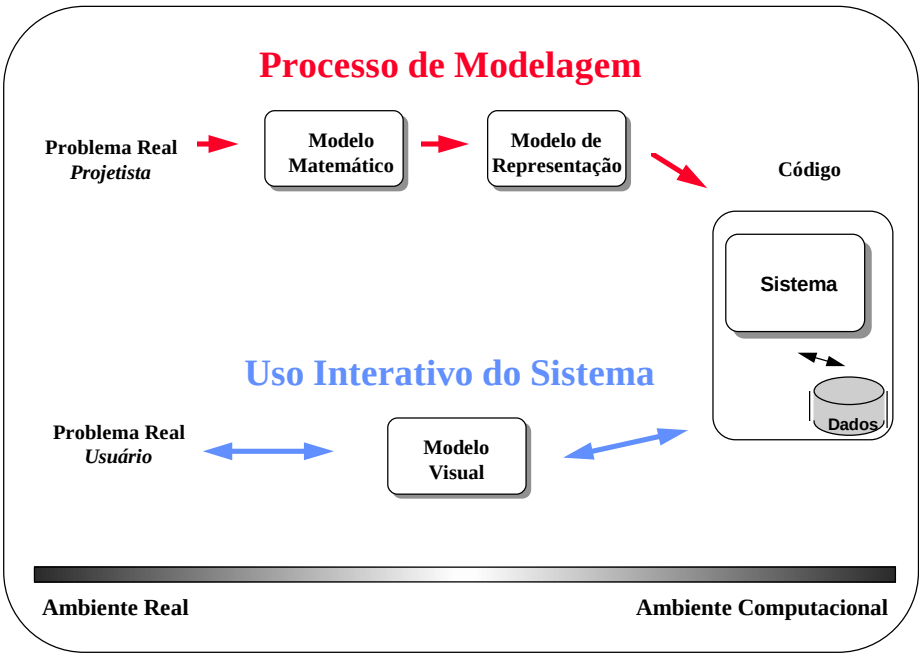


Figura 2.4. Processo de modelagem e interação nos SMG

Para sustentar a especificação desta representação surge uma nova etapa no universo de modelagem do sistema, o modelo visual. Nesta etapa de modelagem é determinada a representação visual correspondente à representação interna do modelo discretizado (sistema implementado). Esta representação não precisa ser necessariamente realista, pode ser esquemática ou totalmente convencional. A representação visual definida pelo projetista deve ser coerente com a complexidade do problema modelado, e coerente com o tipo de operações possíveis sobre o objeto. Representações mais esquemáticas ou mais realistas podem alterar as soluções adotadas na linguagem de interação. Por exemplo, soluções esquemáticas remetem a modelos matemáticos e podem ser tratadas como tal, enfatizando o caráter abstrato da aplicação.
Através da representação visual o usuário interage com o sistema (parte inferior da Figura 2.4). O modelo visual é a síntese dos dois mundos, ele é interpretável por máquina e usuário [Harel 95]. No ambiente computacional a manipulação dos elementos visuais está sujeita às restrições do modelo abstrato e do modelo de representação adotados na modelagem. O modelo abstrato determina as regras de consistência dos dados capturados, o modelo de representação determina as possibilidades de operação sobre os dados no ambiente computacional. No ambiente do usuário a manipulação dos elementos visuais está sujeita ao modelo mental que o usuário tem do objeto. A ponte para ultrapassar o golfo de comunicação entre usuário e sistema, conforme descrito no tópico anterior, é construída sobre o modelo visual. Portanto, a LV será construída no sentido de manipular o modelo visual adotado.

A Manipulação de Objetos Computacionais

Como foi dito, nos SMG a captura de dados é feita através da representação visual. Na interação usuário-sistema, observamos a ocorrência de conflitos referentes à consistência da percepção visual do usuário (Capítulo 1). Estes conflitos geralmente estão relacionados a uma expectativa do usuário sobre o comportamento do objeto

computacional. Essa expectativa pode estar ligada a uma semelhança da representação visual com o objeto real, pode estar ligada a uma abstração semelhante utilizada na modelagem em outro contexto (por exemplo papel e lápis), ou, ainda a uma representação semelhante em outro sistema computacional (por exemplo um desenhador).
Em alguns casos, os modelos visuais adotados refletem a cultura do usuário em representações em papel e lápis. Por exemplo, modelos esquemáticos, com base matemática, adotados em sistemas de modelagem para artefatos de engenharia muitas vezes utilizam representações visuais similares às tradicionais, desenhadas manualmente. Essa aproximação com um modelo de engenharia conhecido, facilita ao usuário a construção da representação visual do objeto, em função do conhecimento prévio (cultura formada) sobre a abstração matemática. Porém, deve ser lembrado que apesar da possível semelhança entre as representações computacional e manual, a primeira estará restrita às condições do modelo semântico do sistema. Nos SMG a representação visual é o retrato do objeto no meio computacional, e seu comportamento é compatível com o modelo implementado pelo sistema.
Por exemplo, na Figura 2.5 (a) é apresentada uma representação esquemática simplificada de um túnel passando por diferentes camadas de solo. Se através de um processo de edição o traçado do túnel fosse retirado, a semelhança com o objeto real sugeriria que as camadas de solo fossem automaticamente integradas, conforme solução (b). Porém, no objeto computacional o apagamento do túnel deixa uma interrupção no traçado referente às camadas de solo, solução (c), o qual deve ser refeito pelo usuário. Se, ainda, através de um processo de edição, as linhas de delimitação das camadas de solo fossem recuperadas "ligando-se os pontos", a semelhança com a manipulação de uma mesma representação traçada em papel e lápis sugeriria que uma vez interligadas as linhas, as fronteiras fossem recuperadas com linhas contínuas, conforme solução (b). Porém, isto nem sempre ocorre. Alguns aplicativos, como por exemplo o MTool [MTool 92, 97], em função de seu modelo semântico, mantém um vértice nos locais de ligação, sinalizando a interligação de segmentos de reta distintos (solução (d)), ao invés de um segmento contínuo.

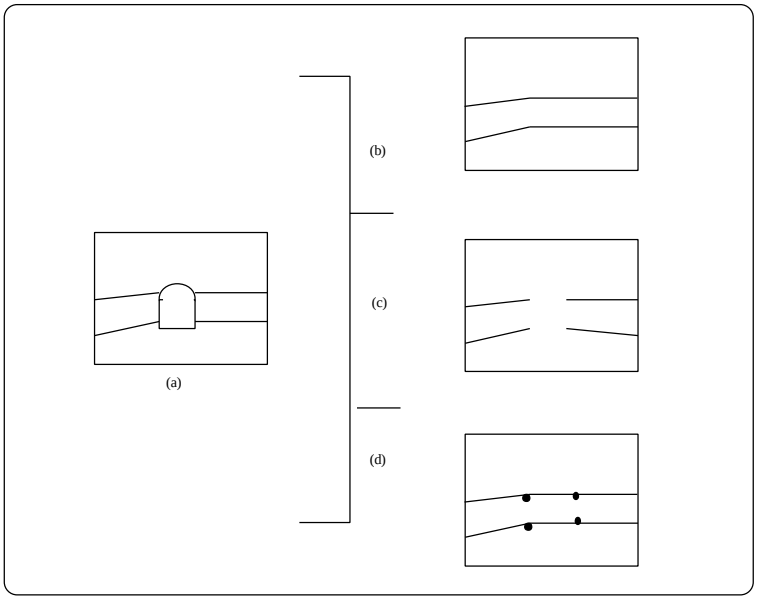


Figura 2.5. Edições sobre a representação visual de um objeto

A expectativa do usuário também pode estar ligada a um comportamento compatível com o meio computacional. Por exemplo, a partir de uma cultura formada no uso de desenhadores, pode-se supor que o alongamento da representação visual de um objeto é possível em um SMG. O "alongamento" é uma questão marcadamente representacional, como o é, também, o "encolhimento" de uma imagem para metade de suas dimensões. Em um SMG, porém, "alongar" um objeto representado só pode ter como semântica redimensionar o modelo subjacente. Assim, o alongamento pode ser bloqueado em função das restrições do modelo do sistema. Nos vários exemplos citados, a quebra na consistência da percepção visual do usuário decorre de um hiato entre a sua expectativa e o comportamento apresentado pelo objeto computacional. No sentido de superar este hiato é importante que o projetista expresse o melhor possível a natureza do objeto computacional.

2.4. Diferentes Abordagens ao Projeto de Interfaces Gráficas

Neste tópico faremos uma análise de diferentes abordagens teóricas ao projeto de interfaces. O projeto de interfaces pode ser tratado através de enfoques tão diferentes como um enfoque cognitivo [Card et alii 83; Norman 86], estético [Marcus 92], voltado para ambientes de programação, como orientação a objetos [Lee 93], ou linguístico semiótico [Andersen 90, Souza 93b]. Cada um deles oferece vantagens e desvantagens, atendendo a objetivos específicos relativos à interação homem-máquina. Como o universo de trabalhos dirigidos a projetos de

interfaces é muito grande, com o intuito de direcionarmos nossa análise, restringiremos a busca àqueles mais diretamente relacionados às LVs.

2.4.1. Abordagens Orientadas para a Legibilidade Visual

Algumas abordagens atuam a nível estético, orientando sobre a legibilidade visual (legibility) [Marcus 92]. Por exemplo, as orientações estéticas auxiliam na definição de layout como: definição de espaços de tela com proporções agradáveis, alinhamento, enquadramento; auxiliam na padronização visual, no sentido de criar interfaces personalizadas; auxiliam também na composição visual como: uso de bordas, botões, cor, fontes, e projetos de ícones [Marcus 92, Microsoft Corporation 95].

As orientações estéticas tratam também sobre a utilização de recurso visual como recurso expressivo, para conteúdos genéricos, como, por exemplo, uma sinalização visual sobre o caminho percorrido pelo usuário nas escolhas de menu. Estas orientações são especialmente interessantes porque refletem a cultura estabelecida na área. Quando um fabricante de software como Microsoft Corporation (95) apresenta um conjunto de formas de menus, como por exemplo "menu bar", "drop-down menu", "pop-up menu", ou um conjunto de formas de controle, como por exemplo "buttons", "list boxes", "drop-down combo boxes", está sendo apresentada a cultura estabelecida na área. Fabricantes de aplicativos com o porte da Microsoft têm a capacidade de criar cultura na área, em função de sua capacidade de distribuição internacional de produtos, a preços competitivos, e do grande consumo. No geral, livros publicados por estas empresas documentam um estilo de produto já lançado, registrando uma realidade. Embora não sejam tratados sob este enfoque, estes livros documentam um sistema semiótico vigente.

Neste sentido deve-se observar que, em alguns casos, o poder dos fabricantes de grande porte em termos de imposição de cultura tecnológica é tão grande que se implanta uma certa cultura de erros. Soluções perceptivelmente inadequadas são implementadas e perpetuadas, sendo replicadas em aplicativos de fabricantes de menor porte, ou em aplicativos exclusivos de empresas, simplesmente para aproveitar (ou não contrariar) o conhecimento adquirido do usuário. Por exemplo, em praticamente a totalidade dos aplicativos para Windows a opção de Exit, para encerramento de um aplicativo, aparece associada à opção File do menu. Por coerência esta função só deveria estar associada à opção File caso seu objeto fosse o arquivo e não o aplicativo, inclusive porque pode haver mais de um arquivo aberto simultaneamente. Embora esta opção não seja a mais adequada trata-se de uma cultura já estabelecida e a maioria dos projetistas de interfaces reluta em alterá-la, alegando, com razão, que os usuários já estão "acostumados" a ela, buscando-a automaticamente.

Embora as orientações apresentadas pela Microsoft Corporation (95) sejam de modo geral voltadas para aspectos de legibilidade visual, elas são baseadas em princípios de projeto centrado no usuário. Por exemplo, alguns destes princípios são: a necessidade de consistência na interface, a necessidade absoluta de retorno para o usuário, visibilidade da informação, orientações para usuários principiantes. Alguns destes princípios básicos foram consagrados pelo uso e pela comprovação de sua eficiência, sendo adotados inclusive por outros autores com diferentes abordagens. Por exemplo, tanto em Mullet & Sano (95) quanto em Microsoft Corporation (95), propõe-se a simplicidade como um princípio de projeto, orientando para: eliminação dos excessos de detalhes de embelezamento, das informações irrelevantes e das representações realistas onde as esquemáticas são mais claras. A simplicidade facilita a compreensão do usuário, o uso e o aprendizado da interface.

Embora as orientações que envolvem legibilidade visual devam ser consideradas no projeto de LVs, pelo fato de representarem uma cultura estabelecida na área, por si só elas não atendem à especificação de LVs por MD. Em geral são oferecidas linhas de orientação, a maior parte das quais definidas com base em processos de observação e experimentação. Não é oferecido um arcabouço teórico para tratamento do recurso visual como um recurso linguístico. Nota-se claramente que as referidas propostas para o projeto de interface aplicam-se depois que uma LV fundamental tenha sido projetada e não em lugar de uma.

2.4.2. Abordagens com Enfoque Cognitivo

Compreender o raciocínio humano, o que a mente faz e com que recursos, é um trabalho sobre o qual muitos pesquisadores têm se debruçado há anos, através da psicologia. No final da década de 50 o surgimento do computador ofereceu um novo paradigma para a psicologia, chamado psicologia cognitiva [Booth 89, Preece et alii 94]. Através de uma analogia da mente humana como o computador, o ser humano é visto como um processador de informações, no qual os órgãos sensores (visão, audição, olfato e tato) funcionam como entradas de dados . "*In general, cognition refers to the processes by which we become acquainted with things or, in other words, how we gain knowledge. These include undestanding, remembering, reasoning, attending, being aware, acquiring skills and creating new ideas*" [Preece et alii 94]. A psicologia cognitiva desenvolveu-se fortemente na década de 70 dando origem à ciência cognitiva [Booth 89].

No projeto de interfaces as abordagens cognitivas são aquelas centradas nos processos de percepção e compreensão do usuário [Card et alii 83; Norman 86]. Através da teoria cognitiva aplicada ao estudo da interação usuário-sistema, busca-se compreender o raciocínio do usuário, no sentido de atingir os objetivos a que se propõe, este ligados a tarefas. Busca-se compreender como o usuário percebe e processa a informação, e como estrutura seu conhecimento, o que lhe permite transformar suas tarefas em atividades orientadas a um objetivo [Preece et alii 94]. Esta averiguação inclui processos de armazenamento de informação, considerando o tempo que o usuário retém determinados tipos de informação (memória de curto prazo, memória de longo prazo) [Preece et alii 94],

inclui também processos de categorização da informação [Jackendoff 83, 87], assim como processos metafóricos e metonímicos presentes na linguagem [Lakoff & Johnson 80, Norman 88].

Ferramentas de Especificação e Análise nas Abordagens Cognitivas

Na abordagem cognitiva das interfaces, as gramáticas surgem como ferramentas de especificação e análise, com a propósito de auxiliar a descrição das interfaces de acordo com o modelo de tarefa do usuário. Por exemplo, a TAG - Task-Action Grammars [Payne & Green 86], uma gramática com base na tarefa, busca mapear a intenção do usuário em ativações do sistema. A D-TAG [Howes & Payne 90, Payne 91], propõe uma extensão da TAG, considerando a reestruturação das tarefas em função dos artefatos de mediação entre os objetos e o usuário. Porque as gramáticas são gerativas, garantindo a expressão regular dos procedimentos elementares, elas garantem um certo nível de sistematicidade na linguagem. De forma semelhante, o modelo GOMS - Goals, Operators, Methods and Selection Rules [Card et alii 83] busca capturar a forma como o usuário estrutura sua tarefa em metas e sub-metas. Ambos buscam antecipar a forma como o usuário raciocina, transferindo este conhecimento para a estrutura da interface.
No projeto de interfaces textuais as gramáticas se demonstram bastante úteis como ferramentas de especificação e análise. Porém, no caso específico das LVs por MD elas se demonstram insuficientes. De modo superficial podemos dizer que esta insuficiência está ligada ao seu escopo de atuação. Quando utilizada para especificação de interfaces textuais, a gramática não trata o nível de lexicalização da linguagem em separado, porque o projetista geralmente lança mão de um código linguístico já conhecido, pela sua proximidade com a linguagem natural. O projetista aproveita uma lexicalização já existente. Por exemplo, na TAG pequenos aspectos de lexicalização, como o uso de um determinado sinal do mouse para expressar um certo conteúdo, são especificados misturados às regras gramaticais, que definem as sentenças da linguagem. No caso da LV, como o código visual é especificado pelo projetista, seria necessário que a ferramenta oferecesse recursos para tratamento específico da lexicalização.
Gramáticas como a TAG também não permitem uma verificação adequada e sistemática sobre a especificação de uso do recurso visual. Por exemplo, é dificil verificar sobre a gramática se um mesmo recurso visual está sendo utilizado para expressar conteúdos de naturezas diferentes, como por exemplo, a cor amarela caracterizando um material e a cor verde caracterizando um estado. Além disto, ela não oferece mecanismos de controle para questões específicas das LVs, como por exemplo a sobreposição de recursos expressivos, onde recursos como cor e textura podem se sobrepor sobre a forma, interferindo um com o outro. As causas desta insuficiência serão discutidas no Capítulo 5, quando será feita uma análise detalhada da TAG. No Capítulo 6 será proposta a notação para sistematização de escolhas expressivas, que é uma extensão da TAG, baseada em princípios semióticos de estruturação de linguagens, com vista a especificação de LVs por MD.

Um enfoque linguístico sobre o projeto de interfaces
Mullet & Sano (95) apresentam uma nova visão do projeto visual. Eles questionam o uso "pobre" dos recursos visuais, onde toolkits estão disponíveis para construção de interfaces gráficas, porém muito pouco suporte é oferecido para o projeto visual orientado à comunicação. Sua proposta de projeto visual estabelece uma preocupação com a forma, no contexto de uma tarefa específica, "*...attempts to solve communication problems in a way that is at once functionally effective and aesthetically pleasing.*" [Mullet & Sano 95, p. 1]. Eles abordam o projeto de interfaces como um projeto orientado à comunicação, onde o objetivo é desenvolver uma mensagem visual que possa ser transmitida e interpretada com precisão. Pressupõem um vocabulário formal contendo os elementos básicos de projeto, a partir do qual são codificadas as representações de alto nível, e uma sintaxe visual descrevendo de que forma os elementos podem se combinar dentro deste sistema. Esta proposta busca um tratamento sistemático da expressão visual, incorporando à interface uma estrutura e regularidade linguísticas, objetivando basicamente a consistência. "*When the visual language of a particular GUI standard is not used consistently throughout the user's environment, its ability to reinforce communication is greatly diminished.*" [Mullet & Sano 95, p. 152].
Apesar desta visão abrangente da interface como um transmissor de mensagens visuais, a abordagem falha no tratamento dos SMG porque: (a) o elemento visual básico tratado como signo é um elemento de alto-nível, não é oferecida uma estrutura linguística para tratamento do recurso visual básico de traçado, (b) a representação visual tratada não é manipulável pelo usuário, como deve ser em um SMG, ela é apenas um meio de expressão do sistema. Lembramos que neste trabalho estamos tratando da manipulação da representação visual dos objetos do domínio do sistema.

2.5. A Abordagem Semiótica ao Projeto de Linguagem de Interface

Entre as abordagens semióticas para computação duas linhas de destacam: a de Andersen (90), cuja origem está na Escola de Glossemática de Hjelmslev; e a de autores como Nadin (88), Saint-Martin (90) e Souza (93b), cuja origem está nas definições de Peirce (31) e Eco (76). Andersen (90) propõe um estudo dos signos e de sua utilização em ambiente computacional. Ele parte do pressuposto que os sistemas de computadores são ferramentas simbólicas, em um ambiente específico, e que são construídos basicamente para pessoas. Portanto, o estudo do processo de significação no ambiente computacional especificamente pode tornar a comunicação com os sistemas

mais acessíveis. Embora seguindo o estruturalismo europeu (glossemática de Hjelmslev), Andersen aborda o ambiente de sistema a partir da tríade peirceana (representamen, object and interpretant), alegando que em termos de utilização os conceitos peirceanos auxiliam mais. Ele vê o sistema " *...in terms of interpretants produced by a group of users, be they developers, end-users, system administrators, maintenance programmers, teachers, or salemen*." [Andersen 93], defendendo que se a execução do programa é vista como signo, o texto do programa pode ser estruturado enfatizando a formação de signos. Apesar de sua visão bastante abrangente, tratando as ocorrências em ambiente computacional como ocorrências semióticas, ele não oferece ferramental linguístico para especificação de linguagens de interface.

Quanto à linha de abordagem semiótica com base em Peirce, Peirce (31) estabelece uma série de conceitos semióticos ainda hoje referenciados. Ele apresenta o processo de significação através de uma tríade. Ele define o conceito de signo como um produto de três elementos correlacionados: um elemento de conteúdo, um elemento de expressão e o interpretante. O signo não é apenas a correlação entre um elemento do conteúdo e um elemento da expressão, mas é o produto dinâmico dessa correlação na presença de um intérprete. Através do conceito de interpretante, ele apresenta o conceito de semiose ilimitada, envolvendo os incontáveis interpretantes possíveis de um determinado conteúdo (conceitos detalhados na Seção 1.3.2.). Estabelece também os conceitos de ícone, índice e símbolo, sobre os quais a maior parte da literatura para computação se apoia até hoje. Segundo Peirce, a qualidade do ícone é sua similaridade com o objeto representado, a qualidade do índice é ser uma causa influenciada pelo objeto, ou marca contiguamente relacionada ao objeto, e a qualidade do símbolo é ser convencional, arbitrado.

Nadin (88) trabalha com base nos conceitos peirceanos. Sua grande contribuição é a relevância que atribui ao tratamento do sistema de signos no projeto de interfaces. Ele explora o fato de que a linguagem permeia a vida humana, que todo conhecimento é compartilhado através da linguagem. Segundo ele, não interagimos diretamente com as ferramentas de nova geração, como acontecia com um martelo ou uma tesoura, mas indiretamente, através de comandos, programas, ou seja, através de linguagem. Portanto, uma ruptura na capacidade destas ferramentas não é mais um problema físico, mas um problema conceitual. Ele defende a incorporação de princípios semióticos à atividade de projeto. " *... to consider design as an activity throught which designers structure systems of signs in such a way to make possible the achievement of human goals* ...." [Nadin 88, p. 51]. Ainda segundo ele, a semiótica pode prover ferramentas prescritivas para suportar uma interação usuário-sistema eficiente, como regras de coerência do repertório de signos utilizados e regras de consistência na operações destes signos. O tratamento prescritivo do sistema de signos permite análises qualitativas e quantitativas sobre a adequabilidade da representação e sobre a dinâmica do signo. " *... semiotics is less a descriptive theory and more a methodology for improved interpretation and evaluation of how people communicate, represent things, express themselves, ...*" [Nadin 88, p. 46]. No que se refere ao nosso propósito, sua abordagem falha basicamente porque trata um signo de alto nível (ícone, índices e símbolos). Ele não oferece um arcabouço metodológico para tratamento da linguagem de interface a partir dos recursos visuais básicos. À medida em que, através dos signos de alto nível, ele concentra seu esforço em integrar o interpretante do usuário à interface, ele perde a referência de que a interface também é o veículo expressivo do projetista, e que portanto deve ser oferecido a ele uma forma sistemática de tratar os recursos visuais básicos, gerando sua própria linguagem.

Eco (76) mantém a noção postulada por Peirce, na qual o signo só se estabelece na presença do intérprete, propondo a teoria de produção de signos. Eco trata a matéria expressiva e a matéria de conteúdo como contínuos a serem segmentados de acordo com a perspectiva e o conhecimento do usuário na comunicação. Os elementos discretizados destes contínuos são organizados em sistemas de códigos, os quais são correlacionados entre si por regras de codificação. Esta abordagem retrata a dinâmica do processo de significação. Ela é especialmente útil na geração de LVs porque as linguagens geradas são particulares a cada projetista. O projetista deve gerar sua própria LV previamente à especificação da interface. Segundo Souza (93b), a teoria da produção de signos nos oferece uma abordagem orientada à síntese para o tratamento de linguagens, que nos permite estruturar a base de uma abordagem teórica, para o projeto de linguagem usuário-sistema. Com base na teoria da produção de signos, Souza (93b) propõe a engenharia semiótica que trata a interface como um artefato de metacomunicação, identificando dois níveis de mensagens, um do projetista para o usuário e outro entre usuário e sistema (Seções 1.3.1 e 2.1). Nosso trabalho foi desenvolvido com base nas propostas de Eco (86) e Souza (93b). Os conceitos básicos sobre a teoria da produção de signos serão detalhados na Seção 3.1, onde serão discutidas as especificidades do signo visual.

Trabalhos mais recentes utilizam a abordagem semiótica para aspectos específicos das interfaces. Em Pimenta & Faust (97) encontramos uma proposta para a engenharia de sofware, apoiada na engenharia semiótica. Os autores propõem uma estratégia para captura dos signos do usuário, pelo projetista, no processo de elicitação de requisitos, buscando estabelecer uma base comum de diálogo entre o projetista e o usuário.

Uma Abordagem Semiótica com Base nos Aspectos Perceptivos

Alguns autores abordam a percepção visual pelo aspecto físico, mais do que pelo aspecto da interpretação. Eles avaliam a capacidade física do indivíduo em perceber informação [Saint-Martin 90, Marr 82]. Por exemplo, Saint-Martin entende como semiótica a relação recíproca e de transformação que ocorre entre os coloremas. Os coloremas são instâncias físicas de uma imagem. O aparelho de percepção relaciona as várias instâncias formando uma imagem contínua. A análise da semiótica da linguagem visual considera a incisão do raio luminoso sobre o

objeto, e como isto produz a cor, a textura, a dimensão e outras percepções, o que equivale a dizer que a função de incidência do raio luminoso altera o interpretante do observador.
Neste trabalho os aspectos de percepção física estão sendo idealizados com base na percepção individual da autora. Estamos nos concentrando nos aspectos de interpretação da linguagem visual, partindo da abstração de que todos os usuários têm a mesma capacidade física de visão. Ou seja, todos enxergam a mesma imagem, a qual não está sujeita a fatores externos de variação ao longo do tempo. Em ambiente computacional, o meio de comunicação visual é o terminal de vídeo. A representação exibida no terminal está sujeita às características técnicas do hardware, e também às condições de representação interna, que muitas vezes determinam a qualidade da imagem exibida. Mas, ao mesmo tempo em que esta representação sofre as restrições do meio, ela apresenta a regularidade e constância características das representações computacionais. As nuances de representação visual que muitas vezes encontramos em ambiente de comunicação humana, não ocorrem em ambiente computacional. Neste ambiente, não acontecem variações nas condições de produção do signo. Portanto, entendemos que as possíveis diferenças de conteúdo percebidas pelos usuários estariam por conta de aspectos de interpretação sobre a imagem capturada. Na análise sobre a utilização do recurso visual nos concentramos em como os usuários interpretam o que vêem.

2.6. Aspectos Cognitivos da Comunicação

Na teoria da produção de signos, Eco (76) trata sobre a discretização dos elementos linguísticos e sua estruturação em sistemas de códigos. Segundo ele, os contínuos são segmentados de acordo com os conhecimentos e as necessidade dos usuários. Por exemplo, um químico observa as substâncias em termos de suas moléculas formadoras; um médico observa a mesma substância em termos de suas propriedades e efeitos orgânicos; um físico pode ainda observar a mesma substância em termos de suas propriedades enquanto elemento flúido. Em cada caso, o mesmo contínuo de conteúdo será segmentado de formas distintas, de acordo com as necessidades e motivações particulares. Eco não discute porém, os fatores cognitivos que estariam presentes nos processos de segmentação e estruturação, como não discute as questões cognitivas que atuariam na definição do código que correlaciona os elementos discretizados da expressão e do conteúdo. Em Jackendoff (83, 87, 91), onde há ênfase sobre processos cognitivos, encontramos suporte teórico para estas questões.
Na abordagem cognitiva ao raciocínio humano existe uma tendência a negligenciar o conteúdo, sendo enfatizado o processamento da informação, nos seus aspectos de estruturação e categorização. O ser humano tem uma capacidade inata de categorizar [Piaget 42, Piaget & Inhelder 59, Piaget & Chomsky 79]. Existe uma estrutura presente e atuante no raciocínio humano, de tal forma que não é possível estudar a natureza do pensamento sem tomar consciência desta estrutura. "*... description in the structure mode is indispensable to psychological theory* ..." [Jackendoff 83, p. 7].
Ao longo do período de crescimento e aprendizado, o ser humano "arquiva" informações a partir das quais reconhece o mundo que o cerca. Uma criança não aprende todas as frases possíveis da língua. No processo de aprendizado, as crianças abstraem regras de formação da língua, de modo a não só poder criar novas frases como a poder reconhecer frases nunca ouvidas antes. As pessoas reconhecem "palavras" ou "fonemas" da língua, o que mostra que têm algum tipo de princípio operante de categorização ou estrutura para tratar palavras distintamente de fonemas, diferenciando-os e reconhecendo-os como pertencentes ou não a um determinado tipo ou grupo. O ser humano vê o mundo através de sua própria estrutura informacional, realimentando-a ininterruptamente. A segmentação que o usuário faz do mundo (sua percepção) é coerente com sua própria estrutura informacional. "...the world as we perceive it is the product, not the source, of such differentiation." [Jackendoff 87, p. 39]. Por exemplo, uma pessoa que não dê importância a carros não consegue distinguir um modelo de outro, porque seu processo de discretização não chega ao nível de detalhes que permitiria distinguir os modelos. Neste exemplo, esta pessoa simplesmente vê um carro ao invés de um Fiat Uno.
Jackendoff (91) estuda os processos de mapeamento da informação sensorial (visão, audição) em informação conceitual. Ele analisa as várias etapas através das quais a informação capturada pelos órgãos sensoriais humanos vai sendo transformada em informação conceitual. Por exemplo, a informação é mapeada da audição para uma estrutura fonológica, daí para uma estrutura sintática e para uma estrutura conceitual, ou então, diretamente dos órgãos sensores (visão, ação e outros) para as estruturas conceituais. Em cada um destes estágios de mapeamento existe um código estruturado com uma gramática interna, que permite, por exemplo, que o ouvinte reconheça um fonema ou uma estrutura sintática como sendo parte da língua, ou uma informação conceitual. Esta última será reconhecida se ele já possuir a informação previamente ou puder inferi-la. O processo de reconhecimento em cada estágio, de acordo com a estrutura interna, não é suficiente para a compreensão. É necessário o mapeamento de um estágio para outro, ou seja, o mapeamento entre os códigos dos diferentes estágios.
O processo de discretização e categorização não é absoluto, o significado das palavras não se restringe a categorias clássicas [Jackendoff 83, 91]. Existe um contínuo através do qual o significado se define como uma instância estereotipada ou uma instância marginal, de acordo com as regras preferenciais. Jackendoff (91) apresenta o exemplo da cadeira como uma categoria funcional, caracterizada por um sistema de regras preferenciais que combinam forma e funcionalidade. Por exemplo, existe um estereótipo para forma de cadeira, combinado a uma funcionalidade padrão de "um lugar para sentar". Os objetos que atendem plenamente as regras preferenciais de forma e funcionalidade são categorizados como uma cadeira clássica. Alguns objetos podem atender de forma parcial a regras preferenciais. Por exemplo, uma banqueta não atende a regras de forma, mas atende a regras de funcionalidade. Estes objetos são instâncias marginais da categoria.

Uma questão relevante para o nosso trabalho é que, segundo Jackendoff, a informação estruturada é mais fácil de ser percebida e compreendida. Sobre a importância da estrutura no aprendizado e na memória, Jackendoff diz que "*... it is impossible to study processes of perception and memory without asking what structures the subjects impose on the material being perceived and remembered.*" [Jackendoff 87, p. 40]. A importância que o autor atribui à estrutura é evidente. Transportando esta afirmação para nosso caso particular, resta-nos averiguar que tipo de estrutura nosso usuário impõe à linguagem de interface, no processo de interação com o sistema.

Sobre o Processo de Percepção do Usuário

É possível traçar um paralelo entre a abordagem semiótica de Eco e a abordagem de Jackendoff. Se a informação estruturada é mais fácil de ser percebida e compreendida, a linguagem estruturada de forma sistemática também será mais facilmente compreendida pelo usuário. A abordagem de Eco ao processo de geração de signo é consistente com a abordagem de Jackendoff, no sentido de que ele discretiza o contínuo de matéria linguística e o estrutura em categorias. Da mesma forma que o já citado químico, que observa as substâncias em termos de suas moléculas formadoras, o usuário observa a interface em termos de categorias que ele reconheça. Por exemplo, a segmentação do contínuo de possibilidades físicas do vídeo poderia ser feita em termos de pixels. Mas pixel é uma categoria que não faz sentido para o usuário, uma vez que ele não distingue o pixel visualmente [Rogers 95]. O pixel faz sentido para programadores de aplicativos gráficos básicos, que trabalham com algoritmos de rasterização de imagem.
Segundo Jackendoff, podem existir estruturas particulares ligadas a tarefas. De acordo com estudos anteriores, em Martins et alii [94, 95], no processo de travessia do golfo de comunicação (Seção 2.1) a intenção do usuário se estabelece em função da tarefa. O usuário vê a interface e formula intenções e soluções sobre ela, em função de seu modelo mental de tarefa. Portanto, a tarefa define as regras preferenciais através das quais o usuário irá perceber a interface. Neste caso, a segmentação do contínuo expressivo do vídeo em termos de primitivas de traçado, passa a fazer sentido para o usuário em função das tarefas que irá executar. De forma semelhante, a segmentação do contínuo de conteúdo deverá ser feita em termos de categorias que o usuário reconheça. A segmentação coerente com as estruturas conceituais do usuário em torno da tarefa facilitará sua compreensão da LV de interface. As estruturas conceituais em torno da tarefa atuam como ativadores da função sígnica.

Sobre o Uso de Metáforas

Segundo Lakoff & Johnson (80) o ser humano raciocina metaforicamente, constantemente transportando conhecimento de um cenário conhecido para outro desconhecido. O discurso em linguagem natural é todo permeado de metáforas. Através das expressões metafóricas, o indivíduo transporta uma estrutura com suas regras de formação e, eventualmente, suas regras de inferência, de uma situação para outra. Por exemplo, através da expressão "Tempo é dinheiro" Lakoff & Johnson mostram como a expressão linguística mapeia não apenas um conceito, mas uma rede de conceitos como: perder tempo, economizar horas, investir um tempo e reservar um tempo. Há também outras expressões onde o tempo se caracteriza como algo que não pode ser desperdiçado. No entanto, nem sempre toda a rede de conceitos pode ser plenamente transportada de uma situação para outra.
Na aplicação de metáforas ao projeto de interfaces, podem ser transportadas a funcionalidade, as características, as propriedades e etc, do objeto no mundo real para o objeto no ambiente computacional [Erickson 90]. Por exemplo, o ícone da lata de lixo permite um mapeamento para a funcionalidade e as propriedades da mesma. Uma das propriedades da lata de lixo é que você pode recuperar o objeto nela colocado. Se ao invés da lata de lixo fosse usado um triturador esta possibilidade deixaria de existir.
A metáfora pode ser total ou parcial. Hallaz & Moran (82) enfatizam a questão de que é muito difícil fazer uma analogia completa entre dois sistemas, especialmente quando se pretende uma metáfora entre o ambiente humano e o sistema computacional. O que se obtém são metáforas parciais que explicam determinados aspectos do sistema computacional. Se metáforas parciais estiverem sendo utilizadas para facilitar ao usuário a compreensão do sistema, elas devem ser tratadas em conjunto, com relação ao modelo conceitual do sistema. Uma metáfora de grande amplitude dificilmente poderá ser transportada na íntegra para o ambiente computacional. Por exemplo, a metáfora do editor de textos como uma máquina de escrever. Em vários aspectos o procedimento de uso do editor de textos assemelha-se ao da máquina de escrever, mas em alguns falha. Por exemplo, no editor de textos, quando o usuário ativa um retorno ele apaga o caracter digitado, na máquina de escrever isso não acontece. Porém, observa-se que uma vez que o usuário tenha tomado conhecimento das diferenças entre os dois sistemas, o antigo e o novo, ele pode desenvolver um novo modelo mental, adequado [Preece et alii 94].
Sob o enfoque cognitivo a metáfora é bastante valiosa no sentido de que traz para o usuário conceitos conhecidos, sem ter de explicitá-los um a um. A metáfora ativa a compreensão do usuário, através do reconhecimento de uma situação conhecida. "*Instead of reducing the absolute complexity of an interface, this approach seeks to increase the initial familiarity of actions, procedures and concepts by making them similar to actions, procedures and concepts that are already know.*" [Carroll et alii 88, p.67]. Em termos do código linguístico, o uso da metáfora torna a linguagem potencialmente mais eficaz. Ela funciona como elemento de ativação da função sígnica. Na Seção 3.3 serão comentadas algumas especificidades sobre o uso de metáforas na LV.

A linguagem visual exige um tratamento diferenciado daquele normalmente adotado para a linguagem textual. No que se refere ao processo de produção linguística, o signo visual não tem a mesma estabilidade do signo textual, no sentido que, teoricamente, o projetista pode optar por um sem número de soluções diferentes quanto à forma de expressão visual adotada. No que se refere ao processo de interpretação linguística, se adequadamente utilizado o recurso visual pode oferecer vantagens expressivas sobre o textual, como por exemplo concisão de conteúdo expresso e facilidade de identificação por parte do usuário. Em ambiente computacional os recursos visuais são restritos. Se por um lado este fato limita as condições do projetista, por outro, viabiliza o tratamento sistemático no uso desses recursos. Neste capítulo é apresentada uma análise sobre o uso dos recursos visuais, como recursos expressivos, em ambiente computacional. Através de uma abordagem semiótica à matéria linguística são avaliadas as características do signo visual, além das peculiaridades da linguagem visual, considerando aspectos lexicais, gramaticais e dialógicos.

3.1. Tratamento Semiótico da Matéria Linguística

A teoria da produção de signos de Eco (76) trata elementos da expressão e elementos do conteúdo em termos de sistemas de códigos, estruturados a partir de elementos discretizados de um plano da expressão e de um plano do conteúdo (veja-se Figura 3.1). Vamos esclarecer este processo tomando por base o plano da expressão, embora ele seja válido também para o plano do conteúdo. O plano da expressão é um " ...*continuum of physical possibilities* ..." [Eco 76, p. 50]. Sobre o plano da expressão é executado um processo de segmentação, que discretiza os elementos de expressão que formarão a matéria linguística da comunicação. O processo de discretização de um contínuo qualquer é feito de acordo com o conhecimento e as expectativas dos elementos humanos envolvidos. Por exemplo, o contínuo expressivo em ambiente computacional é o contínuo da matéria expressiva potencialmente produzida pelos dispositivos de entrada e saída. A segmentação do contínuo expressivo de um vídeo é feita de acordo com a capacidade humana de perceber e tratar os elementos visuais sinalizados por pixels, por combinações de pixels, por combinações de combinações de pixels, e assim por diante. Por exemplo, a letra B de bold (barra de ferramentas de editor de textos) é uma combinação sistemática de pixels. O  (botão de bold para negritar texto) é uma combinação de B e . A barra de ferramentas é uma combinação de combinação de pixels.

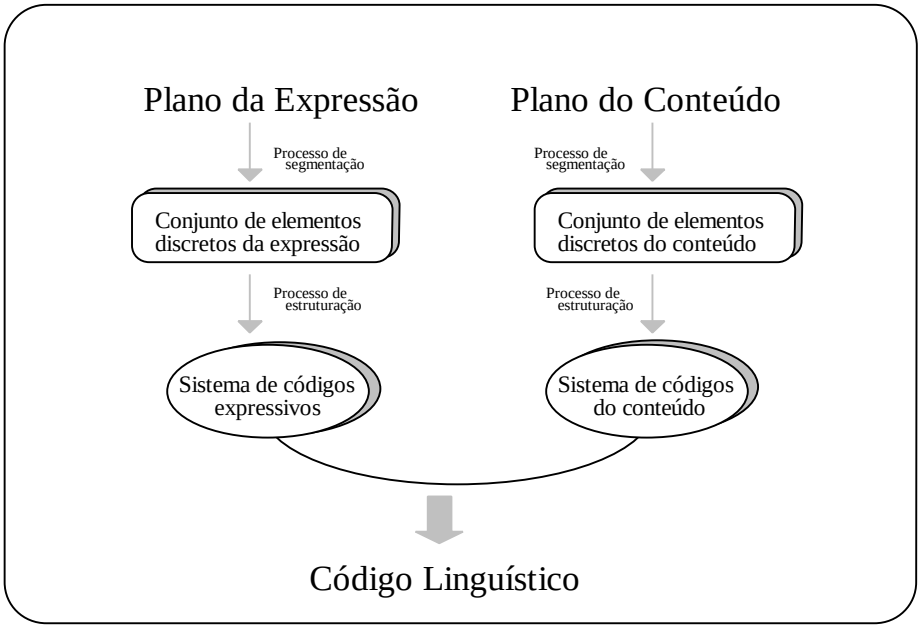


Figura 3.1. Processo de geração do código linguístico

Os elementos discretos são estruturados em sistemas de códigos. O sistema de códigos tem uma gramática interna que estabelece uma estrutura, a qual permite que seja compreeensível e comparável a outros sistemas, além de permitir que sistemas sejam correlacionados entre si. Na linguagem de comunicação a correlação associa um sistema de códigos de conteúdo a um sistema de códigos da expressão, conforme Figura 3.1. Segundo Eco, o código estabelece os tipos gerais da linguagem, ele provê a regra de geração das instâncias da linguagem na interação comunicativa - os signos. Em continuação ao exemplo do parágrafo acima, "B" remete ao conteúdo "bold" ou negrito; "▢" remete ao conteúdo "acionar"; "[B]" remete ao conteúdo "acionar dispositivo para tornar [texto] negrito". O código não gera signos, ele provê regras para geração de signos. Quando um elemento da expressão se correlaciona a um elemento do conteúdo, ambos tornando-se funtivos da correlação, ocorre a função sígnica. Os signos serão as ocorrências concretas do código na presença do elemento humano.

3.2. O Alfabetismo Visual

> "*Hay más de 3.000 lenguas, independientes y distintas, que se usan hoy en el mundo. La universalidad del lenguaje de la visión es comparativamente tan superior que parece realmente rentable superar la dificultad que pueda suponer su complejidad. Los lenguajes son conjuntos lógicos. Pero ninguna sencillez de este tipo puede adscribirse a la comprensión visual, y los que hemos intentado estabelecer una analogia con el lenguaje conocemos muy bien la futilidad de este intento*" [Dondis 73, p. 22].
>
> "*Another obstacle remained in the presupposition that the basic units of visual language had to possess similar characters to those of verbal language. They ought to be simple, autonomous, isolated and independent, constituting a finite set. Obviously, the categories of elements composing visual language are quite numerous and each category offers an almost infinite possibility of proliferation.*" [Saint-Martin 90, p. 3].

As citações acima são ilustrativas das dificuldades de se definir um alfabetismo visual em condições genéricas. O alfabetismo visual se refere a uma decomposição do meio de expressão, em partes componentes e estruturas. Na LV em ambiente humano encontramos dificuldade em definir os elementos básicos de composição, dada a grande variedade de elementos e de meios expressivos, além de dificuldades em definir a imensa variedade de composições possíveis. Por exemplo, a LV pode ser analisada na arquitetura, no desenho, na pintura e outros [Dondis 73], cada um deles tem elementos básicos de composição próprios, a partir dos quais é possível definir uma grande variedade de composições.

Definir os elementos básicos significa segmentar o contínuo expressivo da linguagem em cada situação de uso particular. Por exemplo, uma LV utilizada para comunicação visual é diferente de uma LV artística. A tarefa de discretização pode ser facilitada por abordagens específicas para determinados tipos de LVs. Sobre as dificuldades e a necessidade de se estabelecer o alfabetismo visual, encontramos em Dondis uma posição bastante esclarecedora:

> "*El enjuiciamento de lo que es factible, apropriado, o efectivo en la comunicación visual se ha abandonado en favor de definiciones amorfas del gusto o de la evaluación subjetiva y autorreflexiva del emisor y el receptor sin apenas intentar comprender, al menos, algunos niveles prescritos de lo que llamamos alfabetidad en el modo verbal. Probablemente, esto no se debe tanto a un prejuicio como a la firme convicción de que es imposible el empleo de cualquier metodología*

o cualquier medio para alcanzar la alfabetidad visual. ... Frente al desafio de la alfabetidad visual no podemos seguir encogiéndonos de hombros durante mucho tiempo." [Dondis 73, p. 23].

Sendo a interface um artefato para troca de mensagens, com o objetivo claro de comunicação do conteúdo de sistema, a definição do alfabetismo visual é fundamental. "*Las diferencias fundamentales entre la aproximación intuitiva y direta y la alfabetidad visual están en el nivel de correspondencia y fidelidad entre el mensaje codificado y el mensaje recibido.*" [Dondis 73, p. 51]. O alfabetismo visual não garante o nível de exatidão entre as mensagens codificada e a recebida, mas favorece a aproximação entre os interpretantes dos agentes emissor e receptor. O alfabetismo da linguagem transforma ocorrências visuais casuais em ocorrências metódicas. A LV passa a ser um índice do princípio de projeto, onde índice é uma representação relacionada por contiguidade física ou causalidade com aquilo que representa [Peirce 31]. Quando o usuário captura o índice, ele captura a regra operante por trás. A regra operante é o mecanismo que favorece a aproximação entre o que foi codificado e o que será compreendido.

O fato de a LV para interfaces ser restrita ao modelo semântico da aplicação viabiliza o uso organizado dos recursos visuais. Como foi dito no início deste capítulo, no ambiente computacional os elementos de ativação da interface são mapeados através de diferentes camadas do sistema, até o ambiente de execução da máquina. Toda a representação interna do computador é feita em linguagem textual, porque esta é a sua linguagem de modelização primária. Portanto, qualquer linguagem utilizada na interface (com recursos visuais ou outros) será obrigatoriamente ancorada em uma linguagem textual (a linguagem de especificação, de programação, de implementação), porque a tradução sempre será necessária. Portanto, a LV deve ter uma estrutura (sintaxe e etc) mapeável sobre a linguagem textual. Estes fatores estabelecem uma severa restrição para as LVs, tornando-as motivadamente processáveis, gramaticalizáveis e, portanto, levando à possível definição de um alfabetismo visual.

3.3. O Signo Visual

O sistema de significação da linguagem utilizada para a comunicação verbal humana correlaciona elementos abstratos. A linguagem natural é essencialmente simbólica. A correlação entre os sistemas de códigos da expressão e do conteúdo é socialmente estabelecida e socialmente reconhecida. Sendo o plano da expressão simbólico, as funções que correlacionam instâncias da expressão a instâncias do conteúdo são estáveis. Por exemplo, o conceito abstrato de cadeira é correlacionado à expressão simbólica "cadeira" de maneira estável, na medida em que esta correlação é socialmente reconhecida. Na linguagem natural a convenção é forte, o sistema de significação da linguagem é estável, tanto no que se refere à produção de mensagens, quanto à interpretação.

Nos SMG a LV não remete sempre a conceitos abstratos, ao contrário, ela alude frequentemente à realidade concreta. Conforme ilustrado na Figura 3.2, cada representação visual do conceito abstrato "cadeira" é a representação de uma cadeira específica, ou seja, é uma instância possível do conceito "cadeira". O nível de semelhança da imagem com a realidade depende de vários fatores, entre eles as condições de produção da mesma. A imagem pode variar do nível realista ao esquemático, ou ao simbólico, em diferentes gradações. Em função disto, diferentes codificações são possíveis, associando-se diferentes instâncias do plano da expressão à mesma instância do plano do conteúdo.

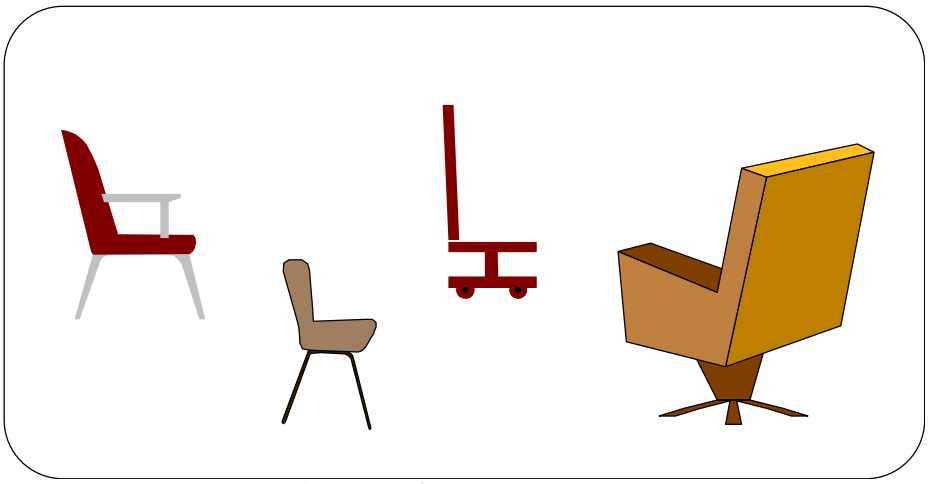


Figura 3.2. Diferentes instâncias visuais do conceito "cadeira"

Neste sentido dizemos que, na produção de mensagens visuais, embora a relação entre significante e significado se estabeleça, ela não tem a mesma estabilidade que na linguagem textual. Essa instabilidade ocorre no processo de produção das mensagens, quando o projetista pode codificar a mensagem de acordo com seu próprio interpretante. No processo de interpretação a relação entre significante e significado é mais estável, no sentido de o usuário reconhecer o signo. Por exemplo, que instância de representação visual de uma cadeira será adotada por um projetista é uma incógnita, mas diferentes instâncias de representação visual de uma cadeira podem ser rapidamente reconhecidas por um usuário. A probablidade de convergir na interpretação é maior do que na produção de signos visuais. Uma ressalva deve ser feita para as linguagens diagramáticas, sistemas de código de

trânsito e outras, que são linguagens simbólicas, cuja estabilidade e cujos processos de produção e interpretação de mensagens são semelhantes à linguagem textual.
Esta questão pode ser traduzida por uma dificuldade em estabelecer o item lexical na LV. Por exemplo, em linguagem natural distinguimos: a palavra escrita, a palavra pronunciada e o item lexical. A palavra pronunciada é tão variável quanto as diferentes instâncias de cadeira acima apresentadas, o mesmo acontecendo à palavra escrita (ou grafada). Basta pensar em diferentes sotaques, timbres vocais, e em diferentes fontes tipográficas e etc. No entanto, na linguagem natural o item lexical "cadeira", uma abstração provida pelo sistema linguístico, é estável e remete a um sub-conjunto estável de significações compartilhado pelos falantes de português. Esta instância de representação abstraída e idealizada, como na linguagem natural, não existe na LV. O que então constitui um item lexical no alfabetismo visual? No alfabetismo visual o item lexical é dinâmico, a função de correlação entre significante e significado é circunstancial. A Figura 3.3 ilustra a relação entre significante e significado em ambas as linguagens.
Dois aspectos podem ser observados através das linguagens visual e textual. Um deles, a pluralidade de representações da LV diminui sua eficácia de expressão quando comparada à linguagem textual. Na linguagem textual utilizada nas interfaces busca-se sempre uma similaridade com a linguagem natural. O resultado é a utilização de um sistema de significação previamente dominado pelo usuário. Por exemplo, uma operação de apagamento de texto será nomeada como "apagar", uma operação de fechamento de arquivo será nomeada como "fechar". A relação de significação entre o conteúdo (ação do sistema) e a expressão (comando) é culturalmente motivada. A interface textual desfruta de um contexto cultural pré-estabelecido que facilita o reconhecimento do signo por parte do usuário. Por exemplo, diferente LVs podem ser especificadas pelo projetista para os SMG. O sistema semiótico adotado pelo projetista pode ou não ser compreendido pelo usuário. Isto é mais difícil de acontecer em interfaces textuais, uma vez que o sistema semiótico já está pré-estabelecido.

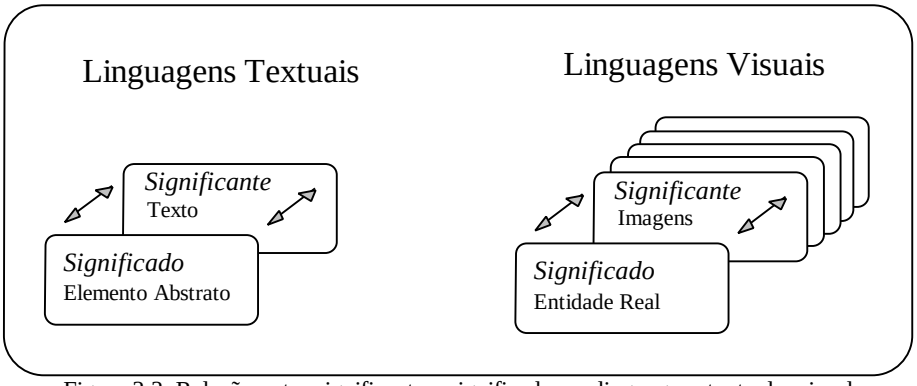


Figura 3.3. Relação entre significante e significado nas linguagens textual e visual

Por outro lado, o signo visual pode ser mais eficaz que o textual, porque o primeiro permite restringir as interpretações do usuário. Stenning & Oberlander (95) em sua proposta teórica sobre raciocínio gráfico e linguístico argumentam que as representações gráficas, pela especificidade da forma, limitam as abstrações e auxiliam a processabilidade. Segundo eles a diferença entre a representação gráfica e a representação linguística está na especificidade da primeira com relação à capacidade de abstrações da última. A linguagem gráfica permite criar sistemas de representação com diferentes graus de abstração. Eles classificam três tipos de sistemas de representação abstrata: de abstração mínima, de abstração limitada e de abstração ilimitada. O primeiro permite que apenas um modelo semântico seja construído para cada representação no sistema. O segundo permite a construção de um número limitado de modelos semânticos, porque deixa algumas características do modelo semântico em aberto. O último permite uma quantidade virtualmente ilimitada de modelos semânticos. Através de um sistema de representação de abstração mínima o projetista pode reduzir a capacidade de interpretações do usuário. Neste caso, embora a linguagem textual desfrute de um contexto pré-estabelecido, a LV pode ser mais específica, e mais eficiente.

O Processo Metonímico no Signo Visual

Embora na linguagem textual de interface sempre se busque uma similaridade com a linguagem natural, muitas vezes esssa similaridade é falsa. Por exemplo, a noção de arquivo em português é diferente da noção de arquivo em ambiente computacional, este referente a um arquivo de texto, arquivo binário e outros. O uso da palavra arquivo para arquivo de dados é, mesmo na linguagem natural, um processo de significação que estende ou metaforiza o conceito estabelecido de arquivo (não computacional). Em linguagem natural este processo de extensão ocorre sem problemas, porque usos metafóricos e metonímicos [Lakoff & Johnson 80] são absolutamente comuns.
A reprodução deste processo analógico em LV não é tão simples. O signo visual não possui a mesma "elasticidade" expressiva da linguagem textual. Por exemplo as palavras On/Off servem perfeitamente para ligar/desligar tanto uma unidade de áudio como as maiúsculas de um teclado. Já o desenho de um interruptor

serviria perfeitamente para ligar/desligar a unidade de áudio, mas não serviria para ligar/desligar as maiúsculas do teclado.

3.4. Especificidades da Linguagem Visual

Previamente à análise das particularidades da LV, vale a pena um esclarecimento sobre diferentes formas de utilização do recurso visual. Por LV entendemos o uso do recurso visual como recurso expressivo. Frequentemente, em interfaces gráficas, observamos o uso estético do recurso visual simultaneamente ao uso como recurso expressivo. Em princípio, o uso estético estaria fora da linguagem visual, no sentido de que esta deve expressar o comportamento do objeto gráfico. Porém, para efeito de controle de uso dos recursos visuais expressivos, aqueles recursos utilizados com finalidades estéticas podem ser registrados como tal na especificação da LV. Isto garante ao projetista de interfaces que o mesmo recurso não seja reutilizado como recurso expressivo, causando uma dubiedade para o usuário. Por exemplo, um fundo de tela na cor cinza claro, utilizado com finalidade estética. Se, na especificação da LV, for registrada a associação da cor cinza claro a um conteúdo "estético", ficará evidenciado para o projetista a utilização da cor, podendo ser evitada sua reutilização para expressar conteúdo do sistema.

A linguagem visual objeto deste trabalho é aquela que permite ao usuário "ver" e "operar sobre o que vê", uma representação, na tela do computador, do objeto abstrato que está sendo tratado no universo computacional. Para que a LV seja eficaz ela deve se aproximar o máximo possível do objetivo de expressar "tudo e só" o que constitui a semântica do objeto graficamente representado. Basicamente, o objeto deve parecer aquilo que é, ou ainda, ter um comportamento coerente com aquilo que sugere visualmente. A imagem não deve sugerir ao usuário um modelo conceitual que não seja verdadeiro no ambiente do sistema. A LV deve orientar o usuário, evitando que ele se frustre na tentativa de executar operações incorretas. A produtividade do usuário cresce na medida inversa de sua carga cognitiva, ou seja, na medida inversa do seu esforço para compreender, aprender e reter informação sobre o sistema [Norman 86].

A consistência da linguagem é um dos objetivos a serem perseguidos na busca por eficiência. Segundo Payne & Green (86), a consistência é difícil de ser definida, mas o sentido básico de consistência na linguagem pode ser identificado como regularidade. Grudin (89) argumenta que nem sempre a consistência absoluta é uma boa solução. Ele demonstra que é difícil obter um critério de consistência que atenda a todas as situações de forma satisfatória. A determinação de um critério rígido de consistência traz vantagens como a capacidade de inferência do usuário, mas em situações pontuais pode ser prejudicial. Ele recomenda que a interface seja sempre avaliada com relação à tarefa do usuário, sendo permitidas pequenas quebras de consistência onde elas possam trazer um resultado positivo. Na LV a regularidade do código é particularmente importante, uma vez que a linguagem é específica de cada aplicativo, ou seja, em tese, cada interface pode utilizar uma linguagem particular. O uso sistemático dos recuros visuais, ao contrário do uso ocasional, facilita a compreensão do usuário, permitindo a inferência do comportamento do sistema em situações similares. "*... one part of the language prompts expectations about the remainder.*" [Payne & Green 86, p. 96]. Quanto mais o código linguístico for provido de regras geradoras de tipos e quanto menos soluções ocasionais ele contiver, mais eficaz será.

Sobre o Léxico do Código Visual

Como já dissemos, a LV não pode ser tratada pelos mesmos referenciais da linguagem textual. Algumas peculiaridades devem ser observadas, uma vez que afetam a potencialidade do código expressivo. Duas características importantes são: limitação dos recursos expressivos e sobreposição de recursos expressivos no código visual.

Em LV para interfaces os recursos expressivos são discretizados do contínuo de possibilidades físicas dos dispositivos de comunicação. Por exemplo, para comunicação do usuário estão disponíveis os sinais do mouse e para comunicação do sistema os recursos de vídeo, como forma, cor, textura e brilho. Com relação aos recursos de vídeo algumas restrições ainda devem ser observadas. Por exemplo, linhas excessivamente finas, cores muito claras ou um conjunto de cores muito próximas entre si, podem não ser distinguidas pelo usuário, devendo ser descartados como elementos expressivos [Andersen 90]. Portanto, são poucos os recursos expressivos restantes.

Outro aspecto restritivo no que se refere aos recursos expressivos é a sobreposição de recursos sobre a forma. Os recursos expressivos disponíveis, como por exemplo cor e textura, aplicam-se sobre o recurso básico que é a forma. A forma é o elemento que dá existência ao objeto gráfico. Mesmo que a forma não seja distintiva, ou seja, mesmo que ela não expresse nenhum conteúdo, ela é necessária para que outros recursos se concretizem. Por exemplo, um caso onde diferentes substâncias químicas são visualmente representadas através de círculos e a distinção de substância é feita pela cor dos círculos. Neste caso a forma é um recurso neutro e a cor é o recurso de distinção.

Na linguagem textual pode-se expressar várias informações a respeito de um objeto, referenciando-se este objeto através da linguagem. Por exemplo, através do discurso linguístico pode-se associar um número indefinidamente grande de atributos a uma pessoa. Pode-se dizer que ela é magra, morena, agitada, alegre e etc. Na LV por MD as informações sobre um objeto não são uma predicação linguística, mas representações sobre sua forma. As propriedades que caracterizam o objeto são representadas sobre ele, por exemplo uma cor caracterizando um estado ou uma textura caracterizando um material. Existem limites para a aplicação de recursos como cor, textura e brilho sobre uma mesma forma, em função da interferência entre eles. Por exemplo, recursos semelhantes não

podem ser usados simultaneamente. Embora haja uma grande quantidade de cores disponíveis em um vídeo colorido, apenas uma cor pode ser utilizada de cada vez para expressar informações, porque não é possível sobrepor duas cores ao mesmo objeto simultaneamente. Outro exemplo, recursos diferentes podem causar interferência mútua. A utilização da cor amarela para expressar uma propriedade e de textura pontilhada para expressar um material, ambas sobre o mesmo objeto, pode confundir o usuário, que pode entender a textura amarela como um código único, não distinguindo as duas informações de conteúdo.
Os aspectos apresentados caracterizam a LV como uma linguagem bastante pobre. Estas questões serão novamente tratadas na Seção 4.3.2, onde será feita a análise para discretização dos recursos expressivos do sistema.

Aspectos da Gramaticalidade do Código Visual

Outro aspecto específico do código visual é relativo à sua capacidade de articulação. Como foi dito (Seção 2.5), Saint-Martin (90) aborda a percepção visual pelo aspecto físico. Ela observa a formação da linguagem visual a partir dos coloremas. Os coloremas são instâncias físicas do campo visual humano. Segundo ela, na LV os elementos básicos (coloremas) são agrupados através de certas leis e com o auxílio de operações perceptivas, que são influenciadas pela estrutura e organização de cada campo visual em particular. No ambiente computacional, no caso do vídeo, existe uma estrutura física de justaposição de pixels que permite a formação de imagens. Esta composição de imagens obedece regras sintáticas de forma que, por exemplo, um ponto visível no vídeo é uma composição que pode se repetir com regularidade, sempre que as mesmas regras forem aplicadas sobre os mesmos elementos básicos. A partir da imagem do ponto novas regras podem ser aplicadas de forma a gerar novas composições.
Em nossa análise estamos desconsiderando este nível de gramaticalização, ou de agregação física da imagem. Nosso contínuo expressivo será segmentado em um nível mais alto de estruturação, no qual os elementos discretos expressivos são elementos básicos de imagens já compostas, como: forma, cor e textura. De acordo com a teoria semiótica esta abordagem seletiva é válida, uma vez que os contínuos expressivos e de conteúdo serão segmentados de acordo com o conhecimento e a necessidade dos usuários da linguagem de comunicação [Eco 76].
Partindo dos elementos discretos citados, constatamos que a LV é pobre em articulação. Em linguagens articuladas cada nível de articulação contém potencialmente muitos conteúdos semânticos de nível superior ao lexical. Na LV não encontramos articulação de elementos básicos do léxico, de forma a compor unidades semânticas de mais alto nível. Por exemplo, na linguagem textual a partir do alfabeto verifica-se uma articulação de fonemas em lexemas, palavras, frase e etc. Na LV não encontramos esses níveis de articulação, não existe articulação nem mesmo a partir dos elemento básicos visuais, as formas compostas são primitivas. Não existe, por exemplo, agregação entre linha e cor para compor um novo elemento lexical. O léxico da linguagem se compõe das primitivas gráficas. A agregação das primitivas será feita de acordo com a organização física do objeto real, traduzida em regras geométricas e topológicas na modelagem. No léxico da linguagem também podem estar presentes formas compostas, como formas primitivas. Por exemplo, o desenho de uma ampulheta que pode ser usado para sinalizar um estado de execução do sistema. Segundo Eco (76), os chamados signos icônicos podem ser considerados textos visuais, não analisáveis ulteriormente. Ou seja, são textos visuais cujos elementos articuladores são indiscerníveis.
Como consequência, a relação entre elementos da expressão e do conteúdo será de praticamente um para um. Pela baixa articulação, cada turno tipicamente conterá um (ou muito poucos) itens lexicais, ou unidades semânticas de mais baixo nível. Ou seja, a utilização da linguagem se faz praticamente sobre os itens lexicais básicos.

Sobre a Estrutura de Diálogo
No diálogo entre pessoas, a conversação é segmentada em unidades tópicas de discurso, também chamadas unidades tópicas de conversação, em torno de um determinado foco. A conversa começa, continua iterativamente e termina, para um determinado tópico, ao longo do qual existe um controle de alternância entre os agentes da comunicação, em torno da idéia de "de quem é a vez". Na interação usuário-sistema o diálogo ocorre de forma semelhante. Por exemplo, na comunicação usuário-sistema através de linguagem natural encontramos uma segmentação da conversação e uma alternância de agentes semelhante à conversação humana, com a restrição de que os agentes da comunicação não compartilham o mesmo contexto [García 95]. Nestas interfaces o usuário formula uma mensagem com base em seu modelo mental da tarefa, envia a mensagem para o sistema e "passa a vez"; o sistema processa a mensagem em um ambiente formal, referente ao modelo conceitual implementado pelo projetista, envia um retorno ao usuário e "passa a vez".
Segundo Brennan (90), neste processo de diálogo alguns aspectos da MD a assemelha mais à conversação humana. Ela argumenta que não existe dicotomia entre a MD e a conversação humana, apontando importantes fatores do contexto dialógico humano que estão presentes na MD. Entre estes fatores, por exemplo, está a representação contínua do objeto de interesse, que na conversação real corresponde a um modelo mental compartilhado pelos agentes da comunicação. Na MD existe um retorno imediato à ativação, sobre o próprio objeto. " *... because the feedback was so timely and relevant, it could be considered analogous to backchannels, ... in human/human comnication. ... a hearer reacts to a speaker with eye contact, nods, and um hmm's.*" [Brennan 90, p. 395]. Neste ambiente não existe propriamente uma alternância de agentes, existem procedimentos de ação/reação quase simultâneos.
Na LV a unidade tópica da conversação se desenvolve em torno da tarefa [Martins, Souza & Gattass 94, 95]. Ela é composta por toda manipulação referente a um conjunto de objetos. A seleção de um objeto pode ser identificada

como o protocolo de abertura das conversações. Após a seleção, diferentes tarefas podem ser executadas, interativamente, até o encerramento, que pode ser definido pela seleção de outro objeto. A unidade tópica de conversação caracteriza-se por um contexto de pré-condições, condições de operação e sinalizadores de retorno. Por exemplo, suponha-se um objeto selecionado, onde a seleção seja sinalizada pela cor. Quando uma unidade de conversação é aberta, o objeto selecionado tem sua cor original alterada para a cor que expressa a seleção. Supondo que a cor original deste objeto significasse, por exemplo, uma propriedade, este recurso de expressão estaria provisoriamente inativo e a informação representada estaria mascarada. A informação original voltaria a ser expressa somente ao final da unidade de conversação, quando a seleção fosse desfeita. Nesta unidade de conversação a expressão visual do objeto teria sido alterada. Ainda segundo Brennan, a MD permite que as referências sejam feitas por apontamento. Portanto, dentro da unidade tópica de discurso o apontamento é o equivalente às referências anafóricas, uma vez que toda a manipulação é feita sobre o objeto.

Níveis de Verificação de Consistência na LV

Diferentes níveis de consistência podem ser avaliados em uma linguagem. Payne & Green (86), na proposta da TAG apontam o nível de consistência sintático-semântica e semântica. O primeiro refere-se à forma de expressão, ou seja, conteúdos semelhantes devem ser expressos de forma semelhante. Por exemplo, em linguagem textual de interface o comando "copiar arquivo" tem uma forma sintática semelhante ao comando "apagar arquivo". São operações da mesma natureza semântica, expressas de forma semelhante. Ambas obedecem à sintaxe <operação/objeto>. Em LV por MD o equivalente seria: se a operação de cópia fosse expressa na forma <selecionar-objeto/copiar-objeto>, a operação de movimentação deveria ser expressa como <selecionar-objeto/movimentar-objeto>. Este tipo de consistência pode ser verificado nas linguagens textuais e visuais, de forma semelhante. A consistência semântica está mais a nível de comportamento do sistema, do que propriamente da linguagem de expressão. No que se refere a LVs, supondo que a consistência semântica exista, a questão é expressá-la corretamente. Neste sentido, a consistência lexical é o ponto chave. Como foi dito acima, um léxico definido de forma sistemática é uma ferramenta de expressão eficaz.

Os níveis de consistência lexical, sintático e semântico são comuns a qualquer linguagem de interface. No que se refere às LVs, a exceção é que o léxico é particular a cada linguagem, e como tal necessita de mecanismos específicos de verificação de consistência. Nas linguagens textuais, embora o projetista tenha liberdade, busca-se sempre a utilização de um léxico próximo da linguagem natural, no sentido de aproveitar o conhecimento do usuário. Este léxico textual irá desfrutar da estrutura e regularidade da linguagem natural.

A LV também exige a averiguação de um quarto nível de consistência não existente em linguagens textuais - a consistência a nível de discurso. Neste caso, a verificação do nível lexical ao semântico mostra-se insuficiente, porque em cada nível a expressão do sistema é tratada de forma independente da expressão do usuário. Como foi apontado acima, na LV por MD as informações sobre o objeto são sobrepostas à sua forma, seus atributos, propriedades e estados, e portanto ocorre sobreposição de recursos expressivos visuais. Esta sobreposição não só pode ocorrer a nível da expressão do sistema, como pode ocorrer na unidade de discurso, na interação com o usuário. A sobreposição de recursos pode causar conflitos, inclusive com anulação mutua. Portanto, é necessária uma avaliação do uso dos recursos expressivos no diálogo, de forma a averiguar se a expressão conjunta de usuário e sistema não está causando sobreposição de recursos.

Motivações para o Uso de Linguagens Visuais

Em resumo, algumas das características que observamos para a LV:

a) Limitação de recursos expressivos - Os recursos visuais expressivos são limitados e concorrem na sua especificação sobre a forma, podendo causar interferência mútua.

b) Falta de convenção a nível de signos simples - A linguagem não oferece um alfabeto a partir do qual se possa operar. O que se observa eventualmente é o uso de ícones e símbolos como se fossem palavras, como unidades encerradas em si, não decomponíveis e não utilizáveis para compor símbolos ou ícones mais complexos. Por exemplo, a maioria dos sistemas gráficos opera a partir de primitivas, como arestas, retângulos e círculos, definidas por conveniência dos pacotes gráficos. O retângulo não é tratado como uma composição de um signo mais simples, mas ele próprio como uma unidade elementar.

c) Falta de uma gramática a nível dos signos complexos - Não existe uma gramática para articulação de signos complexos.

d) Níveis de averiguação de consistência - A LV necessita averiguação de consistência a nível do léxico, a nível do diálogo e a nível de diferentes códigos interativos sobre a mesma representação visual.

Apesar das restrições, existem vantagens na utilização de LVs.

A grande motivação para o uso de LVs em interfaces usuário-sistema está em qualidades como: (a) economia de espaço, (b) independência de processamento linguístico e (c) a brevidade da expressão de ações sobre objetos. Quando utilizada de forma adequada, a LV é muito mais concisa do que a textual. A MD da representação visual também, se implementada adequadamente, permite ações rápidas e intuitivas do usuário sobre o objeto.

3.5. Contribuições Especiais da Teoria Semiótica para as Linguagens Visuais

Considerando as especificidades da LV, três aspectos da abordagem semiótica são interessantes à nossa pesquisa: (i) uma abordagem ampla dos fenômenos de significação, ou uma meta-teoria; (ii) a não distinção entre semântica e pragmática e (iii) o enfoque do processo de significação como algo dinâmico.

Quanto ao primeiro aspecto, a investigação da interação homem-máquina é uma possibilidade extremamente abrangente, pode ser abordada sob vários aspectos, entre eles podemos citar o hardware interativo, o estilo do software interativo, a compatibilidade entre modelos de tarefas, o impacto organizacional da informatização de processos e outros [Booth 89]. Cada um destes enfoques lança mão de bases teóricas desenvolvidas em diferentes áreas da Ciência, como: matemática, linguística computacional, aspectos cognitivos, psicologia social, psicologia organizacional, sociologia e outros. O tratamento da interface enquanto linguagem de comunicação pede, de fato, a consideração de aspectos das várias disciplinas ligadas ao processo de comunicação em sociedade. Porém quando o pesquisador lança mão de uma teoria específica, normalmente ele restringe seu trabalho a uma determinada orientação, perdendo a visão macro do processo de interação. A semiótica, vendo cada disciplina como um conjunto de sistemas de códigos de significação, permite abordar o processo de significação global como um sistema de correlação entre os códigos que caracterizam as várias disciplinas. Como foi dito, o código provê regras para gerar o signo, entre estas regras podem estar componentes cognitivos, psicológicos, sociais, linguísticos e outros. Este enfoque permite incorporar ao código linguístico a cultura estabelecida na área. Esta visão caracteriza uma meta-teoria.

A abordagem semiótica permite tratar as linguagens naturais e formais sob um mesmo arcabouço teórico. Para nós esta definição é bastante confortável, porque, como foi apontado no processo de modelagem, ambos os ambientes (natural e formal) estão presentes no processo de comunicação. Nestes ambientes, conteúdo e expressão foram discretizados de forma diferente, atendendo a condições específicas. Mas ambos mantém as mesmas características estruturais, podendo, por isto, ser tratados da mesma forma. É possível tratar a linguagem de interação usuário-sistema de maneira sistemática, englobando tanto os aspectos do usuário quanto os do sistema, em um só problema. A semiótica permite uma abordagem única no estudo de uma linguagem de interação que, como dissemos, é as síntese de dois mundos, o ambiente computacional e o ambiente humano.

Quanto ao segundo aspecto, a abrangência proposta pela semiótica, através de uma meta teoria, permite reunir os dois níveis de tratamento de significado tradicionalmente utilizados na linguística. A distinção entre os níveis de significado semântico e pragmático tem sido utilizada com sucesso com finalidades didáticas. Nos sistemas restritos a interfaces textuais, esta distinção didática é especialmente útil, uma vez que a semântica da linguagem é pré-estabelecida, e o contexto de uso é "congelado" na codificação do significado. Nas interfaces com multimodalidade de elementos no s-código, como elementos visuais, sonoros, de movimento e outros, esta abordagem semântica já não dá conta do problema, pois o próprio componente semântico é dinamicamente resolvido em termos de dados contextuais tipicamente tratados na pragmática das linguagens naturais. Nelas, a percepção do usuário é atingida por novos estímulos, não sendo mais possível fazer o congelamento do contexto de uso da linguagem, no significado.

Como já foi citado, em interfaces gráficas de captura de dados o usuário pode atuar no nível sintático (imagem) da representação visual ou no nível semântico (imagem virtual), conforme a tarefa que esteja desenvolvendo. Esta alternância não permite mais uma distinção clara entre significado semântico e pragmático, pedindo uma nova abordagem da idéia de significado.

Quanto ao último aspecto, o enfoque do processo de produção sígnica como algo dinâmico. Segundo Eco "*a sign is not a fixed semiotic entity but rather the meeting ground for independent elements (coming from two different systems of two different planes and meeting on the basis of a coding correlation*" [Eco 76, p. 49]. Este tratamento do signo como resultado de uma correlação que pode ser dinâmica, permite considerar a capacidade de retro-alimentação do sistema de significação, e a adaptação do usuário a este processo.

De acordo com a abordagem semiótica adotada neste trabalho, a especificação da linguagem de interface exige um tratamento prévio dos contínuos do conteúdo e da expressão, segmentando-os e definido os sistemas de códigos, que correlacionados, definirão o código visual. Neste capítulo é apresentada uma análise dos contínuos do conteúdo e da expressão, de usuário e sistema, no ambiente dos SMG. Nesta análise, buscou-se identificar a semântica fundamental de cada recurso expressivo, a partir do que são apresentadas as diretivas de orientação para codificação da linguagem visual (DOCLV).

4.1. Definições Básicas

Conforme foi apresentado no Capítulo 2, a proposta de Norman (86) sobre o golfo de comunicação na interação usuário-sistema distingue os processos de expressão do sistema e do usuário. Diferente da comunicação humana, onde ambos os agentes compartilham os planos da expressão e do conteúdo, na comunicação usuário-sistema os agentes têm planos da expressão e do conteúdo diferentes. No primeiro caso, em função dos dispositivos de entrada e saída, cada um dos agentes tem planos da expressão diferentes. Para expressar ao usuário o conteúdo do sistema, o projetista dispõe, por exemplo, dos recursos visuais de um vídeo. Para expressar suas intenções, ativando as tarefas do sistema, o usuário dispõe comumente dos recursos do mouse e do teclado. Embora a linguagem visual seja formada pelo conjunto destes recursos expressivos, a análise da capacidade expressiva de cada agente da comunicação deve ser feita em separado.

Também diferente da comunicação humana, onde o usuário expressa o conteúdo sobre o qual raciocina, na comunicação usuário-sistema o usuário raciocina sobre o domínio do sistema e expressa conteúdos de ativação, de acordo com a informação oferecida pelo sistema. A expressão do usuário é associada à expressão do sistema. Por exemplo, quando o usuário ativa o sistema através de um item de menu, ele está expressando uma escolha sobre uma informação oferecida pelo sistema. Sua intenção foi formulada em função do domínio do sistema, mas o conteúdo expresso está vinculado à informação oferecida na interface. Portanto, o plano do conteúdo a ser expresso é distinto do plano de raciocíno do usuário.

A segmentação do contínuo expressivo de cada agente deve considerar as características dos dispositivos específicos. Neste trabalho vamos considerar como dispositivos de comunicação um mouse de dois botões e um monitor de vídeo colorido de alta resolução, que são os dispositivos básicos utilizados pela grande maioria dos aplicativos gráficos. A restrição a um equipamento básico facilita a caracterização do problema, mas não invalida o trabalho com relação a dispositivos com funções similares, como por exemplo um mouse com manipulação para três dimensões. A estrutura básica a ser apresentada neste trabalho poderá ser transportada e ajustada para outros ambientes. Esta questão será comentada no Capítulo 8 - Conclusões Finais.

Para cada agente da comunicação serão executadas três etapas na definição do sistema de código expressivo: (a) a segmentação do contínuo expressivo, (b) a análise do potencial expressivo de cada elemento discretizado e (c) a estruturação dos elementos discretos em um sistema de códigos expressivos. Por exemplo, no que se refere às duas primeiras, no processo de segmentação do contínuo expressivo do sistema são discretizados a cor e o brilho. No processo de avaliação observa-se que a cor tem um potencial expressivo muito maior do que o brilho. No que se refere à percepção do usuário a cor tem uma definição superior ao brilho, isto é, em geral a cor é mais perceptível ao olho humano do que o brilho, especialmente se ambos estiverem sendo usados em conjunto. Além disto, o uso do brilho está sujeito a mais restrições do que o uso da cor. Por exemplo, numa superfície grande o uso do brilho causaria desconforto visual ao usuário. No processo analítico poderão ser descartados os recursos cuja capacidade expressiva seja considerada irrelevante para a LV em questão. Vamos buscar os recursos mais eficazes individualmente e as possibilidades de interação entre eles.

Quanto à terceira etapa, o sistema de códigos expressivos é um sistema estruturado, que possui uma gramática interna. Este sistema será posteriormente correlacionado ao sistema de códigos do conteúdo. Por exemplo, na linguagem natural o alfabeto é um conjunto de elementos discretos de representação. O alfabeto é estruturado em morfemas e lexemas, a partir de determinadas regras de composição, chegando até o léxico. O alfabeto e sua estrutura compõem o sistema de códigos expressivos que será associado ao sistema de códigos do conteúdo, gerando o código da linguagem natural.

Procedimento semelhante, nas três etapas, será executado para definição dos sistemas de códigos do conteúdo, para usuário e sistema. A partir da definição dos sistema de códigos serão averiguados os eixos de lexicalização, para usuário e sistema. Por exemplo, o elemento cor se adequa a expressar atributo, porque ele se sobrepõe à forma assim como os atributos se associam aos elementos representados pela forma. Serão apresentadas regras de orientação para definição do código linguístico, ou seja, o código de correlação entre o sistema de códigos do conteúdo e o sistema de códigos da expressão. O código de correlação cria os tipos gerais da linguagem.

Uma importante observação deve ser feita com relação aos critérios de segmentação. Vamos observar que no ambiente computacional os critérios de segmentação dos contínuos expressivos são mais evidentes do que os de segmentação dos contínuos de conteúdos, por se tratar de dispositivos físicos eletrônicos. Por exemplo, o contínuo expressivo do usuário está limitado à capacidade do mouse, e a discretização dos tipos expressivos é relativa aos sinais emitidos pelo mouse. Na segmentação dos contínuos de conteúdo é mais difícil adotar um critério absoluto, existem nuances que às vezes tornam difícil a segmentação. Por exemplo, no caso do usuário, considerando que a interação é feita com base na tarefa, o contínuo de conteúdo será segmentado com base nas intenções do usuário. Ou seja, serão avaliadas as intenções que o usuário pode expressar com relação à tarefa que deseja executar e com relação à interface sobre a qual formula estas intenções. A segmentação apresentada para os contínuos de conteúdo do usuário e do sistema não são absolutas. Casos particulares podem exigir algum tipo de adequação.

As diretivas apresentadas neste trabalho para o código linguístico serão colocadas em termos de adequação, não de possibilidades definitivas. Sempre que afirmarmos que um recurso não é o mais adequado à expressão de uma determinada idéia, não estaremos afirmando que ele não possa ser utilizado. Estaremos apenas dizendo que em uma determinada condição sua aplicação pode ser mais difícil ou mais restrita. Por exemplo, de modo geral o uso excessivo da cor como recurso expressivo pode causar problemas, no sentido de exceder a capacidade do usuário em memorizar a correlação estabelecida entre conteúdo e recurso expressivo. Em Miller (56) encontra-se uma referência a este limite como sendo de sete mais ou menos dois, além do qual o usuário pode não reter o

significado associado. Esta é uma orientação genérica. Situações de uso criativas, quando coerentes com as diretrizes gerais de orientação, podem permitir a superação de certos limites com eficiência. A decisão final sobre a correlação entre elementos de expressão e conteúdo pertence ao projetista.
Em resumo, neste capítulo, nosso objetivo geral é: (a) identificar o conjunto de recursos expressivos, (b) buscar a semântica fundamental de cada recurso expressivo, (c) buscar um conjunto possível de combinações.

4.2. A Linguagem do Usuário

4.2.1. Sistema de Códigos do Conteúdo do Usuário

Como foi dito no tópico anterior, nas interfaces por menus a expressão do usuário é associada à expressão do sistema. O usuário não expressa comandos para o sistema, ele expressa escolhas sobre as opções oferecidas através do menu. A mensagem do usuário para o sistema é definida pela informação de escolha associada à informação do menu. Embora neste trabalho o foco seja a MD, o tratamento de interfaces por menus não pode ser desconsiderado, porque o uso exclusivo de MD é insuficiente para expressar todas as necessidades típicas de interação usuário-sistema. Nos SMG a MD coexiste com menus, constituindo um estilo de interfaces multimodal. Portanto, o sistema de códigos do conteúdo do usuário será avaliado em função de uma interface multimodal. Para um tratamento adequado da linguagem, sob o enfoque semiótico, esta definição é fundamental, uma vez que o plano do conteúdo será discretizado em elementos de conteúdo, os quais serão correlacionados aos elementos expressivos, definindo o código linguístico para interação. Nesta definição vamos distinguir o que chamaremos de plano do raciocínio do plano do conteúdo do usuário.
Inicialmente vamos esclarecer a diferença entre interfaces por menus e por MD, no que se refere ao plano do conteúdo. Para isto, vamos compará-las a interfaces por linguagens de comandos. Em interfaces por menus, o sistema apresenta ao usuário o conteúdo relativo ao seu modelo funcional. Por menus entendemos todo tipo de informação que o sistema oferece ao usuário, como: menus textuais, ícones, botões e outros [Booth 89]. Portanto, no processo interativo, o usuário apenas expressa escolhas, ativações, indicações ou similar, sobre o conteúdo apresentado pelo sistema. De forma diferente, em interfaces por linguagens de comandos e por MD o usuário expressa de fato o procedimento a ser executado. Nas primeiras, ele expressa de forma textual os comandos do sistema. Nas últimas, ele expressa de forma gestual, o comando está implícito na manipulação executada.
No sentido de esclarecer o leitor, vamos fazer um paralelo com uma situação do cotidiano. A mesma diferença encontrada entre interfaces por linguagens de comandos e interfaces por menus verifica-se entre um usuário que vai a uma mercearia e ao supermercado. No primeiro caso, o usuário expressa ao balconista os artigos que deseja. No segundo caso os artigos estão expostos, o usuário expressa apenas escolhas, através do gesto de colocar a mercadoria no carrinho. Em ambos os casos a intenção do usuário (plano de raciocínio) e o produto final (mercadoria adquirida) são os mesmos. Porém, no primeiro caso o conteúdo expresso é o produto desejado, ele coincide com o plano de raciocínio. No segundo caso o conteúdo expresso é uma escolha, ele difere do plano de raciocínio. De acordo com o exemplo, nas interfaces por menus o plano do raciocínio do usuário é distinto do plano do conteúdo expressivo. O primeiro é relativo ao modelo funcional do sistema, ele raciocina sobre as operações do sistema. O segundo é relativo às ativações sobre a interface, ele expressa manifestações possíveis sobre o menu. As interfaces por MD são semelhantes àquelas por linguagens de comandos, nelas o plano do raciocínio coincide com o plano do conteúdo.
Se o plano do conteúdo é relativo ao domínio do sistema ou se é relativo a ativações sobre a interface, isto altera totalmente a natureza do elemento discretizado. Nos dois tópicos a seguir serão caracterizados e discretizados os planos do conteúdo para interfaces por MD e por menus, respectivamente. No terceiro tópico a seguir ambos serão agregados em um único sistema de códigos do conteúdo do usuário.

4.2.1.1. O Plano do Conteúdo do Usuário - Interfaces por MD

Antes de iniciarmos vale a ressalva de que todas as operações tratadas como sendo por MD podem ser executadas por menus. Neste tópico estamos averiguando apenas as possibilidades de MD.

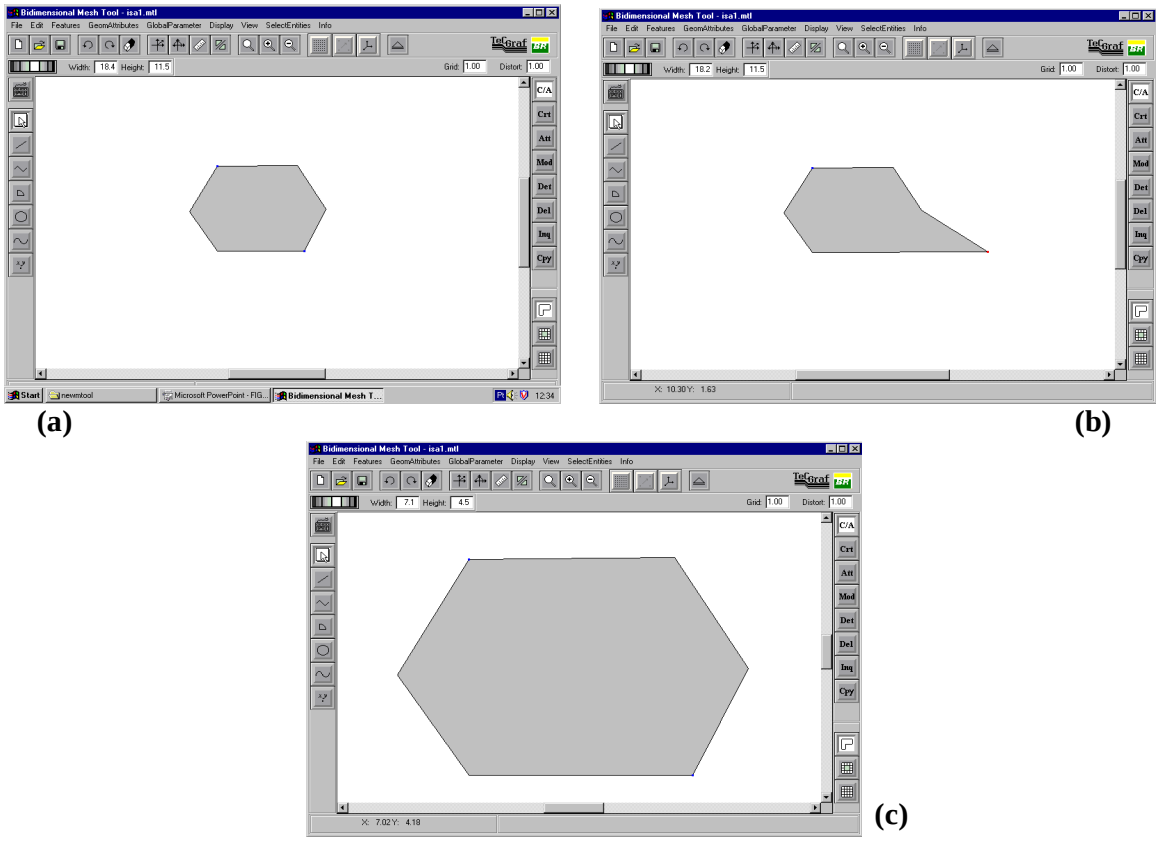


Figura 4.1. Procedimentos de manipulação do objeto e da relação usuário-objeto

Na MD constatamos dois tipos de interação: um que interfere no objeto e outro que interfere na relação do observador com o objeto. Por exemplo, quando o usuário altera a forma de um objeto, está interferindo sobre o mesmo. A Figura 4.1 (b) ilustra um procedimento de alteração de forma executado sobre o objeto da Figura 4.1 (a). A este procedimento chamamos de manipulação do objeto. Quando o usuário altera a distância de observação do objeto (zoom), está interferindo na sua relação com o mesmo. A Figura 4.1 (c) ilustra este procedimento de alteração também com relação à Figura 4.1 (a). Neste caso, embora a visão do objeto seja alterada, não há interferência sobre ele. A este procedimento chamamos de manipulação da relação usuário-objeto.

Quanto à manipulação do objeto, o usuário pode interferir sobre dois elementos do objeto gráfico: a forma e a localização no espaço. Por exemplo, quando o usuário alonga um objeto ele interfere na sua forma, quando arrasta um objeto ele interfere na sua localização no espaço. Observando a natureza do elemento manipulado em cada caso, percebemos uma ontologia que classifica a manipulação. Criar e destruir formas são operações básicas, ontogênicas, uma vez que a forma é o elemento primitivo, aquele que dá existência visual ao objeto gráfico. Nos SMG a forma é também o repositório de outros recursos gráficos, como cor e textura, que muitas vezes representam traços semânticos do objeto. As operações de deformação e movimentação estão em uma segunda categoria, de alteração de aspectos da forma. Ambas atuam sobre a forma, embora a primeira interfira na forma de fato e a segunda na sua localização no espaço. A Tabela 4.1 (a) classifica a manipulação física sobre o elemento forma.

Também através da manipulação o usuário pode interferir sobre estado ou atributo do objeto. Este é um caso particular por dois motivos. O primeiro é que a interferência não ocorre na forma física do objeto, mas na sua interpretação, trata-se de uma qualificação do objeto. O segundo motivo é que por ser uma qualificação semântica é necessária a representação dos atributos através de menus, icônico ou textual (botão). O atributo a ser associado ao objeto, por ser um elemento semântico, deve estar representado de alguma forma distinta do objeto. Caso ele esteja representado por um ícone, a MD pode ser feita através de um movimento do ícone por sobre o objeto gráfico. A sobreposição do ícone sobre o objeto dá a idéia de absorção do atributo. Uma vez que a associação tenha acorrido, ela poderá ser sinalizada, por exemplo, pela alteação de cor do objeto. A Tabela 4.1 (b) classifica a manipulação semântica sobre o elemento forma, ou seja, manipulação de qualificação do objeto. Novamente cabe aqui uma ressalva. Este tipo de operação, embora seja considerado MD, não é o tipo de MD sobre o qual está focado este trabalho. Ela está sendo considerada em função de uma definição completa do plano do conteúdo do usuário.

Quanto à manipulação da relação usuário-objeto, neste caso não há interferência sobre o mesmo, apenas na sua relação com o observador, por exemplo: alteração da distância do observador (zoom), alteração do ângulo de observação (rotação). Em ambos os casos a manipulação está interferindo na localização do observador com relação ao objeto, as diferenças são geradas pelo sentido e direção em que teoricamente o observador se

movimenta. Na rotação o observador se movimentaria em torno do objeto, no zoom ele se movimentaria perpendicularmente ao objeto. A Tabela 4.2 (a) esclarece a natureza dos elementos manipulados. Neste conjunto não se caracteriza nenhuma ontologia, todas as operações estão em uma mesma categoria.

Manipulação sobre o Objeto Operações Secundárias	**Natureza do Elemento Manipulado**
Alterar	Aspectos da forma: localização (mover) constituição (deformar)

Manipulação sobre o Objeto Operações Básicas	**Natureza do Elemento Manipulado**
Criar	Forma
Destruir	Forma

(a)

Qualificação do Objeto	**Natureza do Elemento Manipulado**
Associar	Atributo da forma

(b)

Tabela 4.1. Operações de manipulação (a) física do objeto, (b) semântica do objeto

A manipulação da relação usuário-objeto poderá ser feita através de um elemento de apoio, uma representação gráfica que represente esta relação e que possa ser manipulada pelo usuário. Por exemplo, a manipulação da distância pode ser feita através de um elemento semelhante a uma barra de rolagem, onde o usuário altera a posição de um cursor, indicando o afastamento ou aproximação do objeto em foco. Para coerência da linguagem este elemento de apoio deve permitir duas analogias: uma referente à manipulação física do objeto e a outra referente ao controle da distância. Por exemplo, o cursor na barra de rolagem é um elemento manipulável através do mouse, como se o mesmo fosse a extensão da mão do usuário. Como veremos mais à frente esta metáfora está presente em toda a MD. O caminhamento do cursor na barra de rolagem permite uma analogia com o afastamento ou aproximação do usuário em uma linha perpendicular ao objeto. O elemento de apoio gera um nível a mais na linguagem, impondo mais um nível de tradução entre a intenção do usuário e a manipulação. A questão da analogia e dos diferentes níveis de tradução linguística serão novamente tratadas na Seção 4.2.2.1 sobre a presença de metáforas na linguagem do usuário.

Manipulação da Relação Usuário-Objeto	**Natureza do Elemento Manipulado**	**Tipo de Interferência**
Rodar	Localização do observador	Ângulo de observação
Aproximar / Afastar	Localização do observador	Distância em profundidade

(a)

Manipulação da Relação Usuário-Objeto	**Natureza do Elemento Tratado**	**Tipo de Interferência**

Projetar	Visão sobre o objeto	Visualização da forma

(b)

Tabela 4.2. (a) Operações de manipulação física da relação usuáro-objeto, (b) operações com interferência na interpretação do usuário sobre o objeto

Também na manipulação da relação usuário-objeto verifica-se um segundo tipo de interferência, Tabela 4.2 (b). O usuário pode solicitar alterações na projeção do objeto. Todo objeto em 3D deve ser projetado em 2D para ser exibido no vídeo. Esta projeção pode ser feita de diferentes formas, por exemplo paralela ortogonal ou oblíqua. As diferentes perspectivas alteram não apenas a visão, mas também a compreensão do usuário sobre o objeto. Por exemplo, a projeção paralela oblíqua oferece uma visão do objeto como um todo, embora suas proporções sejam distorcidas, a projeção paralela ortogonal oferece uma visão parcial com proporções reais [Foley et alii 93]. Portanto, esta alteração não interfere fisicamente na relação usuário-objeto, ela interfere na interpretação do usuário sobre o objeto. Embora esteja sendo usado o termo interpretação, esta operação não deve se confundir com a qualificação do objeto. Ela é diferente no sentido que não interfere nas características semânticas do objeto, apenas na visão do usuário sobre o mesmo. Este tipo de atuação sobre o objeto é fruto das restrições do meio computacional (vídeos em 2D). Este tipo de operação normalmente não é executada por MD, ela deve ser ativada através de um menu, icônico ou textual, que represente as diferentes formas de projeção.

	Conteúdo Expresso pelo Usuário	**Nível de Atuação**
Manipulação Física		
Objeto	- indicação de objetos - indicação de locais - movimento para definição forma - movimento para alteração posição - movimento para alteração forma - alongamento/encurtamento	
Relação usuário-objeto	- indicação de locais - movimento para alteração da localização do usuário	sobre elemento de apoio
Manipulação Relativa à Interpretação		
Objeto	- movimento para associação de atributos	sobre menu icônico
Relação usuário-objeto		sobre menu

Tabela 4.3. Conteúdo expressivo em interfaces por MD, especificado por categorias de operações

A partir das Tabelas 4.1 e 4.2 foi feita uma reclassificação e síntese dos dados, que está apresentada na Tabela 4.3. Esta Tabela apresenta o conteúdo expresso pelo usuário nos diferentes tipos de manipulações apontados. Por exemplo, na criação de forma o usuário expressa indicação de localização da forma e a definição da forma; na destruição de forma o usuário expressa indicação de objetos; na movimentação ele expressa indicação e carregamento da forma para nova posição; e na alteração de forma ele expressa movimento de partes do objeto. Observar que na alteração de forma uma primitiva pode ter parte de si movimentada no sentido de alterar sua forma. Por exemplo, uma das pontas de um segmento de reta pode ser movimentada no sentido de alongar ou encurtar o segmento.

Na Tabela 4.3 os dados foram reclassificados em manipulação física e semântica, do objeto e da relação usuário-objeto. Na manipulação física da relação usuário-objeto o usuário interage com um elemento de apoio, que permite a manipulação simbólica da posição do usuário. Nos exemplos apurados verificamos que através dos elementos de apoio o usuário expressa indicação e movimento. Na interferência sobre a interpretação sobre o objeto o usuário interage através de menus icônicos. Neste caso, o usuário expressa indicação e movimento (ícone para sobre o objeto).

Em suma, observamos que na MD o conteúdo expressivo do usuário resume-se a indicações e movimentações.

4.2.1.2. O Plano do Conteúdo do Usuário - Interfaces por Menus

No plano de conteúdo do usuário foram constatados dois níveis de conteúdo expressivo, um de referência direta aos itens do menu, como a indicação, e outro de referência indireta a procedimentos do sistema através dos itens do menu, como a ativação. A indicação é um apontamento simples, que reflete uma escolha do usuário sobre um elemento da interface, por exemplo, a escolha de um objeto na área de canvas, a escolha de uma cor em uma palete, a escolha de um atributo em um menu. A indicação simples é utilizada para dados do sistema, como: objetos, locais e atributos. Embora a indicação de objetos e locais possa ser feita através de MD, como foi visto no

tópico anterior, ela também pode ser feita através de menus. Por exemplo, a seleção de um objeto na área de canvas pode ser feita por menus, se for oferecido um menu relacionando os objetos visualmente representados.
A ativação é um apontamento indireto através do qual o usuário aciona um procedimento do sistema. A ativação está em um nível de abstração diferente da indicação simples, uma vez que o apontamento de um elemento do menu não é o objetivo fim, ele é um meio para ativação de um processo representado por este elemento. A Tabela 4.4 esclarece o conteúdo expresso pelo usuário em cada uma das categorias. Obviamente, se o usuário pode ativar um procedimento também pode interrompê-lo. A interrupção geralmente é associada a um sinal direto do mouse (botão da direita) ou a uma tecla de função (escape).
Em resumo, observamos que nas interfaces por menus o conteúdo expressivo do usuário compõe-se de indicações, ativações e interrupções.

	Conteúdo Expresso pelo Usuário
Referência direta	- Indicação de dados
Referência indireta	- Ativação de Processos - Interrupção de Processos

Tabela 4.4. Conteúdo expressivo em interfaces por menus, especificado por categorias de operações

4.2.1.3. Definição do Sistema de Códigos do Conteúdo do Usuário

Conforme foi exposto nos dois tópicos anteriores, a partir do contínuo do conteúdo do usuário foram discretizados os seguintes tipos: indicação, movimentação, ativação e interrupção, apresentados no Resumo 4.1. No sistema de códigos do conteúdo do usuário não foram identificadas regras de estruturação, portanto o sistema de códigos coincide com os tipos discretos.

Tipos Discretos de Conteúdo do Usuário
- Indicar
- Movimentar
- Ativar
- Interromper

Regras de Estruturação
- (nenhuma regra)

Resumo 4.1. Sistema de códigos do conteúdo do usuário

4.2.2. Sistema de Códigos da Expressão do Usuário

Os tipos discretos da expressão do usuário foram segmentados a partir do contínuo de possibilidades físicas do dispositivo mouse, conforme Resumo 4.2. No sistema de códigos expressivos do usuário foi detectada a ocorrência de apenas uma regra, que associa dois sinais em sequência, gerando o tipo expressivo clique duplo. Embora a regra não expresse, existe um fator tempo distinguindo o clique duplo de dois cliques consecutivos. O clique duplo é caracterizado por um delta de tempo bastante pequeno entre os dois sinais. No Resumo 4.2 a regra foi codificada na forma de regra gramatical.

Tipos Discretos da Expressão do Usuário
- Clique simples (CS)
- Clique botão da direita
- Pressão-e-arrasto

Regra de Estruturação
- Clique duplo = CS CS

Resumo 4.2. Sistema de códigos expressivos do usuário

4.2.2.1. A Metáfora na Linguagem do Usuário

Como pode ser constatado no Resumo 4.2, o sistema de códigos expressivos do usuário é bastante simples, sua capacidade expressiva depende em grande parte da utilização de metáforas no ambiente de interação. Na MD os processos analógicos são utilizados em larga escala, de tal forma que a análise do sistema de códigos expressivos do usuário não pode ser feita desconsiderando a presença das metáforas.

A principal metáfora implícita na MD é a utilização do mouse como uma extensão da mão do usuário. Quando o usuário aponta um objeto na tela, posicionando o cursor através do mouse, a analogia é de que ele estaria apontando com o próprio dedo [Carrol et alii 88]. Por exemplo, se o usuário deseja desenhar um segmento de reta na tela, e para isto ele clica dois pontos correspondentes aos pontos inicial e final do segmento, a analogia é de que ele está apontando os pontos com o próprio dedo. Se ele pressiona-e-arrasta o mouse posicionado sobre um objeto na tela, a analogia é de que ele está pegando o objeto, carregando-o para outra posição e largando.

De modo geral, a utilização de metáforas permite que uma quantidade menor de informação precisa estar explícita [Stefik 95]. A metáfora de utilização do mouse como uma extensão da mão tem uma presença bastante forte na comunicação usuário-sistema, no sentido de que o usuário tende a lançar mão dela sempre que o contexto permite. Quando o usuário captura a metáfora utilizada, de forma consciente ou intuitiva ele transporta sua experiência em ambiente normal para o ambiente computacional, executando a manipulação naturalmente, como se o mouse fosse de fato uma extensão de sua mão.

Dois pontos dificultam a aplicação plena da metáfora de extensão da mão na linguagem do usuário. O primeiro refere-se à natureza dos contínuos expressivos do usuário e do sistema. Foi dito que o contínuo expressivo de cada agente é dependente de dispositivo, o mouse para o usuário e o vídeo para o sistema. Esta diferença de dispositivos de comunicação cria algumas dificuldades na implementação da metáfora de extensão da mão, além de acomodar a presença de outra metáfora - a metáfora da visão. O segundo ponto refere-se às limitações impostas pelas metáforas à expressão do usuário. Dentro do escopo de uma única metáfora os tipos expressivos são insuficientes, criando a necessidade de tipos complementares, e causando rupturas. A implementação parcial de metáforas gera irregularidades no código linguístico, aumentando a carga cognitiva imposta ao usuário. A seguir veremos em detalhes cada um destes pontos, analisando a atuação das metáforas, seus limites e seus pontos de ruptura.

4.2.2.2. A Coexistência de Duas Metáforas no Plano da Expressão do Usuário

Via de regra, o ser humano lança mão de seus cinco sentidos para interagir com o ambiente em que vive. Na interação entre pessoas são utilizados maciçamente os sentidos da audição e visão, acrescidos do recurso da fala. Quando seres humanos se comunicam, eles compartilham um mesmo plano da expressão. Expressando-se através da fala, cada agente da comunicação tem um retorno auditivo, tanto da sua própria fala, quanto da resposta do outro, conforme ilustrado na Figura 4.2 (a). Neste diálogo, o sentido da audição é predominante e o sentido da visão é complementar. Por exemplo, uma pessoa pode dialogar com um parceiro que não está vendo, como ocorre ao telefone. Fala e audição referem-se a dispositivos emissores e receptores, que operam através de um recurso expressivo de mesma natureza, no caso a linguagem verbal. Ambos os agentes da comunicação dispõem dos mesmos dispositivos. Portanto existe uma homogeneidade do recurso expressivo.

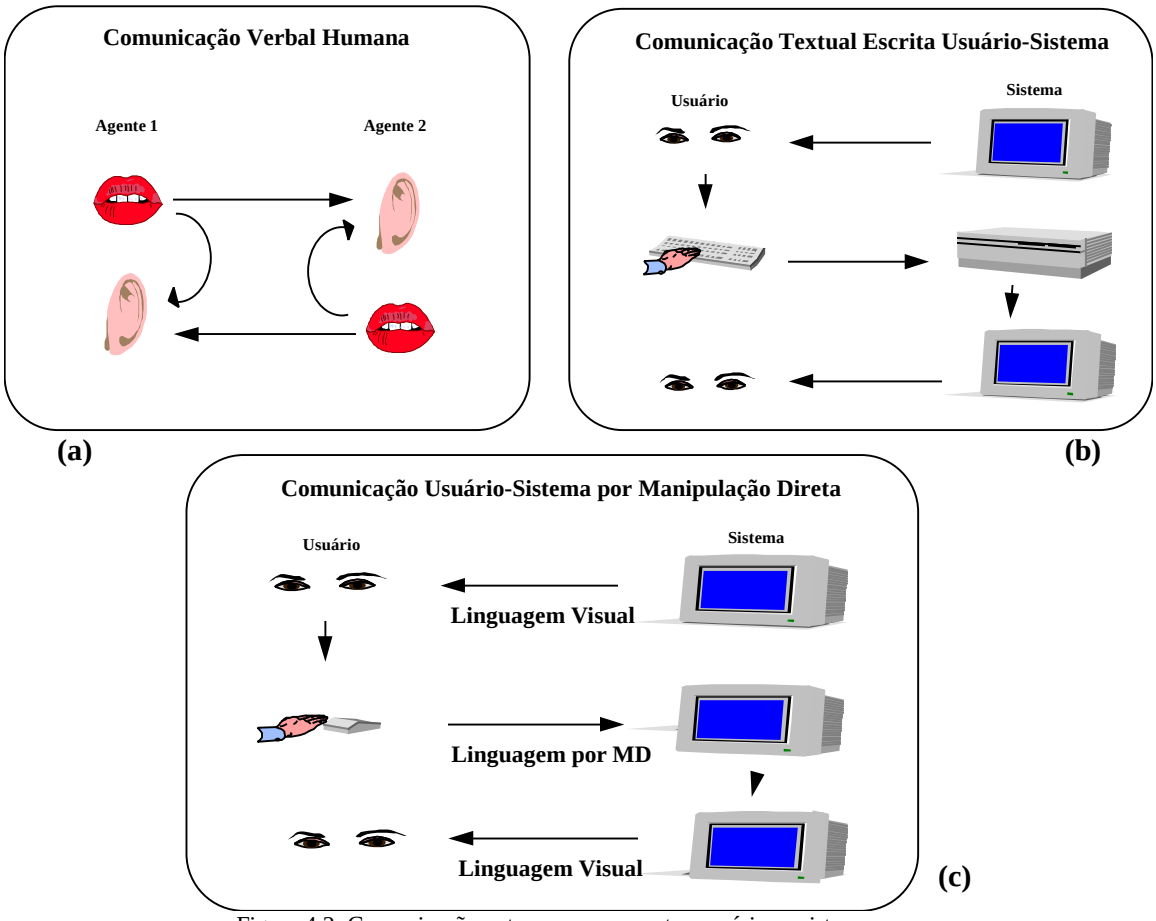
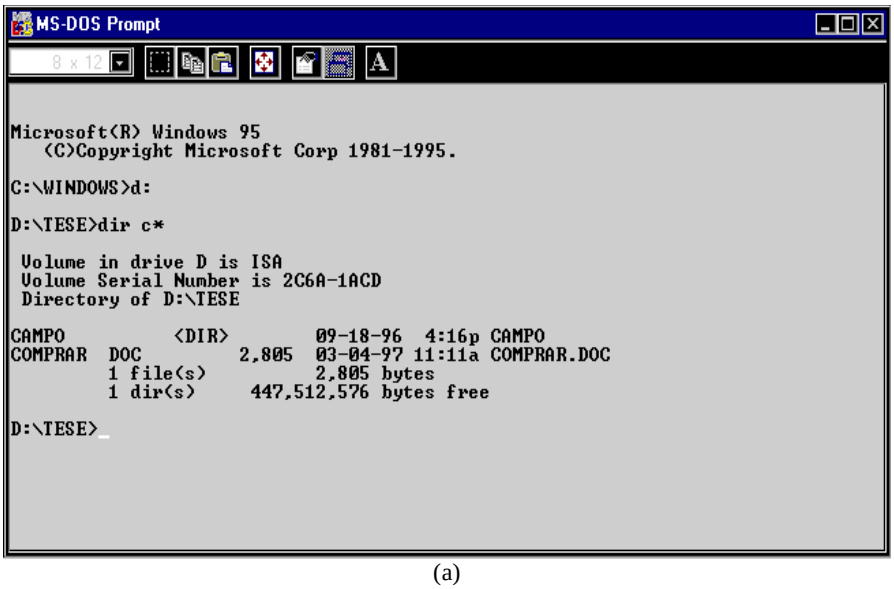
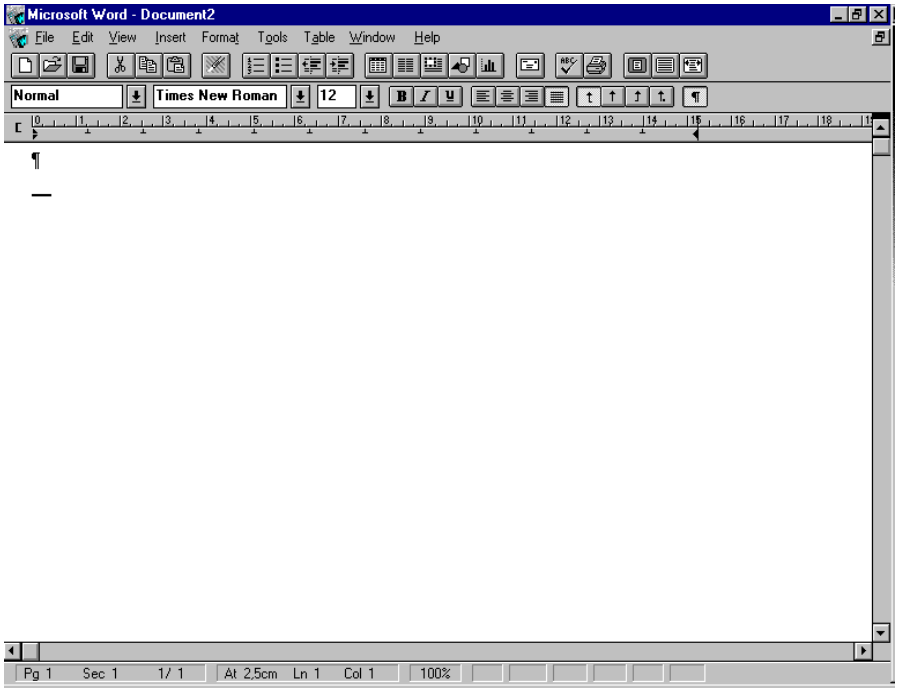


Figura 4.2. Comunicação entre pessoas, e entre usuários e sistemas

Na comunicação com o computador esta relação se altera. No caso do usuário, o recurso da fala é substituído pela manipulação do mouse ou teclado, e o retorno não é auditivo, é visual. Permanecem em uso os sentidos da visão e da audição, porém com uma diferença, a visão passa a ser predominante enquanto a audição passa a ser complementar. Ou seja, a interação com o computador pode dispor da comunicação auditiva mas não pode dispor da comunicação visual, traçando-se aí um paralelo com a comunicação escrita.



(a)



(b)

Figura 4.3. Exemplos de diálogo usuário-sistema através de linguagem textual escrita

Quando a comunicação usuário-sistema é feita através de uma linguagem textual (por exemplo, menu ou linguagem de comandos), o usuário recebe no vídeo um retorno relativo ao que foi digitado, na mesma linguagem. Se o usuário estiver usando uma linguagem de comandos, ele recebe o retorno do comando digitado. Se ele estiver usando uma interface por menus, em geral, recebe um retorno visual assinalando a opção selecionada, como por exemplo um fundo reverso. Mesmo neste caso o retorno textual se mantém. Nas interfaces textuais, apesar da comunicação através de dispositivos diferentes, a relação se mantém semelhante à comunicação humana, com um recurso expressivo homogêneo, conforme ilustrado na Figura 4.2 (b).

A Figura 4.3 apresenta dois exemplos de diálogos em linguagem textual, na Figura 4.3 (a) um diálogo por linguagem de comandos, na Figura 4.3 (b) um diálogo por menus. Em ambos os casos o usuário se expressa e recebe o retorno através de linguagem textual.

Na comunicação por MD o usuário (a) formula sua intenção sobre uma informação visual, (b) expressa esta intenção através do mouse, e (c) recebe no vídeo um retorno visual, correspondente à manipulação. Associada à MD está a metáfora de extensão da mão, portanto a expressão por manipulação tem uma natureza mecânica. Associada à expressão visual está a metáfora do olho, portanto a expressão visual tem uma natureza essencialmente simbólica. Ou seja, o usuário recebe um estímulo de natureza simbólica, expressa-se através de um recurso de natureza mecânica e recebe o retorno através de um recurso de natureza simbólica. A Figura 4.2 (c) ilustra esta situação. Recursos expressivos de natureza dessemelhante coexistem no plano da expressão do usuário. A Figura 4.4 apresenta um exemplo de operação por MD com retorno visual, na qual uma linha está sendo criada. A linha é inicializada por um clique na área de canvas, quando então o sistema retorna uma linha sinalizadora (rubber-band).

Em uma manipulação em ambiente normal, o ser humano tem retorno pelo tato. Por exemplo, ao tocar um objeto uma pessoa identifica sem muita dificuldade se ele é de madeira ou metal. Este retorno não existe no ambiente computacional. Na utilização do mouse tradicional o único retorno sensitivo possível é de posicionamento espacial, em função da rolagem do mouse sobre a base, embora este retorno não seja preciso em termos de valores. No processo de rolagem, quando a base acaba é possível levantar o mouse, reposicioná-lo e continuar rolando. Esta descontinuidade no plano da base não tem correspondente na tela. Para que o usuário tenha um retorno preciso de posicionamento são necessárias referências visuais na tela do vídeo.

A metáfora de extensão da mão está fortemente ligada ao formato do mouse. Por exemplo, nos laptops o dispositivo de caminhamento que mais se aproxima do mouse convencional é o painel de toque, mas os dispositivos mais utilizados são a esfera (trackball) e o pino. Nestes equipamentos existem dispositivos de pressão correspondentes aos botões do mouse. Em alguns casos os dispositivos de pressão não são conjugados ao dispositivo de caminhamento, o usuário pode trabalhar com as duas mãos ao mesmo tempo, uma executando o caminhamento e outra pressionando. Neste casos o retorno de posicionamento é menor ainda, ficando praticamente o retorno visual. A insuficiência do retorno manual-sensitivo do usuário evidencia a presença das duas metáforas, sendo necessária uma informação visual de retorno para que ele se posicione corretamente.

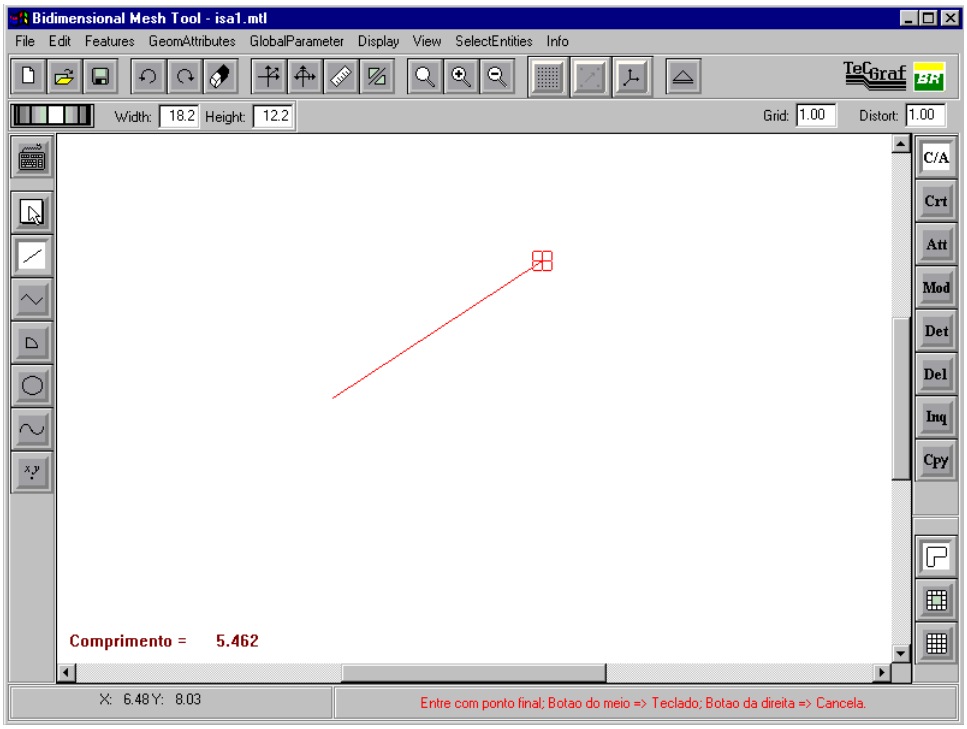


Figura 4.4. Exemplo de diálogo usuário-sistema através de manipulação direta

4.2.2.3. A Consistência da Linguagem do Usuário com Relação às Metáforas

Na análise do plano do conteúdo do usuário foram constatados dois tipos básicos de operação por MD, a manipulação do objeto e a manipulação da relação usuário-objeto. Entre elas vamos encontrar uma diferença fundamental na associação do conteúdo ao recurso expressivo. No caso de manipulação do objeto a intenção do usuário pode ser traduzida diretamente para o gesto, de acordo com a metáfora de extensão da mão. Por exemplo, se o usuário deseja deslocar um objeto gráfico, o tipo discreto movimentação pode ser diretamente associado ao tipo expressivo pressão-e-arrasto, referente ao mouse. A idéia de mover um objeto é coerente com a metáfora, onde a mão pode pegar ou pressionar um objeto, deslocando-o de lugar. Portanto, a tradução é imediata.

No caso de manipulação da relação usuário-objeto, um elemento gráfico de apoio será necessário. Supondo-se, por exemplo, que o usuário deseje aproximar o objeto gráfico, ele necessitará um elemento de apoio para fazer a manipulação da distância entre ele e o objeto. Sendo uma manipulação por mouse, para que seja intuitiva o elemento gráfico de apoio deve permitir duas analogias: (a) a uma manipulação compatível com a metáfora de extensão da mão e (b) uma manipulação compatível a idéia de alongamento/encurtamento de distância. A natureza do elemento gráfico de apoio deve permitir a tradução da intenção do usuário para uma manipulação de natureza mecânica, além da idéia de tratamento de distância.

Vamos exemplificar esta questão através de duas manipulação diferentes para uma operação de aproximação (zoom-in). De acordo com os processos de tradução semântica e articulatória apontados por Norman, tratados no Capítulo 2, a tradução entre a intenção do usuário e uma ativação do sistema ocorre em três etapas. Na primeira etapa o usuário formula a intenção de aproximar a imagem restringindo sua visão a uma determinada área, na segunda etapa esta intenção é traduzida em uma operação conceitual do sistema, no caso uma operação de aproximação, e, na terceira etapa, a operação conceitual do sistema é traduzida em uma ativação. Por exemplo, uma barra de rolagem na qual o usuário possa correr um cursor dando a idéia de aproximação ou afastamento do objeto. Este elemento gráfico visual permite que o usuário expresse uma manipulação mecânica, ele faz com que a manipulação da relação usuário-objeto assemelhe-se à própria manipulação do objeto. A associação da expressão do usuário a um recurso expressivo de pressão-e-arrasto, estaria de acordo com a metáfora de extensão da mão. A relação entre o cursor e a barra de rolagem permitiria a analogia com a distância entre o usuário e o objeto. Portanto, este elemento visual favoreceria uma tradução intuitiva.

Observe-se que a proposta da barra de rolagem permite aproximar/afastar a imagem sem controle de foco. Supondo-se que a aproximação fosse feita com relaçao ao centro da tela, o controle de foco poderia ser feito através de rolagem do conteúdo da tela, trazendo para o centro a área de interesse.

Outro recurso bastante utilizado nas operações de aproximação é a definição de uma janela sobre a representação gráfica para demarcar a área de visualização. Neste caso não existe uma aproximação gradual, a aproximação ocorre aos saltos, conforme as áreas de visualização demarcadas. Este tipo de recurso, embora muito comum é menos favorável a uma tradução intuitiva do que o exemplo anterior. Embora o traçado da janela faça parte da cultura do usuário, não é uma tradução mecânica como a anterior. Outro motivo pelo qual o segundo exemplo não

favorece a tradução intuitiva é que, sobre a área de canvas, o recurso da janela só pode ser utilizado para a manipulação de aproximação, quando o usuário delimita uma área de visualização menor do que a área corrente. A analogia só é válida para a aproximação, mas é quebrada no afastamento. Para a operação de afastamento, que é similar, a linguagem teria que lançar mão de um recurso textual, através de menu, ou de um recurso gráfico adicional, por exemplo uma janela menor com uma visão completa da representação gráfica, sobre a qual o usuário pudesse delimitada uma nova área de visualização, maior do que a área corrente. A segunda solução obviamente implica em uma área da tela reservada para esta visão geral do objeto. Com relação à proposta da barra de rolagem, esta proposta tem a vantagem de permitir uma aproximação focada.
Em ambas as propostas de manipulação da relação usuário-objeto foram necessários dois passos na tradução semântica do usuário. Ele traduz sua intenção em uma operação do sistema (zoom-in), traduz a operação do sistema na manipulação de um elemento que produziria o efeito desejado, em seguida ele executa a ativação. No caso da janela a ativação corresponde ao traçado da janela, no caso da barra de rolagem corresponde à manipulação do cursor.

Pontos de Ruptura da Metáfora da Mão

A geração do código linguístico de acordo com a metáfora da mão sofre algumas limitações impostas pela própria metáfora. Dizer que a codificação vai além do escopo da metáfora, significa dizer que as associações feitas entre expressão e conteúdo, em ambiente computacional, não têm correspondentes entre aquelas associações que seriam feitas pelo usuário em ambiente análogo. Por exemplo, a utilização de uma pressão-e-arrasto para alongar um segmento de reta, é diferente da utilização do mesmo recurso para movimentá-lo. No primeiro caso o usuário necessitaria de duas mãos, uma para segurar o objeto e outra para puxar, alongando-o. No entanto, em geral, tanto as operações de movimento quanto as de alongamento são associadas ao mesmo sinal do mouse. Elas se distinguem pelo ponto de posicionamento do cursor. No segundo caso o projetista oferece ao usuário uma referência visual dos pontos de alongamento, e a tradução da intenção do usuário em uma manipulação viabiliza-se através deste referencial visual simbólico.
Outros exemplos de ruptura da metáfora são o clique duplo, usado para ativação de processo, e o clique do botão da direita, usado para interrupção de processo. Nestes casos a associação entre conteúdo e recurso expressivo é totalmente simbólica, fugindo ao escopo da metáfora.

4.2.3. A Codificação das Correlações entre Conteúdo e Expressão do Usuário

A linguagem do usuário será definida pelo código entre conteúdo e expressão. O código cria os tipos gerais da linguagem. De acordo com o que foi visto até agora, no sentido de uma linguagem consistente, seria desejável que a codificação obedecesse as metáforas utilizadas, de forma que a estrutura conceitual envolvida na metáfora fosse transportada plenamente para o ambiente computacional. Por outro lado vimos também que a metáfora é restrita, ela não pemite esta amplitude de utilização. Portanto, ao analisarmos um código já estabelecido em ambiente comercial, estaremos comparando codificações reais contra codificações desejáveis.
Na Tabela 4.5 é apresentado um código bastante comum, encontrado nos aplicativos gráficos. Parte dos sinais do mouse é utilizada de acordo com a metáfora de extensão da mão (dois primeiros casos), outra parte obedece a uma codificação simbólica. Quando se esgotam as associações no escopo da metáfora, os tipos expressivos passam a ser associados arbitrariamente. Quanto mais simbólica e arbitrária a codificação maior a carga cognitiva sobre o usuário.
Pelos exemplos anteriores concluímos que nas operações de apontamento e movimento a metáfora de extensão da mão é respeitada. Na operação de alongamento já não existe uma metáfora propriamente dita, mas uma alusão a ela. Neste caso, o código é um índice no sentido adotado por Peirce [Peirce 31], onde a operação manual executada é um indicativo da metáfora de manipulação pela mão do usuário. Nos casos do clique duplo e do botão à direita verifica-se uma associação totalmente arbitrária. Muitos códigos arbitrários, semelhantes a estes últimos, podem ser criados, por exemplo um clique duplo no botão da direita, com o ônus de aumentar cada vez mais a carga cognitiva imposta ao usuário. Concluímos que os sinais do mouse codificados de acordo com a metáfora são apenas uma parcela do conjunto total, e que o mesmo sinal (pressão-e-arrasto) é reutilizado em codificações diferentes.
Como foi dito, a natureza do dispositivo de entrada afeta a metáfora. Por exemplo, nos laptops, onde são utilizados o pino e a esfera, a ruptura da metáfora e as associações arbitrárias ficam mais evidentes. Nestes equipamentos, em função da independência dos dispositivos de pressão, o usuário pode trabalhar com as duas mãos ao mesmo tempo, uma executando o caminhamento e outra pressionando. Nesta situação, mesmo a operação de apontamento, se não rompe a metáfora de extensão da mão, pelo menos a deixa bastante enfraquecida.

Tipo de Conteúdo	Tipo Expressivo
Indicar	Clique
Movimentar	Pressão-e-arrasto
Ativar	Clique duplo

Interromper	Clique botão direita

Tabela 4.5. Um exemplo de codificação sobre os sinais do mouse

O exemplo da Tabela 4.5 é um caso particular de codificação, não é o único encontrado. Por exemplo, nem sempre a interrupção é associada ao botão da direita. Em algumas linguagens vamos encontrar conteúdos semelhantes associados a sinais diferentes. Por exemplo, vamos encontrar operações de criação semelhantes, como criação de linha e polilinha, uma delas implementada com clique simples e outra com pressão-e-arrasto (como se fosse um alongamento). A ausência de uma linha clara de lexicalização propicia este tipo de solução ocasional.

Em resumo, os seguintes pontos se destacam na codificação da linguagem de expressão do usuário: (a) no que se refere à expressão do usuário, na LV, não é possível tratar a linguagem desconsiderando a utilização de metáforas; (b) existe um hiato no plano da expressão do usuário causado pela coexistência de meios expressivos de natureza diferente, o usuário expresssa-se através de uma metáfora mecânica (física) e recebe o retorno através de uma metáfora visual (simbólica). Este hiato representa um aumento na carga cognitiva imposta ao usuário que, ao se comunicar com o sistema, deve fazer traduções de uma metáfora para outra. Esta carga cognitiva pode ser minimizada através de uma codificação sistemática da linguagem em torno da metáfora; (c) existe um hiato no plano da expressão do usuário causado pelas manipulações que exigem uma intermediação, como a manipulação da relação usuário-sistema e a qualificação de objetos (manipulação de elementos semânticos); e (d) as metáforas devem ser respeitadas como eixos de lexicalização para a produção de uma codificação sistemática.

4.3. A Linguagem do Sistema

Semelhante ao que foi feito para a linguagem de expressão do usuário, vamos segmentar os planos do conteúdo e da expressão do sistema, discretizando tipos e buscando as associações mais adequadas entre os tipos do conteúdo e da expressão. No caso da linguagem do usuário, os recursos expressivos restringem-se aos sinais do mouse, e boa parte de sua utilização já está culturalmente estabelecida. Embora em algumas situações possamos constatar que a utilização feita não é a mais adequada, geralmente encontra-se muita resistência em alterar alguma coisa já estabelecida no plano comercial. No caso da linguagem do sistema é diferente; os recursos expressivos são mais variados e sua utilização mais flexível. Por exemplo, a informação de que uma primitiva está selecionada pode ser traduzida através de forma, cor ou textura. Esta flexibilidade pede um tratamento mais detalhado no sentido de definir que tipo de recurso pode ser mais adequado em cada caso, considerando a eficácia do recurso expressivo no ambiente computacional.

4.3.1. Sistema de Códigos do Conteúdo do Sistema

Conforme foi apresentado no Capítulo 2, através da interface o projetista deve transmitir ao usuário o modelo funcional (o que o sistema faz) e o modelo de interação do sistema (como ativá-lo). O modelo de interação abrange toda a interação do usuário com o sistema, parte da qual não será tratada neste trabalho, uma vez que estamos nos restringindo à MD. A segmentação do contínuo expressivo do sistema será tratada a partir desta distinção entre modelos funcional e de interação, e cada um dos modelos será subdividido em outros componentes, conforme demonstrado na Figura 4.5.

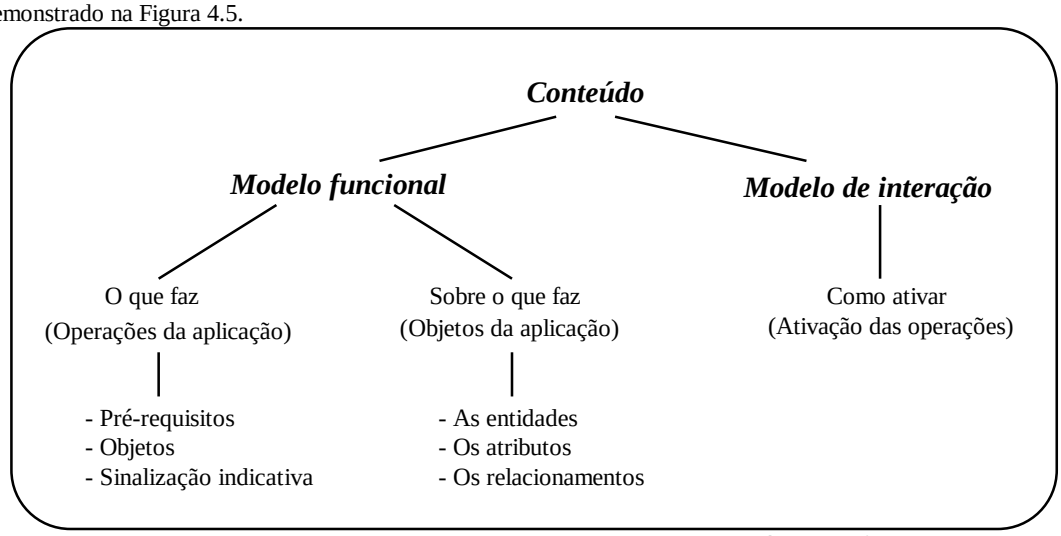


Figura 4.5. Modelo conceitual do sistema a ser expresso através da interface

4.3.1.1. Segmentação Referente ao Modelo Funcional

O contínuo expressivo relativo ao modelo funcional do sistema será segmentado a partir das operações (comportamento do sistema) e dos objetos (natureza dos dados) envolvidos em uma dada aplicação. Como foi dito

na Seção 4.1, esta segmentação não é absoluta, conforme o caso ela pode exigir algum tipo de adaptação, no sentido de considerar outros componentes.

Segmentação Referente às Operações

Como já foi dito, as operações por MD não são explicitadas na interface. Portanto, no que se refere ao comportamento do sistema a linguagem de interação pode expressar os seguintes componentes, sem serem obrigatórios: (a) estados do sistema, (b) pré-requisitos, (c) especificação do objeto da operação.

(a) Estados do Sistema
Para efeito de tratamento podemos dividir os estados do sistema em duas categorias: os estados que interferem no resultado da operação (estados correntes) e os que apenas caracterizam operações em progresso (estados de execução). Como exemplo da primeira categoria temos os atributos de primitivas no desenhador MsDraw [Word 95]. Neste aplicativo os atributos são implementados como estados correntes do sistema. Quando uma primitiva é traçada, ela recebe os valores correntes para atributos como cor, espessura e estilo de linha. A alteração do valor corrente destas variáveis é uma ação que deve ser ativada pelo usuário. A variável de estado corrente interfere no resultado da operação no sentido que altera as características da primitiva gerada.
A grande maioria dos sistemas comerciais não expressa as variáveis de estado do sistema. No corpo de aplicativos selecionado para o Estudo de Caso, apenas o aplicativo MGE [MGE 96], no caso das operações sujeitas a variáveis de estado do sistema, abre uma pequena janela mostrando o valor corrente dessas variáveis. Obviamente, se o projetista desejar mostrar ao usuário as variáveis de estado, ele deverá usar a unidade tópica de discurso como referência, sinalizando apenas as variáveis envolvidas. Dentro da unidade tópica, o conteúdo expresso deve estar de acordo não apenas com a operação em execução, mas com o foco de atenção do usuário. Mesmo para os sistemas que não expressam as variáveis de estado, elas devem ser declaradas como tipos e instâncias do conteúdo de sistema. É uma forma de orientar o projetista a considerá-las na especificação da gramática.
Os estados de execução, que caracterizam operações em progresso, têm em geral duas instâncias: ativo e inativo. Uma instância dos estados de execução é, por exemplo, o estado de criação. Este estado é ativado quando o usuário solicita a criação de uma primitiva, e desativado quando o traçado é encerrado. Por exemplo, quando o usuário solicita o traçado de uma linha o sistema entra em estado de criação, aguardando a entrada de dados. O estado de criação ativo é geralmente expresso por algum tipo de cursor diferenciado do cursor corrente. Se o cursor corrente é uma seta, o cursor para estado de criação ativo pode ser, por exemplo, um  . Após a especificação do ponto inicial da linha, o sistema entra em novo estado de espera (o anterior ainda não foi encerrado), que poderemos chamar de estado de finalização, uma vez que ele está aguardando a especificação do ponto final. Este novo estado é geralmente sinalizado por uma linha de traçado (rubber-band), que se movimenta em função do movimento do mouse, indicando a forma corrente do segmento de reta.
Um outro exemplo pode ser verificado no aplicativo MTool (97). Quando o usuário traça uma curva de Bezier, o sistema oferece a curva traçada com 4 pontos de manipulação para que o usuário dê a ela a forma desejada. Neste caso, o estado de criação daquela primitiva específica é sinalizado por pontos de manipulação. Uma vez que o usuário tenha encerrado o traçado, os pontos de manipulação desaparecem.

(b) Pré-requisitos
Os pré-requisitos são condições que viabilizam a execução de uma operação. Um exemplo de pré-requisito é a seleção de objetos. Na maioria dos aplicativos um objeto deve estar selecionado para que possa ser manipulado. Em geral o pré-requisito está associado a uma variável de estado do objeto. Esta variável atua como um protocolo de abertura e encerramento de uma unidade de discurso. No exemplo anterior, uma vez que o objeto seja selecionado, ele pode passar por uma série de operações consecutivas até que a seleção seja desfeita.

(c) Especificação do Objeto da Operação
Neste ponto da análise de discretização não estamos nos referindo à expressão das características do objeto, esta questão será tratada no tópico sobre segmentação do conteúdo expressivo no que se refere aos objetos. Aqui, estamos tratando da especificação do objeto da operação.
Um objeto pode ser prévia ou posteriormente especificado com relação à operação. Um exemplo de especificação prévia é a já citada seleção de objetos. A seleção é uma especificação prévia de objeto, para várias operações que podem ser executadas em sequência. Um exemplo de especificação posterior é a criação de primitivas por MD. Neste caso, a operação deve ser especificada antes do objeto, ou seja, antes da entrada de dados, para que o sistema saiba interpretar corretamente os dados de entrada. Por exemplo, dois pontos assinalados na tela podem delimitar um segmento de reta ou o raio de uma circunferência.

Segmentação Referente aos Objetos da Aplicação

Para segmentação do plano do conteúdo do sistema no que se refere aos objetos, foi adotado o modelo ERA - Entidades, Relacionamentos e Atributos [Chen 76, Batini et alii 92]. A escolha deve-se ao fato de ser um modelo muito difundido, no sentido de especificar o domínio do sistema, e adequado aos sistemas com os quais estamos

trabalhando. Não é a única opção possível, outro modelo poderá ser adotado como referência na segmentação do contínuo expressivo do sistema, desde que dê conta de todos os elementos de conteúdo do domínio do sistema.
Na utilização do modelo ERA pode ocorrer uma dubiedade com a palavra atributo. Para os SMG, atributos são elementos específicos que podem ser associados às entidades. Por exemplo, a especificação de um tipo de material, como concreto ou ferro, para um objeto é um atributo. Para o modelo ERA atributo é tudo que pode ser associado a uma determinada entidade, sem a especificidade dos SMG. Apesar do uso da palavra ser mais restritivo nos SMG, os atributos do SMG também são atributos no modelo ERA. Portanto, neste trabalho, as diferenças podem ser ignoradas. Com base no modelo ERA, o sistema pode expressar: entidades, relacionamentos das entidades, atributos sobre as entidades, atributos sobre os relacionamentos.
(a) As Entidades
As entidades são as primitivas do sistema em questão. Além das primitivas referentes ao domínio do sistema, vamos encontramos as primitivas de interação, utilizadas para auxiliar o usuário no traçado da representação gráfica. Nem sempre as primitivas do sistema são equivalentes às primitivas de interação. A definição destas atende a critérios específicos. Esta questão está sendo tratada em detalhes na Seção 5.2.3 sobre a lexicalização das primitivas de interação.
(b) Os Atributos
Nos SMG os atributos que atuam sobre as entidades são: os atributos (grupo de qualificadores chamados de atributos nos SMG), os estados e as propriedades. Os nomes destes qualificadores de objetos podem variar de um sistema para outro. Atributos podem ser, por exemplo: material, força, peso. Propriedades podem ser, por exemplo: é uma face, passa calor, é isolante. Estados podem ser, por exemplo: selecionado/não selecionado, ativo/desligado.
Porque os atributos se aplicam aos objetos e porque serão representados sobre a forma dos objetos, é necessário buscar uma classificação a partir da qual se possa buscar uma forma de expressão mais adequada. Por exemplo, uma classificação possível seria entre: atributos dinâmicos ou estáticos, contínuos ou discretos. Por exemplo, um atributo contínuo seria uma propriedade, como ser isolante, um atributo discreto seria uma força associada a um elemento de uma estrutura.
(c) Os Relacionamentos
Nos SMG encontramos relacionamentos como, por exemplos: hierarquia e agrupamento.
Em resumo, os tipos discretizados no que se refere aos objetos das operações são: entidades, atributos e relacionamentos.

4.3.1.2. Segmentação Referente ao Modelo de Interação

Expressar o modelo de interação significa expressar informações sobre como ativar o sistema. O modelo de interação referente aos menus (textuais, icônico, de botões e outros) não será tratado neste trabalho. Trabalharemos apenas as questões de interação referentes à MD.
Na interação por MD encontramos basicamente os sinalizadores de apoio à manipulação, eles auxiliam o usuário durante a manipulação. Por exemplo, no aplicativo MTool (97) a curva de Bezier é inicialmente traçada com a aparência de um segmento de reta, com quatro pontos sinalizados para manipulação, para que o usuário dê a ela a forma desejada. Por exemplo, os ■ sinalizando pontos de manipulação são símbolos convencionalizados, existindo uma cultura estabelecida sobre esta convenção.
Os sinalizadores de apoio formam uma categoria especial que será bastante utilizada nas operações de manipulação da relação usuário-objeto. Como foi visto na Seção 4.2.2.2, o aspecto importante no que se refere a estes sinalizadores de apoio é que o usuário, ao utilizá-los, está fazendo a tradução da metáfora visual para a metáfora de manipulação. Portanto, o sinalizador deve sugerir algo manipulável.

4.3.1.3. Definição do Sistema de Códigos do Conteúdo do Sistema

Os tipos discretos de conteúdo do sistema, referentes aos modelos funcional e de interação, encontram-se relacionados no Resumo 4.3. A estruturação do sistema de códigos do conteúdo do sistema será feita sobre os tipos discretizados, de acordo com o domínio do sistema.

Tipos do Conteúdo do Sistema
- Variáveis de estados correntes do sistema
- Variáveis de estados de execução do sistema
- Pré-requisitos
- Sinalizadores de apoio
- Entidades
- Variáveis de estado do objeto
- Atributos
- Relacionamentos

Resumo 4.3. Sistema de códigos do conteúdo do sistema

4.3.2. Sistema de Códigos para Expressão do Sistema

A seguir faremos uma análise dos recursos visuais elementares em ambiente computacional. A análise será feita a partir de uma segmentação do contínuo expressivo visual encontrado em Dondis [Dondis 73], para uso na comunicação humana. Faremos um estudo sobre os elementos visuais por ela apresentados, no sentido de verificar seu comportamento em ambiente computacional, elegendo os mais eficazes para a comunicação neste ambiente.

Sobre a Forma

As formas mais primitivas são o ponto e a linha. A forma é a representação básica, ela concretiza a existência do objeto gráfico, e por isto é suporte para outros recursos visuais. A forma pode ser um recurso neutro ou um recurso de distinção. O recurso é dito neutro quando ele próprio não tem significado, servindo apenas de suporte para outro recurso expressivo. Por exemplo, temos o caso já citado onde diferentes substâncias químicas são visualmente representadas através de círculos e a distinção de substância é feita pela cor dos círculos. Neste caso a forma é um recurso neutro e a cor é um recurso de distinção. Ou ainda, um organograma onde cada caixa não tem significado por si, mas pela posição que ocupa. Nos SMG, porque a entrada de dados é feita através da representação visual, as formas utilizadas no objeto modelado sempre são elementos de distinção.
Nos SMG, uma vez que a forma tenha sido utilizada como elemento de distinção, ela não pode veicular nenhum outro tipo de significado, como por exemplo, uma mudança de estado dinamicamente representada através da forma. A expressão desta idéia exigiria alterações na forma, correspondentes às alterações de estado. Nos SMG, pelo fato de a captura de dados ser feita através da representação visual, uma alteração de forma como esta implicaria em uma ruptura semântica. Nestes sistemas as primitivas de forma são pré-definidas, não podendo sofrer alterações que as descaracterizem.

(a) O Ponto

O ponto é a unidade mínima de representação visual. Quando representado em um dispositivo de saída, vídeo ou impressora, torna-se pouco perceptível aos olhos do usuário. Por isto, pontos que são objetos do sistema e que podem ser individualmente manipulados pelo usuário são representados por uma pequena forma, como por exemplo um asterisco. Neste caso a forma não corresponde exatamente ao conteúdo, ela é um índice [Peirce 31] da forma original, o ponto.

(b) A linha

Nos SMG a linha é o traço básico de representação tanto das formas esquematizadas, como das formas mais realistas. Nestes sistemas ela atende a características técnicas, como por exemplo: escala, espessura e outros.

(c) As Formas Básicas

A partir de formas básicas como o quadrado, o círculo e o triângulo, compõem-se a maioria das formas encontradas na natureza. Nos SMG, estas formas básicas, quando ocorrem, são associadas a um modelo semântico, ou seja, representam ou compõem a representação de alguma entidade do modelo semântico. Por exemplo, nos SMG com base em modelos construtivistas [Foley et alii 93] as primitivas simples são combinadas através de operadores de conjuntos Booleanos, para compor modelos complexos.

Sobre a Cor

Em ambiente computacional a cor é o elemento menos específico, ou seja, é aquele que mais se presta à expressão de diferentes tipos de informação. Porque a cor é necessariamente um atributo sobre uma forma, ela caracteriza facilmente atributos do objeto semântico, onde mudanças de cor sinalizam mudanças de atributo. Ela é um elemento de discretização, podendo ser usada para caracterizar praticamente tudo: distinção de entidades, estados, propriedades, relações (por exemplo agrupamento). A cor permite expressar conteúdos estáticos e dinâmicos com a mesma flexibilidade. Ela pode veicular significados conotativos associados à realidade, como céu-azul, mata-verde, ou significados simbólicos convencionalizados, como vermelho-perigo. Ela também pode ser usada apenas para conforto visual, como um fundo de tela em uma cor repousante para a vista, ou com finalidades estéticas, como um título colorido para criar um efeito de embelezamento.
Se por um lado a flexibilidade deste recurso permite a expressão de um grande leque de informações, por outro exige o máximo de cuidado. O uso casual da cor pode confundir o usuário, que não identificando nenhum tipo de sistematicidade no uso do recurso, pode também não identificá-lo como recurso expressivo. Este tipo de problema ocorre com qualquer recurso de expressão, mas no caso da cor o problema é potencialmente mais grave, em função da excessiva flexibilidade de utilização que ela propicia. Por exemplo, uma situação de uso casual da cor vermelha é quando o projetista a utiliza para: (a) mensagens de erro, com o objetivo de alertar o usuário; (b) áreas de títulos, por motivos estéticos e (c) assinalar estado de seleção de primitivas. Neste caso o usuário pode não reconhecer nenhuma mensagem através do uso da cor vermelha. O uso casual concomitantemente a um uso sistemático, pode anular o feito deste.
Existem algumas restrições de uso para cor, relativas ao tamanho das superfície e a cor utilizada. Por exemplo, uma linha fina em uma cor clara, contra um fundo também em cor clara pode não ser visualmente capturada. Ou, uma superfície grande com uma cor excessivamente forte pode ser visualmente cansativo ou desagradável.

Porque a cor oferece uma variedade grande de opções, alguns sistemas, como por exemplo o ambiente Windows, permitem a customização de cores do ambiente. A utilização deste recurso exige cuidados, especialmente nos ambientes em que a cor é um recurso expressivo. Nos casos onde a cor é apenas um recurso estético, uma troca de palete pode causar desconforto ou confusão visual quando, por exemplo, duas áreas cujas cores deveriam estar destacadas ficam próximas uma da outra. Nos casos onde a cor é um recurso expressivo, a troca de palete pode causar sérios problemas, especialmente se as cores utilizadas tiverem uma conotação já estabelecida, como por exemplo uma mensagem em vermelho para alertar sobre um problema.
No aplicativo MTool (92) encontramos um exemplo onde a linguagem é customizada em tempo de execução. O sistema oferece ao usuário as possibilidades de definir novos materiais associados às estruturas e oferece um conjunto de cores para representá-los. Dois problemas foram verificados nesta codificação em tempo de execução. Um deles é relativo ao excesso de cores utilizadas, dificultando a compreensão do usuário. O sistema utiliza nove cores na linguagem expressiva e oferece mais dez cores para codificação de atributos. O conjunto de dezenove cores excede as marcas mais aceitáveis de tolerância cognitiva do usuário, de retenção de informação, ou seja, ele não consegue reter o significado associado a cada uma das cores [Miller 56]. O outro problema é relativo à disponibilização de cores iguais ou muito próximas de outras já utilizadas na codificação de outras partes da aplicação. Por exemplo, o sistema sinaliza a seleção de faces através de um hachurado vermelho. A cor vermelha foi disponibilizada para ser associada a um tipo de atributo aplicável à face. Quando o atributo está associado à face, ela é representada em vermelho e, consequentemente, quando selecionada, a sinalização de seleção se sobrepõe à representação visual do atributo, passando despercebida. Ambos os problemas poderiam ser evitados através de uma averiguação mais rigorosa da codificação linguística visual.

Sobre a Textura

A textura também é um atributo sobre uma forma, assemelhando-se à cor na sua característica de expressar atributo do objeto. Suas características plásticas estão próximas da própria forma. A textura nada mais é do que um preenchimento de uma área com pequenas formas, criando um padrão visual homogêneo. Tanto que se pode aplicar a cor sobre a textura, com cada uma delas traduzindo significados diferentes.
Semelhante à cor, a textura sofre restrições com relação à forma sobre a qual é aplicada, atuando melhor sobre formas com superfície, ou seja, formas facilmente perceptíveis. Para formas como linhas, a textura não produz um efeito visual tão marcante. Por exemplo, linhas tracejadas seriam um tipo de textura, mas podem causar desconforto visual ou confusão com relação à compreensão da forma.
A textura também pode ser usada para compor imagens realistas, criando, através de um adensamento de textura, a noção espacial de perspectiva.

Sobre o Tom

É comum considerarem-se os vídeos monocromáticos como "não coloridos". Esta é uma postura meramente cultural, ditada pelas características do hardware, onde as tonalidades de cinza são obtidas a partir de diferentes intensidades do feixe de elétrons sobre uma única camada de fósforo. Mas para efeito da percepção humana os tons percebidos no vídeo são decorrente da presença da cor cinza, ou seja, os vídeos monocromáticos representam em tons da cor cinza.
O tom é uma composição sobre o recurso cor, ele só pode ser obtido a partir da aplicação da cor. De modo geral o tom é um elemento adequado à expressão da idéia de transição entre dois extremos, posições físicas ou estados, ou seja, processos contínuos. Um exemplo de aplicação é na sinalização de diferenças progressivas na transição de um estado para outro, utilizado em pós-processadores. Este recurso permite demonstrar, por exemplo, níveis progressivos de tensão sofridos por uma estrutura; a diferença de tons sinaliza a diferença progressiva de tensão. Neste caso, duas idéias podem ser comunicadas: a passagem de um nível de tensão para outro e o sentido da alteração (de menor para maior tensão e vice-versa).
O tom pode sinalizar andamento de processos, embora neste caso seja conveniente que ele venha associado a outro recurso. Isto porque o usuário, não tendo noção dos tons iniciais e finais, não tem como saber exatamente em que ponto do processo ele se encontra. O tom pode ser usado com o simples objetivo de realçar algum elemento na composição visual. Por exemplo, vemos melhor alguma coisa que está próxima de seu contraste, ou seja, vemos melhor um objeto escuro quando está circundado pelo claro, e vice-versa. Ele pode ainda ser usado de forma semelhante à textura, para expressar a noção de perspectiva, sendo a perspectiva uma transição entre um ponto mais próximo e um mais distante.
Como atributo, o tom é um elemento dúbio por dois motivos. Primeiro, pelo fato de diferentes tons serem uma mesma cor, o usuário pode prioritariamente associar o atributo à cor, tendo dificuldade em associar diferentes tons a difentes atributos. Segundo, porque o usuário pode ter dificuldades perceptivas para identificar os diferentes tons.

Sobre o Brilho

O brilho sobre o qual estamos tratando é um elemento característico do ambiente computacional, mais especificamente dos terminais de vídeo. Ele é produzido pelo aumento de luminosidade na superfície do vídeo.

Este recurso sofre algumas restrições de aplicação. Por exemplo, em superfícies muito pequenas ele pode passar despercebido e em superfícies muito grandes pode causar desconforto visual ao usuário. É um recurso adequado para uso intermitente, como por exemplo chamar a atenção para um elemento do traçado em um determinado contexto não contínuo. Se utilizado de forma contínua, o recurso do brilho pode ser absorvido pelo usuário, que passa a não se dar conta de contrastes nesta dimensão.

Tipos Expressivos do Sistema

- Forma
- Cor
- Textura
- Tom
- Brilho

Resumo 4.4. Sistema de códigos expressivos do sistema

Os tipos expressivos identificados para o sistema estão relacionados no Resumo 4.4. No sistema de códigos expressivos do sistema não foram identificadas regras de estruturação, portanto o sistema de códigos coincide com os tipos discretizados.

4.3.3. A Codificação das Correlações entre Conteúdo e Expressão do Sistema

A correlação entre os elementos de conteúdo e os elementos expressivos do sistema será de fato definida pelo projetista no momento de especificação de sua LV particular. A seguir serão apresentadas algumas questões que devem ser observadas neste processo de especificação e algumas diretivas de orientação para uma utilização mais adequada do recurso visual. Via de regra, os recursos visuais não permite ao projetista expressar todo o conteúdo do sistema simultaneamente. Ele deverá então selecionar o melhor conteúdo a ser expresso através do código visual.

Os seguintes pontos deverão ser observados: a interação entre os recursos visuais, a associação de atributos em função de suas características e a priorização de conteúdo a ser expresso. Cada ponto será detalhado a seguir. Ao final será apresentado o conjunto de linhas de orientação gerais para especificação do código visual.

Sobre a Interação dos Recursos Visuais

Os recursos visuais básicos sobre os quais vamos atuar são: forma, cor, textura e tom. Estes recursos têm uma ordem natural de utilização, conforme Figura 4.6, determinada pela sua própria natureza. A forma é um recurso básico, que chamaremos de recurso primário, de sua presença depende a utilização dos outros recursos, que chamaremos de recursos secundários. Existem ainda diferentes relações de aplicação entre os recursos secundários. Por exemplo, o recurso primário da cor é a forma, ou seja, a aplicação da cor depende basicamente da forma, embora ela também possa ser aplicada sobre a textura. Ao mesmo tempo a cor é o recurso primário do tom, porque ele só pode ser obtido a partir da aplicação da cor.

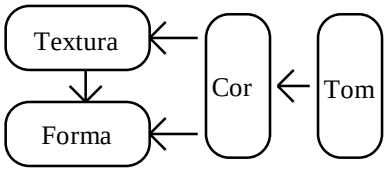


Figura 4.6. Nível de dependência na aplicação do código expressivo

Sobre a Associação de Atributos em Função de suas Características

Nos SMG a forma é sempre um elemento de distinção. Portanto, os atributos do objeto deverão ser expressos através de recursos que se aplicam sobre a forma. No processo de lexicalização devem ser observadas as características dos atributos a serem expressos, buscando-se associá-los a recursos expressivos adequados às suas próprias características. Por exemplo, a cor é um recurso expressivo adequado para conteúdos dinâmicos, como alterações de estado, ou seja, alterações de estado podem ser adequadamente sinalizadas por alterações de cor.

	Forma	**Textura**	**Cor**	**Tom**
Estático	X	X	X	
Dinâmico			X	
Básico	X			

Secundário		X	X	X

Tabela 4.6. Caracterização dos tipos discretos de expressão do sistema

Entre os critérios de classificação possíveis podemos destacar: atributos dinâmicos ou estáticos, contínuos ou discretos. Por exemplo um atributo contínuo seria uma propriedade, como ser isolante; um atributo discreto seria uma força associada a um elemento de uma estrutura. A Tabela 4.6 mostra que tipos de atributos podem ser expressos por que tipos de recursos visuais. Outros atributos diferentes dos apresentados podem ser considerados. Na Tabela 4.6 deve ser observado que a textura é um recurso secundário da forma, a cor é um recurso secundário da forma e da textura, uma vez que pode ser aplicada sobre ambas, e o tom é um recurso secundário da cor, conforme Figura 4.6.

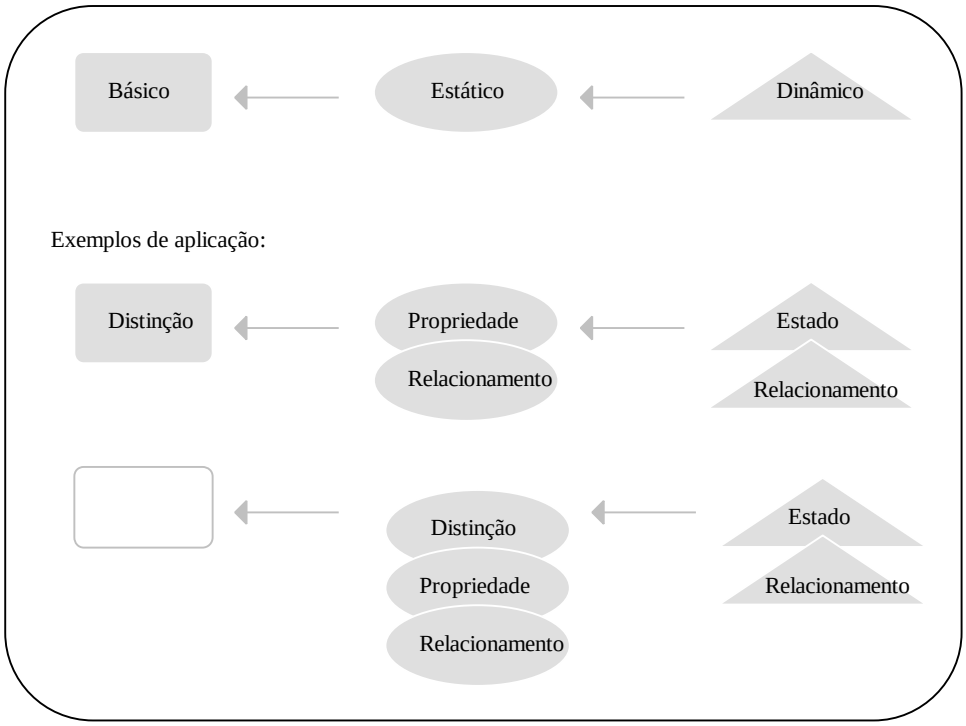


Figura 4.7. Níveis de prioridade na associação de atributos aos recursos expressivos

Algumas prioridades de associação devem ser observadas: sobre o elemento básico devem ser expressos primeiro os atributos estáticos e em seguida os dinâmicos. A Figura 4.7 mostra níveis de prioridade na associação entre elementos estáticos e dinâmicos, considerando um caso em que a forma (elemento básico) é um elemento de distinção e um caso em que não é (apenas um repositório). É assinalada também uma prioridade entre os elementos de cada nível.

Sobre a Priorização de Conteúdo a ser Expresso

Na decisão sobre que tipo de conteúdo deve ser expresso a cada momento, o projetista deve considerar a unidade de discurso e o foco de cada unidade. Se for sinalizado adequadamente ao usuário que ele está entrando em uma unidade tópica de discurso, a interface poderá ser contextualizada, com informações específicas referentes àquela unidade. A contextualização poderá ser feita inclusive no que se refere à sobreposição de recursos expressivos. Por exemplo, suponha-se um objeto representado na cor vermelha, onde esta cor representa um determinado tipo de material. Se o usuário entrar em uma unidade tópica de edição da geometria desse objeto, durante um certo período de tempo a informação de material passa a ser irrelevante. Portanto, dentro da unidade tópica de edição a cor vermelha que representa o material poderá ser ignorada e o recurso cor poderá ser utilizadas sobre a forma do objeto para representar outros atributos, por exemplo estados temporários. A contextualização permite aliviar o excesso de conteúdo representado.

4.3.4. Diretrizes de Orientação para Codificação da Linguagem Visual

Com os critérios até agora apresentados para a LV, a seguir são apresentadas as diretrizes de orientação para codificação da linguagem visual (DOCLV). As justificativas serão apresentadas de forma abreviada, uma vez que as diretrizes praticamente resumem o que foi discutido com relação à linguagem do sistema.

(a) Associação única entre conteúdo e meio de expressão.

Um mesmo recurso não deve ser utilizado com propósitos diferentes. Por exemplo, a cor não deve ser utilizada simultaneamente para simplesmente colorir e para expressar conteúdo do sistema. A associação entre conteúdo e expressão deve ser coerente - respeitando traços semânticos (conteúdos semelhantes, recursos semelhantes).

(b) Associar o recurso expressivo em função das características do conteúdo
Ao fazer a correlação entre conteúdo e recurso expressivo o projetista deverá observar as características do conteúdo, conforme Tabela 4.6.
(c) Observar a sobreposição de recurso expressivos sobre a forma
O projetista deve observar a ocorrência de excesso de recursos expressivos sobre a forma, de tal maneira que possam confundir o usuário.
(d) Priorização do conteúdo a ser expresso em função da unidade tópica de discurso
Como os recursos visuais se sobrepõem sobre a forma, o conteúdo que o projetista pode expressar sobre uma mesma entidade, simultaneamente, é limitado. No processo de escolha do que expressar a cada momento ele deve considerar a unidade tópica de discurso, em fnção da tarefa.
(e) Na escolha do meio de expressão, respeitar seus níveis de dependência.
Se algum nível de dependência de código (Figura 4.6) for saltado, os efeitos devem ser cuidadosamente avaliados. Por exemplo, se for utilizada uma textura colorida com o significado expresso através da cor e não da textura, ou se for usado um tom com o significado expresso através do tom e não da cor, deve ser cuidadosamente avaliado se o usuário perceberá esta opção de projeto.
(f) Observar a cardinalidade do conjunto de significados expressos.
Ao optar por um recurso de expressão, para um tipo ou instância, observar a cardinalidade do conjunto de significados a serem expressos. Por exemplo, se a cor estiver caracterizando um tipo, deve ser observada a cardinalidade do conjunto de tipos que usará a cor como meio de expressão. O uso de recursos como a cor deve ser feito para uma cardinalidade baixa, com o número mágico de Miller em mente (mais ou menos 7) [Miller 56, Marcus 92]. Acima deste valor de referência a codificação fica sujeita a um número muito grande de arbitrariedades, e o usuário, sem nenhuma referência cognitiva, provavelmente terá dificuldade em apreender o código.

O único recurso que permite expressar um conjunto de maior cardinalidade é a forma, porque a associação entre conteúdo e forma não é necessariamente arbitrária, nos casos em que a forma remete a alguma coisa concreta a associação significativa fica facilitada (isto é, quando há uma natureza icônica ou indicial na formação do signo).
(g) Observar limites de utilização do código
Os recursos visuais não são suficientes para expressar todo o modelo funcional do sistema, portanto recursos complementares (como textuais) serão necessários.

5. Estudo de Caso sobre Interfaces Gráficas por Manipulação Direta

Neste capítulo é apresentado um estudo de caso envolvendo quatro aplicativos gráficos com interfaces por manipulação direta. O estudo teve por objetivo analisar o uso de recursos expressivos visuais em aplicativos disponíveis no mercado comercial. A análise foi desenvolvida com base em ferramentas já existentes, para especificação de linguagens de interface. Dois aspectos foram observados com maior ênfase: a adequabilidade das ferramentas utilizadas, para análise e especificação de LVs, e a consistência da LV.

5.1. Sobre o Objetivo deste Estudo e as Condições de Análise

Neste capítulo será apresentada uma análise sobre a LV na interação usuário-sistema, em aplicativos gráficos com interfaces por MD. O corpo de análise foi selecionado entre aplicativos disponíveis no mercado comercial, com o objetivo de caracterizar situações reais de LVs por MD. Dois critérios básicos foram utilizados na seleção do corpo de aplicativos para análise: o tipo de manipulação (orientada à representação ou à construção) e o tipo de modelagem geométrica (de artefatos geométricos ou objetos naturais).

Quanto ao primeiro critério, em aplicativos onde a representação gráfica é o objetivo fim (caso dos desenhadores), a manipulação do traçado não está comprometida com nenhum modelo semântico. Neste caso a manipulação da representação gráfica é orientada à representação. Em aplicativos onde a representação gráfica é um meio para captura de dados, existe um compromisso da mesma com o modelo semântico do domínio, no sentido de manter a consistência dos dados. Apesar deste compromisso, não necessariamente o modelo semântico estará ativo durante a captura. Em alguns sistemas a interpretação semântica é associada à representação visual após o término completo do traçado. Nos sistemas de captura de dados que não têm interpretador semântico ativo durante a interação também se caracteriza uma manipulação orientada à representação, semelhante aos desenhadores. Naqueles sistemas que têm o modelo semântico ativo caracteriza-se uma manipulação orientada à construção. Portanto, temos três tipos de sistemas/interação: desenhadores orientados à representação, sistemas de captura de dados orientados à representação e sistemas de captura de dados orientados à construção.

O segundo critério contempla particularidades de edição, comparando sistemas que fazem captura de dados orientados à representação (sem modelo semântico). Mesmo não havendo validação semântica simultânea ao processo de edição, em alguns casos identifica-se um compromisso indireto das operações de edição com o domínio do sistema. Por exemplo, em um SMG voltado para aplicações geográficas (objetos naturais), o traçado tem por base um modelo de cartografia; algumas operações de edição são adequadas a este modelo, como por exemplo o apagamento por área (apagamento parcial de objeto) e seleção de objetos por área (dentro e fora). No apagamento por área, uma área é determinada e seu conteúdo é apagado desconsiderando a continuidade do traçado além dela.

Em um SMG para artefatos de engenharia os objetos são tratados individualmente, a supressão de parte de um objeto pode alterar sua natureza, invalidando-o. Nos sistemas cartográficos, o traçado é tratado como o traçado de um mapa, no qual a supressão de uma parte não afeta o restante, podendo significar apenas o desinteresse do usuário por aquela parte num determinado momento. Neste caso, na seleção de objetos podem ser selecionados os objetos dentro de uma área, fora de uma área ou começando em uma área e continuando fora dela. Por exemplo, o traçado de um rio pode começar em uma área e terminar fora dela. Um usuário que esteja fazendo um estudo sobre vazão de água pluviais pode selecionar apenas os rios que estejam dentro desta área; ou aqueles que começam nesta área e terminam fora. Em ambos os casos os conceitos de área e de objeto são fortemente vinculado à manipulação cartográfica. Este compromisso descaracteriza a etapa de edição destes sistemas como sendo meras etapas de desenhadores.

Através da análise de sistemas com diferentes modelos conceituais e diferentes formas de interação é possível avaliar a necessidade expressiva do sistema e a capacidade expressiva da LV. A análise das linguagens visuais de interface frequentemente denuncia um hiato entre a concepção sugerida pela representação gráfica e a concepção semântica implementada no sistema.

O Corpo de Aplicativos para Análise

Como desenhador foi selecionado o Microsoft Draw [Word 95], por ser um desenhador bastante conhecido entre usuários em geral. Os desenhadores são importantes nesta análise porque têm uma presença bastante forte na cultura do usuário em termos de MD. O usuário tende a generalizar o conhecimento adquirido na interação com estes sistemas, transportando-o para outros sistemas por MD, inclusive aqueles com manipulação orientada à construção.

Como um SMG voltado para artefatos de engenharia foi selecionado o pré-processador de elementos finitos Mtool - Bidimensional Mesh Tool [Mtool 97]. Os pré-processadores fazem captura de dados através da representação gráfica, para análise e simulação de estruturas. O Mtool é um sistema desenvolvido no meio acadêmico, em convênio entre o TecGraf/Puc-Rio e o Cempes/Petrobrás. É um sistema com menor penetração comercial, mas que tem a vantagem de adotar um paradigma de interação orientada à construção. Este paradigma é uma tendência mais nova no mercado e os sistemas acadêmicos, como frequentemente ocorre, foram pioneiros nesta abordagem.

Como um sistema de grande penetração comercial, amplamente divulgado na área de engenharia, foi escolhido o AutoCAD [AutoCAD 96]. O AutoCAD é um modelador geométrico que surgiu como um desenhador e foi evoluindo para um sistema de captura de dados. Ele adota um paradigma de interação orientada à representação, o qual permite ao usuário manipular livremente o traçado, associando a semântica posteriormente. Funções acopladas ao aplicativo permitem que o objeto traçado seja tratado através de modelos semânticos como por exemplo um modelo de elementos finitos. Neste paradigma a manipulação do traçado é mais flexível que no paradigma adotado para o MTool, mas existe o risco de o interpretador semântico não reconhecer o traçado no momento da associação, em função de inconsistências introduzidas pelo usuário. Uma análise mais detalhada sobre as vantagens e restrições de cada paradigma pode ser encontrada em Martins et alii (94, 95).

E, finalmente, como um SMG voltado para a modelagem de objetos naturais foi escolhido o MGE [MGE 96]. O MGE é um sistema para tratamento de informações geográficas. Ele permite que as informações sejam capturadas

através de sistemas de imagens e sejam editadas, permite também que as imagens sejam construídas através de um editor CAD. Neste sistema o editor gráfico trabalha com um modelo conceitual voltado para aplicações geográficas. A Tabela 5.1 esclarece o tipo de sistema e o tipo de interação para cada sistema do corpo de análise.

	Tipo de Sistema	**Objeto do Domínio**	**Tipo de Manipulação**
MsDraw	desenhador		orientada à representação
AutoCAD	captura de dados	artefatos de engenharia	orientada à representação
MGE	captura de dados	informações geográficas	orientada à representação
MTool	captura de dados	artefatos de engenharia	orientada à construção

Tabela 5.1. Sistemas de composição do corpo de análise

O Conjunto de Operações para Análise

A manipulação do traçado gráfico foi avaliada em termos da tarefa conceitualizada por um usuário. O objetivo é detectar se o sistema permite que um usuário execute a tarefa que conceitua sobre a imagem que vê. Para isto, a autora observou o comportamento de cada aplicativo para um conjunto de cinco operações. São elas: criar, apagar, copiar, mover e alterar forma. A observação feita pela própria autora justifica-se pelo fato de estarem sendo observadas operações básicas, em situações de uso também básicas, e a observação não estar considerando aspectos subjetivos. Por exemplo, para observação do comportamento dos sistemas foram conceitualizadas tarefas básicas como criar uma primitiva ou apagar uma primitiva. A observação foi documentada de forma sistemática, para que fosse permitido avaliar comparativamente o comportamento dos sistemas.
As operações de criação e apagamento foram escolhidas por serem tarefas básicas. A primeira dá origem à representação gráfica e é executada por MD, a segunda obviamente destrói a mesma representação gráfica. A criação do traçado gráfico é feita através das primitivas de interação, as quais têm o objetivo de auxiliar o trabalho do usuário antecipando as formas mais utilizadas. O critério de lexicalização adotado para as primitivas de interação varia entre os aplicativos. Esta questão está apresentada na Seção 5.2.3. A operação de apagamento em geral não envolve apenas MD, parte dela é ativada por menu, mas é extremamente interessante no sentido de que sua execução depende da forma como a representação gráfica está sendo interpretada. Formas visualmente semelhantes podem ser interpretadas de maneira diferente pelos sistemas, de modo que as condições de execução são diferentes. As operações de movimentação e alteração de forma foram escolhidas porque são operações feitas genuinamente por MD e porque sua execução também depende da forma como a representação gráfica está sendo interpretada.
Os exemplos selecionados não têm o intuito de avaliar os aplicativos em si. Eles foram selecionados com o objetivo de ilustrar situações genéricas de ocorrência frequente na LV, e sobre as quais a notação proposta neste trabalho se propõe a atuar.

Os Critérios Adotados para Análise e os Objetivos a Serem Atingidos

A LV deve permitir aos agentes da comunicação expressarem as abstrações sobre a interação através do objeto visual. Com base neste princípio, os critérios que nortearam a análise foram: (a) avaliar a expressão do modelo funcional do sistema, (b) distinguir o conteúdo expresso do recurso expressivo utilizado e (c) avaliar a alternância de vez entre os agentes da comunicação, na unidade tópica do discurso.
Quanto ao primeiro critério, o modelo funcional caracteriza o comportamento do objeto. A expressão adequada do modelo funcional permite a consistência entre o comportamento que a representação visual sugere e o comportamento real do objeto. Por exemplo, a representação visual de um polígono potencializa para o usuário a possibilidade de operações coerentes com as propriedades que caracterizam um polígono, como por exemplo o preenchimento de área. Mas se o polígono foi traçado através de um conjunto de linhas, ele poderá ter um comportamento coerente com um conjunto de linhas e incoerente com a sua aparência visual. A expressão adequada do modelo funcional é importante no sentido de que o usuário conceitua a tarefa a ser executada, sobre a representação visual. Como foi dito no Capítulo 2, na ultrapassagem do golfo de comunicação a expressão adequada do modelo funcional facilita a tradução semântica do usuário.
Buscando avaliar a expressão do modelo funcional, para cada uma das operações foram registrados o comportamento do sistema e o comportamento dos dados. O primeiro através da expressão dos tipos discretos (estados do sistema, pré-requisitos e especifcação do objeto) e o segundo através da expressão dos seus atributos.
Quanto ao segundo critério, a distinção entre conteúdo e recurso expressivo é importante como um princípio de estruturação dos elementos de formação da linguagem. Como também foi visto no Capítulo 2, a especificação adequada da LV dentro de um enfoque semiótico pressupõe um tratamento distinto destes elementos. Por exemplo,

quando o recurso cor (amarelo, azul e outras) é utilizado para expressar propriedades de um objeto (passa calor, isolante e outras) é gerado um código associativo a nível de tipo de conteúdo e tipo de recurso expressivo. A associação a nível de tipos permite uma associação posterior a nível de instâncias, de forma sistemática. Pelo lado do sistema, o uso regular do recurso expressivo permite que conteúdos semelhantes sejam expressos de forma semelhante. Pelo lado do usuário, o uso regular do recurso expressivo facilita a ativação do sistema (tradução articulatória).
Quanto ao terceiro critério, a observação da LV no diálogo é importante por dois aspectos. Um deles é a avaliação do retorno do sistema às ativações do usuário. Este é um aspecto genérico que também poderia ser observado em linguagem textual. O outro refere-se à sobreposição de recursos expressivos, este é um aspecto particular à LV. Como foi visto no Capítulo 4, na LV os recursos expressivos se sobrepõem ao recurso básico forma, algumas vezes interferindo um com o outro de forma negativa.
A seguir será apresentada a análise sobre os aplicativos. Neste capítulo serão apresentados apenas resumos e conclusões, o relatório completo das observações sobre os aplicativos encontra-se no Anexo A. Sobre a operação de criação será apresentada uma análise mais detalhada, com a especificação da gramática através da TAG [Payne & Green 86], e eventualmente através da D-TAG [Payne 91], e com uma discussão sobre as primitivas de interação. Para as outras operações serão apresentados apenas os resultados, obtidos por tratamento semelhante.

5.2. Análise Sobre a Operação de Criação

Diferentes formas de análise foram utilizadas para a operação de criação, dado que cada uma delas cobre aspectos parciais da linguagem. Uma só forma não é suficiente para averiguação dos três aspectos definidos no critério de análise. Portanto, a análise sobre a operação de criação será feita em três etapas. Na primeira, a análise será desenvolvida sobre as especificações da TAG e da D-TAG para a operação de criação. Na segunda, a análise será feita sobre a tabulação dos dados da operação, apresentada no Anexo A. Na terceira, a análise será principalmente por observação. Em alguns casos será utilizada a especificação da operação pelo modelo GOMS - Goals, Operators, Methods and Selection Rules [Card et alii 83].
Através da TAG, por exemplo, podemos avaliar aspectos de tradução semântica. A D-TAG permite avaliar parte dos aspectos de expressão do sistema. Em ambas, serão encontradas especificações iguais para sistemas cujas representações visuais têm comportamentos diferentes. A tabulação busca sistematizar dados da LV que não são tratados pela gramática. Ela permite melhor averiguação de aspectos de regularidade da linguagem, tanto pelo lado do usuário como pelo lado do sistema. Mesmo assim, muitas vezes na tabulação dos dados as diferenças registradas são pequenas, enquanto as diferenças de comportamento são grandes.
A terceira etapa da análise será feita principalmente por observação, em alguns casos utilizando especificações pelo modelo GOMS. Será averiguado o comportamento dos sistemas, especialmente nas operações de apagamento, movimentação e alteração de forma. Nesta etapa da análise o objetivo maior é demonstrar o hiato que ocorre entre a representação visual e o comportamento do objeto gráfico, quando representações visuais semelhantes, com manipulações semelhantes, têm comportamentos diferentes.

5.2.1. Especificação da LV Através da TAG

A TAG foi adotada por ser uma gramática orientada a tarefas. A TAG permite um mapeamento das intenções do usuário (tarefa) em procedimentos do sistema. Ela permite a especificação desde a intenção do usuário até o nível de ativação do sistema,. Ela é geralmente utilizada para o tratamento de interfaces textuais, desfrutando de reconhecimento e aceitação bastante grandes no meio acadêmico. No Exemplo 5.1 foi especificada a TAG para os quatro aplicativos. A indentação e o alinhamento dos componentes especificados teve por objetivo facilitar a leitura e a comparação entre as especificações referentes aos quatro aplicativos. No Exemplo 5.1 (b) a ação interrupção-criação, por não ter nenhum parâmetro a ser valorado, poderia estar especificada dentro da ação Criar. Porém, foi especificada em separado também com o intuito de facilitar a leitura.

```
Criar objeto (Operação=Criar, Objeto)  -->      ação-seleção [Objeto] +
                                                        ação-criação [Objeto]
ação-seleção [Objeto=linha]  -->                Tipo=apontamento +
                                                        Lugar=primitiva-interação +
                                                        "pressionar-mouse"
ação-criação [Objeto=linha]  -->                Tipo=apontamento +
                                                        Lugar=ponto-inicial +
                                                        "pressionar-e-arrastar-mouse" +
                                                        Tipo=apontamento +
                                                        Lugar=ponto-final +
                                                        "soltar-mouse"
```

Exemplo 5.1 (a) Especificação da TAG para criação de linha no MsDraw

```
Criar objeto (Operação=Criar, Objeto)  -->   ação-seleção [Objeto] +
                                                  ação-criação [Objeto] +
                                                  interrupção-criação
ação-seleção [Objeto=linha]  -->             Tipo=apontamento +
                                                  Lugar=primitiva-interação +
                                                  "pressionar-mouse"
ação-criação [Objeto=linha]  -->             Tipo=apontamento +
                                                  Lugar=ponto-inicial +
                                                  "pressionar-mouse" +
                                                  Tipo=apontamento +
                                                  Lugar=ponto-final +
                                                  "pressionar-mouse"
interrupção-criação  -->                     "pressionar-botão-direita"
```

Exemplo 5.1 (b) Especificação da TAG para criação de linha no AutoCAD

```
Criar objeto (Operação=Criar, Objeto)  -->   ação-seleção [Objeto] +
                                                  ação-criação [Objeto] +
                                                  interrupção-criação +
                                                  interrupção-seleção
ação-seleção [Objeto=linha]  -->             Tipo=apontamento +
                                                  Lugar=primitiva-interação +
                                                  "pressionar-mouse"
ação-criação [Objeto=linha]  -->             Tipo=apontamento +
                                                  Lugar=ponto-inicial +
                                                  "pressionar-mouse" +
                                                  Tipo=apontamento +
                                                  Lugar=ponto-final +
                                                  "pressionar-mouse"
interrupção-criação  -->                     "pressionar-botão-direita"
interrupção-seleção  -->                     Tipo=apontamento +
                                                  Lugar=menu-não-criação +
                                                  "pressionar-mouse"
```

Exemplo 5.1 (c) Especificação da TAG para criação de linha no MGE

```
Criar objeto (Operação=Criar, Objeto)  -->   ação-seleção [Objeto] +
                                                  ação-criação [Objeto] +
                                                  interrupção-seleção
ação-seleção [Objeto=linha]  -->             Tipo=apontamento +
                                                  Lugar=primitiva-interação +
                                                  "pressionar-mouse"
ação-criação [Objeto=linha]  -->             Tipo=apontamento +
                                                  Lugar=ponto-inicial +
                                                  "pressionar-mouse" +
                                                  Tipo=apontamento +
                                                  Lugar=ponto-final +
                                                  "pressionar-mouse"
interrupção-seleção  -->                     Tipo=apontamento +
                                                  Lugar=menu-não-criação +
                                                  "pressionar-mouse"
```

Exemplo 5.1 (d) Especificação da TAG para criação de linha no MTool

Comparando-se os quatro aplicativos observamos dois tipos de iteração. Uma das iterações ocorre em função da ação-seleção. Uma vez que um tipo de primitiva tenha sido selecionada, várias instâncias da mesma primitiva podem ser traçadas em sequência, até que esse estado seja interrompido pelo usuário. A outra iteração ocorre em

função da ação-criação. Neste caso, o ponto final de um segmento de reta é automaticamente considerado pelo sistema como o ponto inicial do próximo segmento, a menos que o usuário interrompa o processo. Com relação ao primeiro caso, no MGE e no Mtool observamos que a operação de criação é executada em três passos básicos, com uma opção de iteração no passo intermediário: (a) a ativação do estado de criação (através da ação-seleção), (b) a especificação dos dados de entrada, a qual pode ocorrer de 0 a n vezes e (c) a interrupção do estado de criação (através da interrupção-seleção). Caso o usuário ative o estado de criação e desista em seguida, o passo intermediário não ocorrerá, reduzindo a operação a apenas dois passos, a ativação e a interrupção. No MsDraw e no AutoCAD não existe o estado de criação de uma primitiva, o usuário deve ativar o menu para cada primitiva traçada, mesmo que a mesma primitiva seja desenhada repetidas vezes.
Embora o registro desse tipo de iteração não faça parte do formalismo básico da TAG, poderia ser utilizada uma notação conforme Exemplo 5.2, onde fosse sinalizada uma iteração para a criação de instâncias interligadas do objeto, através de um indicador numérico colocado a seguir ao colchete. Esse indicador poderia variar de 0 a n. Procedimento semelhante poderia ser utilizado para sinalizar as iterações referentes à ação-criação. O uso deste indicador não afeta a TAG, quanto à característica de ser uma gramática livre de contexto.

Criar objeto (Operação=Criar, Objeto) --> ação-seleção [Objeto] +
ação-criação [Objeto]n +
interrupção-seleção

Exemplo 5.2. Especificação de iteração referente à ação-seleção

Ainda sobre os Exemplos 5.1 (a) a (d) observamos que a TAG não registra as informações e os retornos do sistema. Embora as formas de ativação sejam semelhantes, poderiam até ser iguais, os objetos criados têm comportamentos completamente diferente. Essa diferença não é registrada na gramática. As diferenças registradas na TAG referem-se às formas de ativação do sistema, clique ou pressão-e-arrasto, e à quantidade de ativações, uma a mais quando é necessário interromper o traçado.
A D-TAG é uma extensão da TAG que registra os aspectos estáticos da expressão do sistema. "*...D-TAG shows a role for the static aspects of the device's display, but it ignores the dynamics of device responses, ...*" [Payne 91, p. 147]. De acordo com a proposta da D-TAG, parte da expressão do sistema seria registrada, como por exemplo o desdobramento de um menu que fosse ativado na barra de menu principal. Porém, a D-TAG não preve registro para os retornos do sistema durante o processo interativo. Para analisar os retornos do sistema Payne utiliza a notação da árvore de interação [Payne 91]. Trata-se de uma notação visual, que permite registrar a alternância de agentes durante a interação.
Porque a TAG não registra os retornos do sistema, perde-se ocorrências que caracterizam diferenças de comportamento entre as primitivas. Por exemplo, a gramática não registra que no MsDraw e no MTool a última primitiva criada fica automaticamente selecionada, enquanto no MGE e no AutoCAD não ficam. Nos primeiros, a cada nova linha criada a anterior é desselecionada. No caso dos desenhadores a expressão da seleção é feita através de pequenos quadrados (pontos de manipulação) sobre a figura, os quais desaparecem quando a figura é desselecionada. No caso do MTool, a expressão do estado de seleção é feita através da cor vermelha. Quando a figura é desselecionada ela volta à cor original da primitiva ou atributo a ela associado. Na TAG não são registradas as alterações visuais sobre o objeto, como alterações de cor e formas de sinalização. No exemplo específico do MTool, a utilização do recurso cor para expressar a seleção afeta a condição original da primitiva, uma vez que a cor também é utilizada para expressar atributos padrão associados automaticamente a ela, no momento do seu traçado. Por exemplo, neste sistema toda aresta é traçada na cor cinza escuro, que significa um tipo especial de aresta. Porém, a aresta récem-traçada é vermelha, significando que está selecionada. Enquanto a aresta está selecionada a informação de atributo fica encoberta, em função da sobreposição de recursos expressivos. Em termos de uma especificação sistemática da LV é fundamental que esta sobreposição seja claramente sinalizada para o projetista.
O aspecto de tradução semântica, da intenção do usuário para o procedimento do sistema, fica bem registrado na gramática. Através dela observamos que a tarefa do usuário, traçar um elemento da representação visual, é diretamente traduzida em uma operação de escolha de uma primitiva de interação. O usuário não ativa uma operação de criação, ele escolhe diretamente uma primitiva de interação, ficando a operação de criação implícita na escolha da primitiva. Em princípio este hiato na tradução poderia significar um ônus cognitivo para o usuário, mas como este conceito é implementado em praticamente todos os aplicativos, já existe uma cultura na área referente a este tipo de ativação.
Em resumo, com relação aos objetivos de análise propostos, (a) a TAG registra adequadamente a tradução semântica entre a intenção do usuário e o procedimento do sistema, (b) a TAG não registra adequadamente a expressão do modelo funcional do sistema, porque não registra o retorno do sistema e consequentemente o comportamento das primitivas e (c) a TAG não registra a sobreposição de recursos expressivos. Portanto, no sentido de completar a análise vamos recorrer à tabulação dos dados, a qual é apresentada no Anexo A.

5.2.2. Análise sobre a Regularidade das Expressões Linguísticas

A tabulação dos dados facilitou especialmente a análise do uso do recurso expressivo, verificamos alguns pontos de regularidade e algumas diferenças entre os aplicativos. Por exemplo, com relação à expressão do usuário, no

traçado da primitiva apenas o MsDraw é ativado por pressão-e-arrasto, os outros três aplicativos são ativados por cliques. A interrupção do traçado, nos dois aplicativos em que ocorre, AutoCAD e MGE, é feita pelo botão da direita.
Na MD a regularidade da expressão do usuário, dentro do aplicativo e entre aplicativos, é importante porque a forma de ativação não está ligada a nenhum elemento da interface, e, portanto, o usuário fará a tradução articulatória de forma intuitiva. Por exemplo, existe um modelo de interação associado à idéia de botão, no qual o botão deve ser clicado, como se estivesse sendo pressionado. A presença visual do botão remete à idéia de "clique-me". No caso do traçado de linhas não existe nenhum elemento visual sugestivo. O único elemento sugestivo é metafórico, ligado idéia ao uso do mouse como uma extensão da mão. Portanto, neste caso a regularidade de expressão, através de um tipo expressivo adequado, é um elemento de reforço da metáfora, logo, é um forte elemento cognitivo.
Em alguns casos, no traçado das primitivas não existe nenhuma referência para o usuário sobre a forma de ativação. Por exemplo, o MsDraw permite o traçado de desenho livre e polilinha a partir da mesma primitiva de interação (forma livre), diferenciando-as pela forma de ativação. O desenho livre é traçado por pressão-e-arrasto e a polilinha por cliques sucessivos. Desenho livre e segmentos de retas podem ser misturados alternando-se a forma de ativação.
Observando a especificação deste exemplo através do modelo GOMS (Exemplo 5.3) percebe-se a dificuldade a ser enfrentada pelo usuário tanto na tradução semântica quanto na articulatória. No primeiro caso o usuário deve traduzir a intenção de gerar uma polilinha para a ativação de uma primitiva de interação de desenho livre. No segundo caso ele deverá "descobrir" a diferença entre as formas de ativação, no sentido de produzir o efeito desejado. Conceitualmente a polilinha e o desenho livre não têm nada em comum. Provavelmente o seu agrupamento em uma única primitiva de interação está ligado a questões internas da estrutura de dados e algoritmos otimizados do sistema, uma vez que ambas as primitivas são internamente definidas por uma sequências de pontos, de tamanho variável. Neste caso, o que está sendo trazido para a interface é uma estrutura interna do sistema.

Goal: Criar polilinha
    Goal: Selecionar primitiva de interação
        Selecionar primitiva de desenho livre
    Goal: Traçar
        Informar dados por cliques

Exemplo 5.3 (a) Especificação do modelo GOMS para criação de polilinha no MsDraw

---

Goal: Criar desenho livre
    Goal: Selecionar primitiva de interação
        Selecionar primitiva de desenho livre
    Goal: Traçar
        Informar dados por pressão-e-arrasto

Exemplo 5.3 (b) Especificação do modelo GOMS para desenho livre no MsDraw
Com relação à expressão do sistema, na sinalização de seleção das primitivas podemos observar variações na forma de expressão entre os diferentes sistemas, para conteúdos semelhantes. Por exemplo, o MsDraw, o AutoCAD e o MGE oferecem pontos de manipulação semelhantes, sendo que o AutoCAD coloca a primitiva em linhas tracejadas. O MTool sinaliza a seleção através de cor. A diferença na expressão é coerente, uma vez que o modelo semântico do MTool não permite as mesmas manipulações de forma dos outros aplicativos. Se o Mtool utilizasse pontos de manipulação, pela coerência com os outros aplicativos estaria sugerindo uma operação não permitida.
Ainda com relação à expressão do sistema, podemos observar os cursores nas diferentes etapas da interação. Neste aspecto é onde encontramos as maiores variações. Por exemplo, em estado neutro o MsDraw utiliza um cursor em I e o AutoCAD utiliza um cursor em  com linhas de orientação que abrangem toda a tela, em estado de seleção o MGE utiliza um cursor em  e o MTool utiliza um cursor em  , conforme Figura 5.1. Em estado de criação o MsDraw utiliza um sinal de soma pequeno, o AutoCAD somente as linhas de orientação, o MGE um sinal de soma grande e o MTool um  . O MGE ainda tem a particularidade de alterar o cursor após a indicação do primeiro ponto do primeiro segmento de reta, quando o cursor passa de um sinal de soma grande para um sinal de multiplicação. O AutoCAD implementa a criação de linhas consecutivas, da mesma forma que o MGE, mas não altera o cursor durante o processo. É interessante observar que embora o tipo de cursor varie totalmente entre os aplicativos, o uso de uma forma diferenciada de cursor é uma constante, fato importante em termos de retorno para

o usuário. Em qualquer caso, quando o sistema entra em estado de criação o cursor é alterado. Mesmo que os cursores utilizados sejam diferentes, a cultura que é estabelecida entre os usuários é de que a variação de estado é sinalizada pela mudança do cursor.

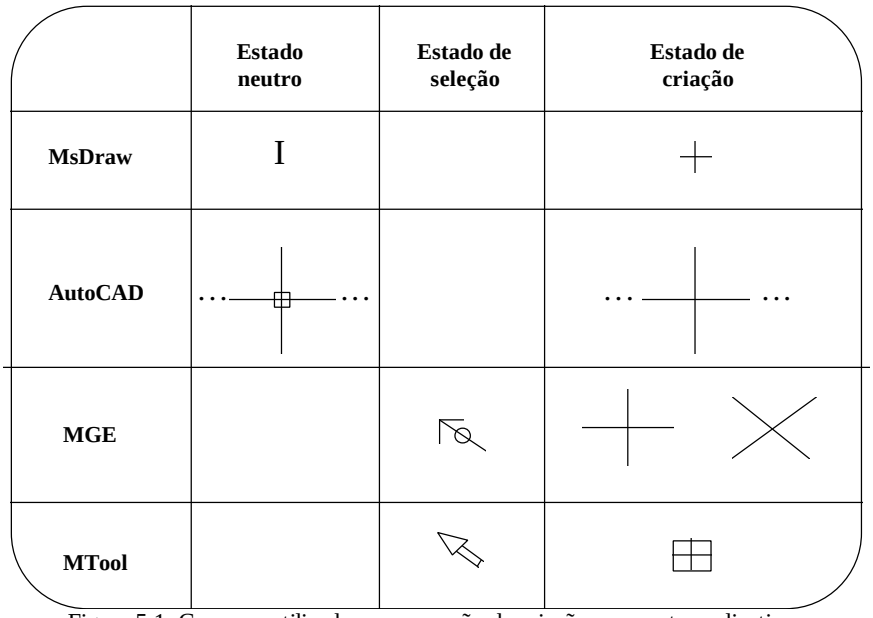


Figura 5.1. Cursores utilizados na operação de criação nos quatro aplicativos

Quanto ao número de passos de interação na operação de criação da primitiva linha, houve variações entre os quatro aplicativos (MsDraw - 7, AutoCAD - 10, MGE - 8, MTool - 9) em função das diferenças no modelo conceitual, relativas às possibilidades de iteração.

Em resumo, na análise dos dados tabulados observamos os seguintes aspectos, referentes à expressão do usuário e do sistema: (a) diferenças nas formas de ativação, para operações semelhantes, (b) diferentes formas de expressão dos sistemas, para situações semelhantes e (c) dificuldades na tradução semântica e articulatória.

5.2.3. A Expressão do Modelo Funcional

A análise desenvolvida no tópico anterior permitiu avaliar de forma mais detalhada as diferenças na forma de expressão do usuário e do sistema. Neste tópico o objetivo é avaliar se a expressão do modelo funcional é adequada e suficiente. Para isso, detalhes do detalhes do comportamento dos sistemas devem ser observados. Se o modelo funcional estiver adequadamento expresso através da interface, o objeto terá um comportamento coerente com aquilo que a representação visual sugere.

A Lexicalização das Primitivas de Interação

O primeiro aspecto a ser observado é com relação às primitivas de interação. Essa observação é interessante no sentido de se comparar o comportamento das primitivas do sistema com as primitivas de interação. A primeira instância onde se distingue a diferença entre o desenho e a concepção das primitivas é no processo de criação. Na criação de entidades por MD, para que o sistema possa interpretar corretamente a entrada de dados é necessária a especificação prévia do objeto da operação. Por exemplo, se o usuário clicar dois pontos na tela, eles pontos podem ser interpretados pelo sistema como sendo o início e o fim de um segmento de reta (Figura 5.2 (a)), ou podem ser interpretados como sendo o início e o fim do raio de uma circunferência (Figura 5.2 (b)). Portanto, previamente ao traçado o usuário deve especificar ao sistema a primitiva que deseja traçar.

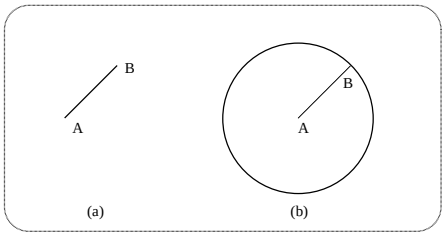


Figura 5.2. Diferentes interpretações do sistema para a mesma entrada de dados

As primitivas são as unidades elementares que compõem a representação gráfica. As primitivas básicas de traçado são o ponto e a linha. Para que o usuário não tenha que compor todas as figuras a partir das primitivas básicas, o sistema oferece, através do menu, um conjunto de primitivas de interação com algumas opções mais elaboradas

[Ziegler & Fähnrich 88]. As primitivas de interação objetivam facilitar o trabalho do usuário, antecipando suas necessidades o sistema transforma em léxico formas repetitivas [Rheinfrank & Evenson 96]. Por exemplo, um polígono pode ser traçado a partir de segmentos de reta, mas para facilitar o traçado gráfico a maioria dos editores gráficos oferece o polígono como uma primitiva de interação, permitindo ao usuário desenhar o polígono diretamente, nas dimensões que deseja.
As primitivas de interação nem sempre correspondem às primitivas do domínio do sistema. Por exemplo, o pré-processador MTool oferece como primitiva de interação o círculo. Uma vez traçado, o círculo transforma-se em uma face circular, delimitada por arestas, as quais são definidas por vértices. Uma vez traçada, a primitiva círculo deixa de ser reconhecida como tal, se o usuário tentar apagar um círculo ele não consegue. A representação gráfica passa a ser tratada pelo sistema em termos de faces, arestas e vértices. Para apagar a figura o usuário deve solicitar o apagamento da face, ou das arestas que a definem. Embora este aspecto seja parcialmente sinalizado na interface, ele não aparece na gramática, uma vez que esta não registra o retorno do sistema.
A lexicalização pode ser feita em diferentes níveis. O conjunto de primitivas pode se restringir a um nível de lexicalização bastante próximo das formas básicas, ou pode avançar em níveis maiores de complexidade de formas. O critério de lexicalização para as primitivas de interação pode objetivar apenas o traçado, ou pode objetivar construções semânticas mais elaboradas. A lexicalização pode ser feita sobre primitivas para composição de formas, como também para composições de formas reconhecíveis pelo modelo semântico do sistema (formas com conteúdo). Nos aplicativos avaliados só foram encontradas lexicalizações sobre formas.
A decisão sobre a lexicalização a ser adotada é uma negociação entre as vantagens e desvantagens que ela pode trazer para o usuário. Esta negociação gira em torno da facilidade de traçado versus inflexibilidade de tarefa. Quanto mais alto o nível das primitivas oferecidas, maior pode ser a facilidade de traçado, mas também maior será a inflexibilidade no que se refere à tarefa, uma vez que o usuário ficará restrito aos tipos já estabelecidos. Se as primitivas forem definidas em muito alto nível, o traçado do objeto gráfico fica mais fácil e mais restrito. Mais fácil no sentido de que o usuário pode traçar mais rapidamente a representação visual desejada, e mais restrito no sentido de que ele está preso às formas pré-especificadas. Se as primitivas forem lexicalizadas em mais baixo nível o processo de construção será mais trabalhoso, mas em compensação o usuário terá maior flexibilidade nas opções de construção. Se a lexicalização adotada determina a flexibilidade da linguagem, ela altera a capacidade de o usuário expressar suas intenções. Com uma linguagem extremamente inflexível o usuário pode não conseguir expressar suas intenções.
Uma alternativa que permite suavizar a decisão do projetista com relação ao léxico é a possibilidade de o usuário customizar suas primitivas de interação. Por exemplo, em aplicativos como MGE o menu de botões apresenta as primitivas supostamente mais utilizadas. A partir do menu principal o usuário pode entrar um submenu. Esta entrada para o submenu é customizável, ou seja, o usuário pode trazer para o menu principal uma das alternativas do submenu, substituindo aquela de entrada. Com isto, o usuário tem acesso a um conjunto de primitivas maior do que o inicialmente exposto, podendo adaptar o ambiente às suas necessidades. Outra forma de customização é o sistema permitir ao usuário a criação de suas próprias primitivas de interação, como se fosse macros. Desta forma, primitivas complexas poderiam ser disponibilizadas no menu, podendo ser facilmente replicadas pelo sistema. Esta opção permite ao usuário antecipar construções semânticas. Neste caso as primitivas seriam equivalentes àquelas do modelo semântico.
O fundamental é que o critério adotado fique claro para o usuário, para que ele possa realizar suas tarefas. Especialmente nos casos em que as primitivas de interação diferem das primitivas do domínio do sistema.

Construção da Representação Visual a Partir das Primitivas de Interação

Vamos fazer uma comparação entre representações gráficas similares editadas no desenhador MsDraw (Figura 5.3) e no pré-processador MTool (Figura 5.4). A escolha desses dois aplicativos deve-se a que o primeiro tem manipulação orientada à representação (manipula formas) e o segundo manipulação orientada à construção (manipula formas com conteúdo). Apesar da semelhança visual, as figuras são conceitualmente diferentes. Os pontos abaixo resumem esta diferença.

- Na figura editada no MsDraw as primitivas traçadas são tratadas de forma independente, ou seja, são primitivas sobrepostas. Quando o traçado das primitivas se intercepta, uma primitiva não interfere com a outra, mantendo-se independentes. A primeira figura contém quatro polígonos, traçados através de linhas. A figura no MsDraw ilustra claramente a sobreposição visual das primitivas. Ela foi construída através de quatro polígonos, dois azuis e dois brancos sobrepostos.

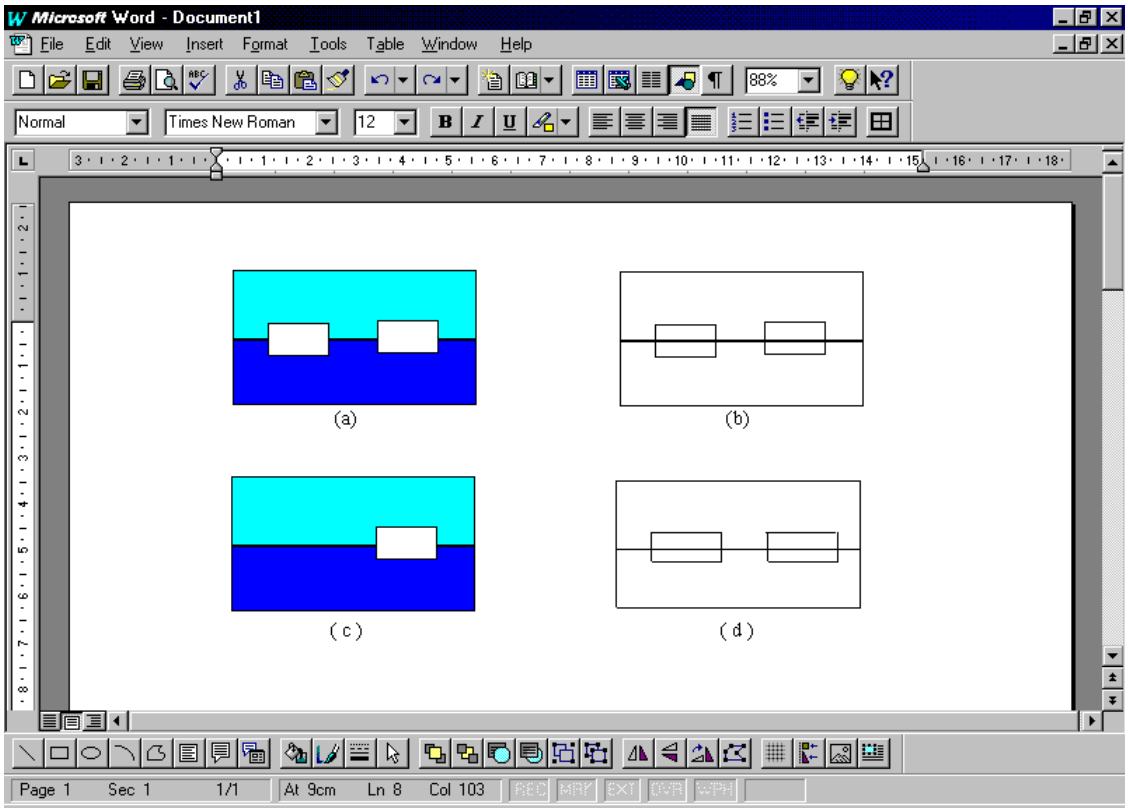


Figura 5.3. Repesentação gráfica editada no MsDraw

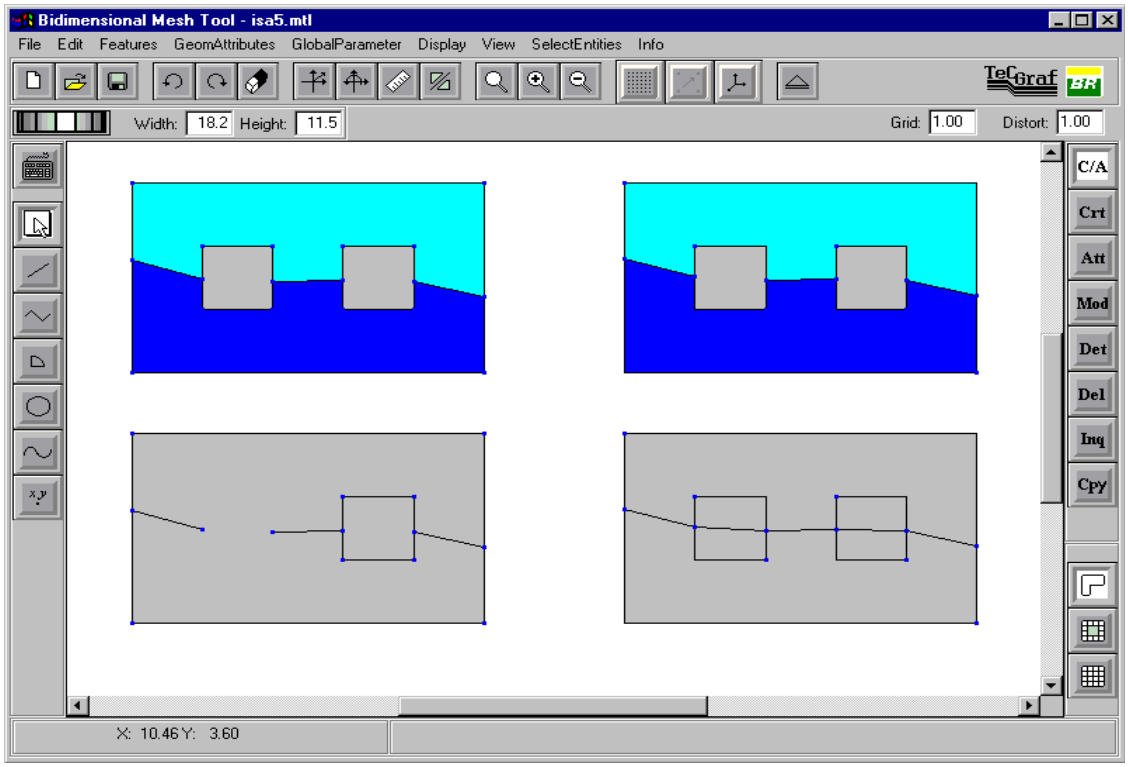


Figura 5.4. Representação gráfica editada no MTool

No MTool as primitivas traçadas têm um comportamento diferente das primitivas de interação. Uma vez traçadas, as formas geométricas oferecidas para interação transformam-se em primitivas semânticas. A linha

traçada é tratada como uma aresta, dois vértices são gerados nos extremos de cada aresta. Quando uma área é delimitada pelos vértices, ela é automaticamente reconhecida como uma face. Quando duas primitivas se interceptam, por exemplo duas linhas, elas se subdividem transformando-se em quatro segmentos de arestas adjacentes. Na figura editada no MTool as primitivas traçadas compõem um objeto. O sistema captura a geometria e topologia deste objeto. Visualmente esta diferença só é sinalizada por pontos que representam vértices, que podem ser observados na Figura 5.4.

- No MsDraw figuras visualmente semelhantes podem ser construídas através de polígonos (Figura 5.4. (b)) ou de linhas (Figura 5.4. (d)). Em cada caso, a natureza da primitiva de interação se mantém após o traçado, as linhas, mesmo formando um polígono, continuam a ser tratadas como linhas. A quebra na percepção visual do usuário pode ocorrer quando ele vê a representação visual de um polígono, construída através de linhas, tenta operá-la como um polígono e não consegue. Por exemplo, quando ele clica sobre uma das arestas para selecionar o polígono, apenas uma das arestas é selecionada. Ou ainda, quando ele tenta preencher o polígono e não consegue. Os polígonos visuais da Figura 5.4 (b) podem ser preenchidos, conforme Figura 5.4 (a), os da Figura 5.4 (d) não podem ser preenchidos porque são formados por linhas.
  No MTool a Figura 5.4 (a) foi construída através de linhas. A Figura 5.4 (b) foi construída através de polilinhas. Foram traçados três polígonos através de polilinhas (área de terreno e galerias), em seguida foram traçadas três linha para dividir o polígono maior em duas áreas (duas regiões de solo). Em ambos os casos, com linhas ou polilinhas, quando uma área é delimitada o sistema a reconhece automaticamente. A diferença visual entre as duas é que na primeira existem vértices suportando cada segmento de reta, enquanto na segunda existem vértices apenas nas diagonais dos polígonos. Em ambas, nos pontos onde os polígonos iniciais foram interceptados pelos três segmentos de reta que delimitam as áreas de solo foram gerados seis vértices. Na representação interna, embora a polilinha tenha sido interceptada e subdividida ela continua sendo reconhecida como uma polilinha.
  As Figuras 5.4 (a) e (b) poderiam ser construídas de três formas diferentes. A primeira, iniciando pela área grande (corte no terreno), definindo-se as áreas menores (galerias) e em seguida as camadas de solo. A segunda, iniciando pelas galerias, definindo-se as área de terreno e em seguida as camadas de solo. Estas duas formas de construir a representação visual, embora inversas, levam ao mesmo objetivo. Uma terceira forma, que seria bastante intuitiva, exigiria um passo de correção no traçado da figura. Nesta terceira, Figura 5.4 (d), é delimitada a área de terreno, são divididas as camadas de solo e em seguida é demarca a área das galerias. Neste caso, como pode ser observado, a área das galerias fica dividida ao meio, conforme a divisão das camadas de solo. Isto porque o traçado obedece a um modelo matemático, onde o usuário não tem como expressar a intenção de construir um buraco no terreno. Quando a área dos túneis é traçada, o modelo matemático entende que está sendo delimitada uma área no plano. Onde as arestas dos túneis interceptam a aresta que delimita as camadas de solo ambas são subdivididas, surgindo um vértice de apoio para os novos segmentos de reta criados. Neste caso ele tem que apagar a aresta do meio. Através de uma sequência de passos bastante natural e intuitiva o usuário produz uma representação visual errada, que necessita ser corrigida. Claro que este exemplo é bastante simples e esquemático. Em um exemplo mais complexo, corrigir a representação visual pode ser trabalhoso. Tem-se assim, uma evidência de que a manipulação está sujeita à estrutura interna do sistema. O usuário tem que operar não em função do que vê, mas em função das restrições do sistema.
- No MsDraw as primitivas obedecem uma hierarquia de desenho, sobrepondo-se à medida em que são desenhadas. Por exemplo, na Figura 5.3 (c) o traçado correspondente a uma das galerias foi apagado surgindo o traçado dos polígonos branco e azul que estavam por baixo. No MTool as primitivas semânticas obedecem a uma hierarquia matemática entre vértices, arestas e faces.
- No MsDraw as cores são meramente estéticas. Na especificação de cores e outros atributos estéticos as primitivas traçadas são tratadas em termos de interior e contorno. Esta diferença causa um hiato na linguagem de interação uma vez que a mesma representação visual é tratada de forma diferente , como pode ser observado no Exemplo 5.4. No entanto este hiato é compensado pela idéia de "preencher" uma primitiva. As primitivas traçadas, como por exemplo polígonos, podem ser preenchidas (fill) ou não preenchidas. A idéia de preenchimento remete a uma distinção entre uma fronteira delimitadora (o contorno) e o seu interior.

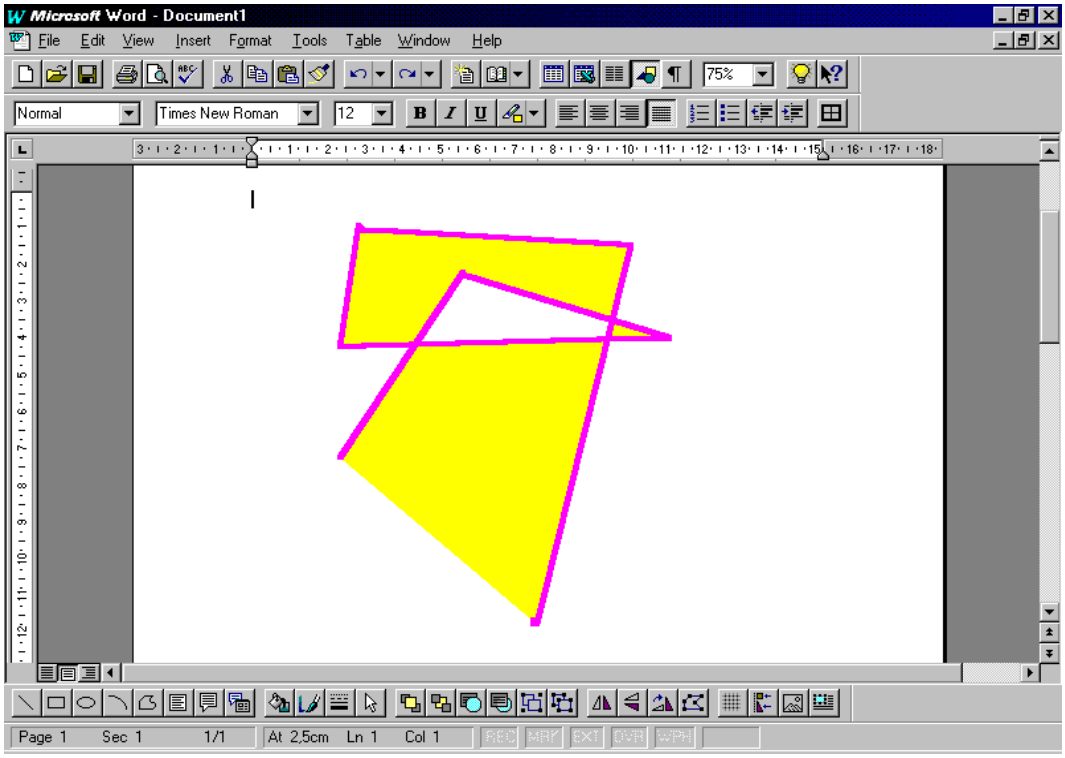


Figura 5.5. Poligonal aberta traçada com opção de preenchimento

---

Goal: Colorir Polígono
- Goal: Selecionar polígono
  - Selecionar polígono traçado
- Goal: Colorir interior
  - Colorir interior do polígono
- Goal: Colorir contorno
  - Colorir contorno do polígono

Exemplo 5.4. Especificação do modelo GOMS para atribuição de cor no MsDraw

Com relação à idéia de preenchimento, no MsDraw foi observada uma particularidade com relação à primitiva polilinha (ou desenho livre), onde o comportamento da primitiva não é coerente com seu aspecto visual. Esta primitiva pode ser preenchida. Como foi mostrado na Figura 5.5 (idêntica à Figura A.2), o sistema traça uma linha imaginária entre o primeiro e o último pontos, como um fechamento e preenche o espaço interno. Caso a primitiva se auto-intercepte o sistema adota algum tipo de critério, o qual não nos preocupamos em observar, determinando áreas internas e externas para preenchimento.

No MTool as cores representam atributos que foram posteriormente associados às faces da figura. No caso, as cores representam diferentes tipos de solo. Na Figura 5.4 (c) quando foi apagado o traçado correspondente a uma das galerias, o sistema perde a fronteira entre as duas regiões de solo, perdendo os correspondentes atributos.

Em suma, neste tópico observamos a importância da lexicalização adotada nas primitivas de interação. No caso do desenhador a estrutura interna dos dados equivale totalmente às primitivas utilizadas para traçado. No caso do pré-processadores a primitiva utilizada caracteriza parte da primitiva traçada (diferença entre o uso da linha e polilinha), mas não totalmente. Uma vez traçadas as primitivas de interação se transformam em primitivas do sistema, interagindo entre si. Outro aspecto observado é que no aplicativo orientado à construção (MTool) nem sempre a construção do objeto pode ser intuitiva, sob o risco de serem necessárias correções na representação gráfica traçada. O processo construtivo está em alguns aspectos restrito ao tratamento interno do sistema sobre a representação gráfica. Portanto, a tradução semântica da intenção do usuário, conceituada sobre a representação visual, está sempre sujeita às restrições do modelo semântico, as quais nem sempre estão expressas.

5.3. Análise sobre Operações de Apagamento e Movimentação

Da mesma forma que no tópico anterior, o objetivo aqui é observar a complexidade do modelo funcional dos sistemas, no que se refere às operações de apagamento e movimentação. Através dessas operações podemos observar de forma especial a expressão do modelo funcional. Muitas vezes, nestes casos, o usuário tenta executar uma operação sugerida pela representação visual e a operação é bloqueada pelo modelo semântico ativo.

5.3.1. Diferenças de Comportamento entre Operações de Apagamento e Movimentação

Entre os aplicativos orientados à representação observamos uma regularidade nas operações de apagamento, uma vez que as primitivas traçadas são independentes. A diferença fica por conta do MTool por ser orientado à construção. Esta questão pode ser esclarecida comparando-se as Figuras 5.3 e 5.4. Embora as Figuras iniciais sejam semelhantes, cada aplicativo produz uma figura final diferente quando o traçado equivalente a uma das galerias é apagado (Figuras 5.3 (c) e 5.4 (c)). No MsDraw, quando o traçado equivalente a uma das galeria foi apagado, aflorou a figura que estava por trás. A representação gráfica é uma composição de quatro polígonos, sendo dois brancos sobrepostos a dois coloridos. Quando um dos brancos foi apagado, o polígono colorido apareceu. O MTool trata as propriedades geométricas e topológicas do objeto. Quando o traçado da galeria é apagado, o sistema não consegue recuperar a fronteira entre as camadas de solo. Neste caso, os tons de azul da figura, que representam os diferentes tipos de solo, são eliminados. Ao perder a fronteira o sistema perde o controle dos atributos. Neste exemplo as diferenças de comportamento estão sendo visualmente sinalizadas no retorno do sistema.

No que se refere à movimentação, entre os aplicativos com manipulação voltada à representação observamos regularidade em alguns aspectos, por exemplo, quanto ao tipo de movimentação permitida e à forma de ativação da movimentação. A exceção fica por conta do AutoCAD no que se refere à movimentação de grupo. No MsDraw e no MGE todo o grupo é movido simultaneamente, no AutoCAD o sistema ignora a seleção de grupo e movimenta apenas a primitiva cujo traçado está sendo apontado no momento da ativação.

No caso do MTool a ativação da movimentação é diferente. Como o sistema não permite manipulação de forma das primitivas traçadas, a seleção não é sinalizada por pontos de manipulação mas por alteração de cor. A movimentação é feita com base em um segmento de reta traçado em qualquer ponto da tela. Todo o grupo selecionado é movimentado no sentido, na direção e na distância indicados pelo segmento de reta.

5.3.2. Operações de Apagamento e Movimentação em um Mesmo Aplicativo

Neste tópico vamos analisar a coerência entre operações de movimentação e apagamento no MTool. A partir da Figura 5.6 (a) (final do capítulo) foram feitas movimentações de duas faces. Inicialmente foi movimentada a face em vermelho (face interna). O sistema tratou a face como um objeto semântico e não como um objeto físico, duplicando as arestas que a definiam, de modo a não deteriorar as faces adjacentes (Figura 5.6 (b)). Considerando que este seja um procedimento válido, no que se refere à linguagem visual seria esperada a representação de um buraco na figura, mas isto não aconteceu. A mesma movimentação foi executada para uma face em amarelo (no limite da figura). Neste caso, apenas duas arestas foram duplicadas, e a retirada da face causou o efeito visual de um buraco (Figura 5.6 (c)). Neste exemplo, operações semelhantes geraram resultados visuais diferentes.

Supondo que o comportamento do sistema para apagamento fosse equivalente ao seu comportamento para movimentação, a partir da Figura 5.6 (a) foram executadas operações de apagamento de face interna e externa, semelhantes às de movimentação. Inicialmente tentou-se apagar a face em vermelho (face interna). O sistema bloqueou a operação como inválida. Do ponto de vista do usuário a operação de apagamento de face é dúbia; ele pode ter a intenção de criar um buraco na figura traçada, ou pode estar querendo desfazer a face, criando uma face maior. Em seguida foi apagada uma aresta da mesma face, o sistema aceitou, conforme Figura 5.6 (d). Visualmente não foi criada nenhuma inconsistência no objeto, uma vez que duas faces transformaram-se em uma. Em seguida foram apagadas mais três arestas, conforme Figura 5.6 (e). Através dos apagamentos de arestas chegou-se à segunda possibilidade apontada como solução para o apagamento de face. A face interna foi desfeita, gerando uma face maior. Caso a intenção do usuário fosse esta, ela poderia ser atingida por este caminho e não pelo anterior, embora do ponto de vista de consistência visual do usuário, a solução anterior fosse mais imediata. Com relação à face externa (face em amarelo), o sistema aceitou o apagamento da face produzindo a imagem na Figura 5.6 (f).

Em suma, neste exemplo observamos que: (a) do ponto de vista de expressão do usuário, a operação de apagamento de faces é dúbia, sua solicitação sugere dois procedimentos diferentes; (b) do ponto de vista visual, nada justifica o comportamento do sistema que não permite apagar a face, mas permite apagar cada uma das arestas que a compõem; e (c) o resultado visual do apagamento deixa claro que as arestas são compartilhadas entre as faces adjacentes, porém este resultado é incoerente com a operação de movimentação, que, ao movimentar uma face mantendo a face adjacente, sugere que existem arestas sobrepostas. Nestes exemplos a LV é dúbia, não permitindo uma inferência clara do modelo funcional do sistema.

5.3.3. Caso Especial de Movimentação

No MTool não é implementada a operação de cópia, ela pode ser executada através de um uso alternativo da operação de movimentação: movimentação deixando o original. Estamos chamando de uso alternativo porque a operação é utilizada para obter um efeito diferente da mera movimentação.

A movimentação "deixando o original" (Leave Original) é um caso particular da operação de movimentação, no qual o usuário movimenta uma cópia do objeto (primitiva ou grupo de primitivas) deixando o original no lugar. Esta manipulação na verdade corresponde a duas operações, uma cópia e uma movimentação. Para ativá-la o

usuário estaria fazendo uma tradução de sua intenção de cópia em uma operação de movimentação deixando o original. A vantagem que esta operação oferece com relação a alguns sistemas é que ao indicar a movimentação o usuário está indicando o local no qual será feita a cópia. A vantagem existe se compararmos esta operação com a operação de cópia oferecida por exemplo pelo MsDraw. Neste sistema a cópia é feita através da área de clipping (área de depósito de conteúdo cortado ou copiado), e a cópia é deixada em lugar determinado pelo sistema. Isto exigiria um segunda manipulação do usuário para posicionar a cópia no local desejado. Através da movimentação deixando o original este posicionamento é feito automaticamente. No entanto, existe um ônus cognitivo na tradução semântica da intenção do usuário para o procedimento do sistema. A especificação através do modelo GOMS apresentada no Exemplo 5.5 esclarece esta questão. Caso o projetista deseje de fato este procedimento, seria interessante que ele fosse adequadamente sinalizado no projeto da linguagem.
No sistema MTool a opção de menu para ligar e desligar o estado de "deixar o original" está localizado no grupo de operações de edição. Para o usuário novato, que não conhece o conceito, caso ele tenha a intenção de fazer uma cópia, a tradução semântica de sua intenção para este conceito é difícil. O estado de "deixar o original" pode ser ligado para todos os tipos de operação de movimentação, como por exemplo rotação e espelhamento.

Goal: Cópia de objeto
    Goal: Ativar estado
        Ligar estado de deixar original
    Goal: Selecionar objeto
        Selecionar objeto a ser copiado
    Goal: Movimentar
        Movimentar objeto para nova posição

Exemplo 5.5. Especificação do modelo GOMS para operação de cópia no MTool

Em termos de LV esta operação é dúbia, no sentido de que a expressão "deixar o original" pode sugerir dois comportamentos diferentes. Um deles é o de cópia e movimentação e o outro é o de desdobramento do objeto. Há situações em que o usuário quer desdobrar uma peça, transformando-a em um objeto com profundidade. Por exemplo, o usuário pode movimentar um polígono correspondente à forma em corte de um túnel, mantendo o original, de modo a criar um "tubo" correspondente ao próprio túnel, conforme Figura 5.7. Esta operação é coerente quanto ao aspecto visual, mas difícil de ser implementada no que se refere à estrutura de dados. Ela exige um tratamento extremamente complexo para manter a consistência da estrutura de dados. Normalmente os sistemas implementam a opção de cópia e movimentação. Este é um exemplo onde a manipulação intuitiva é bloqueada em função das restrições internas do sistema.

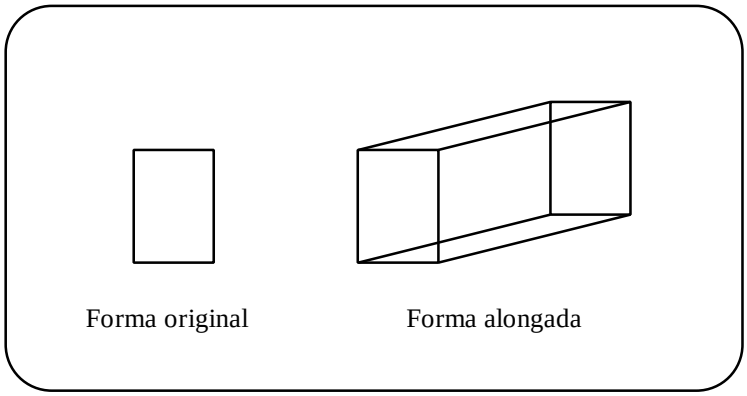


Figura 5.7. Operação de movimentação deixando o original

5.3.4. Comportamento nas Operações de Alteração de Forma

No que se refere à alteração de forma, de maneira semelhante à movimentação, em alguns aspectos observamos regularidade entre os aplicativos com manipulação voltada à representação (MsDraw, AutoCAD e MGE). Por exemplo, observamos regularidade no que se refere ao tipo de alteração permitida e à forma de ativação da mesma. Quanto ao tipo de alteração, sobre as primitivas do sistema os três aplicativos permitem apenas alterações de forma em escala. Nenhum destes aplicativos permite, por exemplo, que o vértice de um retângulo traçado seja manipulado de forma a transformá-lo em um trapézio. Eles apenas permitem que o retângulo seja manipulado em escala, em ambos os eixos do sistema de coordenadas, sem alteração da forma da primitiva. É conveniente ressaltar que, neste exemplo, o retângulo é uma primitiva do sistema, não se tratando de uma figura com a aparência visual de um retângulo, porém construída através de linhas. Para uma figura construída através de linhas, a alteração seria

feita alterando-se a forma de cada uma das linhas, também em escala. Obviamente neste caso, através da alteração de forma das linhas componentes, poder-se-ia obter uma alteração de forma na figura final.
A manipulação em escala geralmente é ativada por pressão-e-arrasto, a partir dos pontos de manipulação assinalados sobre o objeto selecionado. Mais uma vez a exceção fica por conta do AutoCAD no que se refere à alteração de forma em grupo. No MsDraw e no MGE se houver um grupo selecionado, todo ele terá sua forma alterada simultaneamente, o comportamento é coerente com a sinalização visual. No AutoCAD mesmo havendo um grupo assinalado, somente a primitiva cujos pontos estiverem sendo movimentados terá sua forma alterada. Neste caso o comportamento é incoerente com a sinalização visual, especialmente considerando que outras operações no AutoCAD, como por exemplo a operação de apagamento, são executadas para todo o grupo.

5.3.5. Caso Especial de Alteração de Forma

No que se refere à alteração de forma especificamente, o MTool, que é um aplicativo com manipulação orientada à construção, tem um comportamento semelhante àqueles com manipulação orientada à representação. A diferença é que o modelo semântico através do qual as primitivas do sistema são interpretadas permite que o usuário execute a alteração de forma através de um subterfúgio, no caso uma operação de movimentação de vértice.
Pelo fato do MTool ser um sistema cujas primitivas de domínio têm base em um modelo matemático (vértices, arestas e faces), observamos o uso da movimentação de primitivas hierarquicamente inferiores, no sentido de gerar alterações de forma nas primitivas hierarquicamente superiores, conforme Figura 5.8. Pelo modelo matemático, hierarquicamente o vértice é considerado nível 0, a aresta nível 1 e a face nível 2, uma vez que a aresta é delimitada por dois vértices e a face é delimitada por aresta. Portanto, a movimentação do vértice pode causar alterações na forma da aresta e da face. O MTool não oferece operação de alteração de forma, porém ela pode ser alcançada através deste subterfúgio.
Supondo que este efeito seja válido no modelo semântico, a LV estaria exigindo do usuário uma tradução semântica extremamente complexa, na qual a intenção de alteração de forma seria traduzida em uma operação de movimentação de vértice, conforme especificação do modelo GOMS, no Exemplo 5.6. Mais uma vez, seria interessante que o problema fosse adequadamente sinalizado ao projetista no processo de especificação da linguagem de interação.

Goal: Alterar forma de aresta
    Goal: Selecionar vértice
        Selecionar vértice que suporta aresta
    Goal: Movimentar vértice
        Movimentar vértice para nova posição

Exemplo 5.6. Especificação do modelo GOMS para operação de alteração de forma no MTool

Curiosamente o mesmo subterfúgio não se aplica a arestas. Não é possível alterar a forma de uma face movimentando-se uma aresta. Se esta operação for executada o sistema carrega a aresta desvinculando-a das arestas adjacentes.

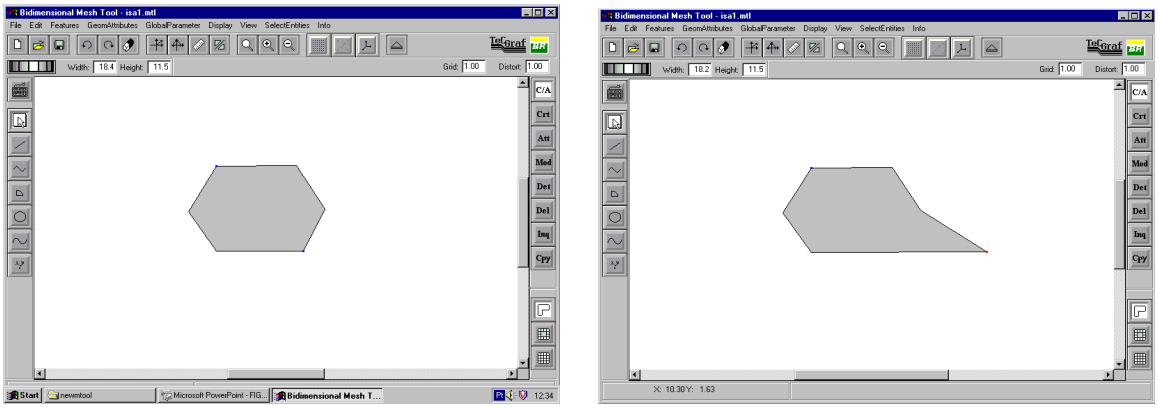


Figura 5.8 Alteração de forma na face, em função da movimentação de vértice

5.4. Necessidade de uma Nova Abordagem na Especificação da Linguagem Visual

Como pode ser constatado através destes exemplos, a expressão consistente do modelo funcional do sistema é uma tarefa difícil, especialmente nos sistemas com manipulação orientada à construção. A LV é limitada na quantidade de recursos que dispõe e na articulação destes recursos. Os recursos de expressão são limitados e são concorrentes. Isto limita a utilização do recurso visual, uma vez que o uso de um determinado recurso pode invalidar ou pelo menos descaracterizar o uso de outro.

A limitação dos recursos visuais nem sempre permite a representação de todos os atributos e estados simultaneamente. O projetista deve então tomar decisões a respeito de quais atributos caracterizam melhor o comportamento da entidade e escolher a melhor forma de expressá-los visualmente, lançando mão de um código complementar para o restante. Portanto, uma ferramenta de auxílio à especificação da LV deveria apoiar o projetista no processo de escolha do recurso visual mais adequado a cada situação, na definição do uso dos recursos e no controle da sobreposição do mesmo.

Outro aspecto relativo à sobreposição de recursos visuais expressivos refere-se à linguagem em uso. A linguagem em uso vai exigir sinalizações do sistema (como, por exemplo, informações de retorno) e sinalizações de manipulação, que podem assim causar sobreposição dos recursos expressivos. Por exemplo, a utilização de cor para expressar a seleção de um objeto pode colidir com a utilização de cor para expressar um atributo sobre o mesmo objeto. A simples definição do léxico da LV não assinala esta colisão. Apenas a análise da linguagem em uso pode determiná-la. Portanto, uma ferramenta de auxílio à especificação da LV também deveria prever a verificação de consistência a nível de discurso.

A ferramenta adotada para especificação da linguagem de interface deve auxiliar ao máximo o projetista, orientando-o a registrar da melhor forma os detalhes de expressão do modelo funcional. Obviamente, este requisito da ferramenta não pode significar, em si, um ônus para o projetista. Isto deverá ocorrer sem um custo adicional de trabalho, ou com um custo cujos resultados sejam bastante compensatórios. Neste sentido, apresenta-se no próximo capítulo um conjunto de recursos que podem integrar uma tal ferramenta, oferecendo a projetistas de LVs um instrumentos de projeto e análise, para apoio à produção de signos visuais de interfaces por MD.

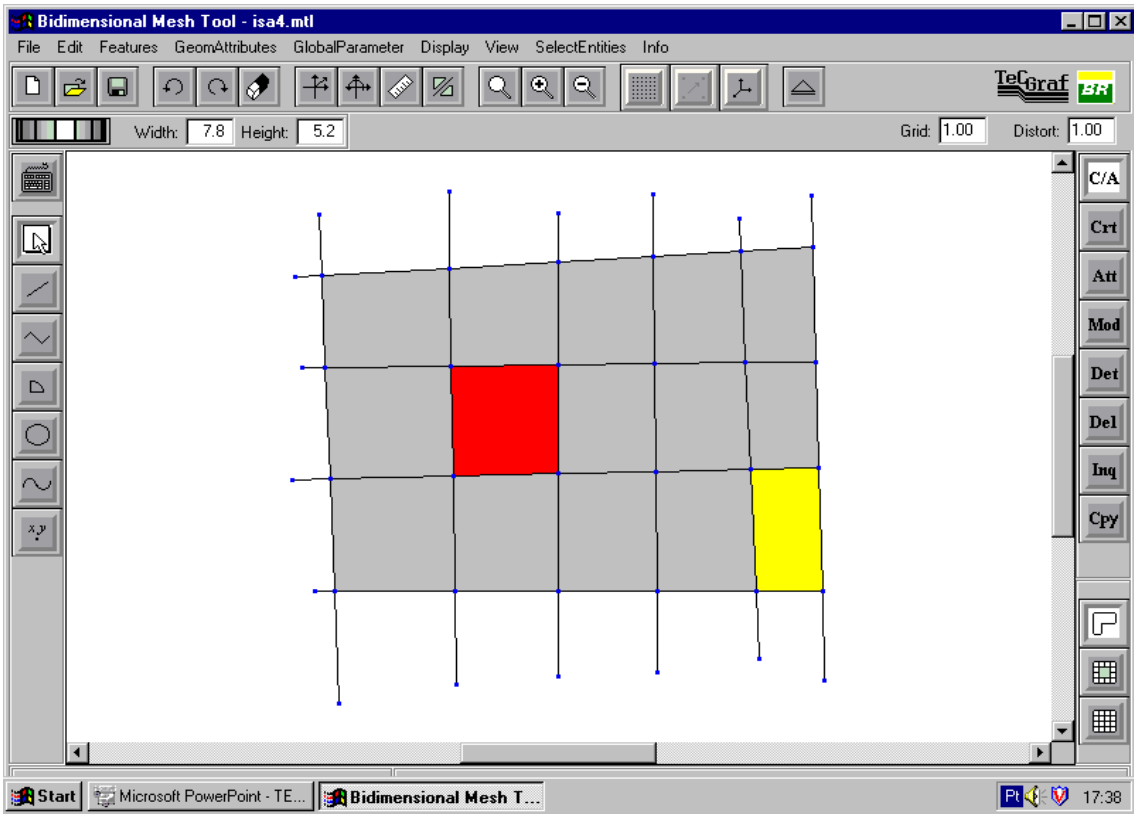


Figura 5.6. (a) Representação gráfica no MTool

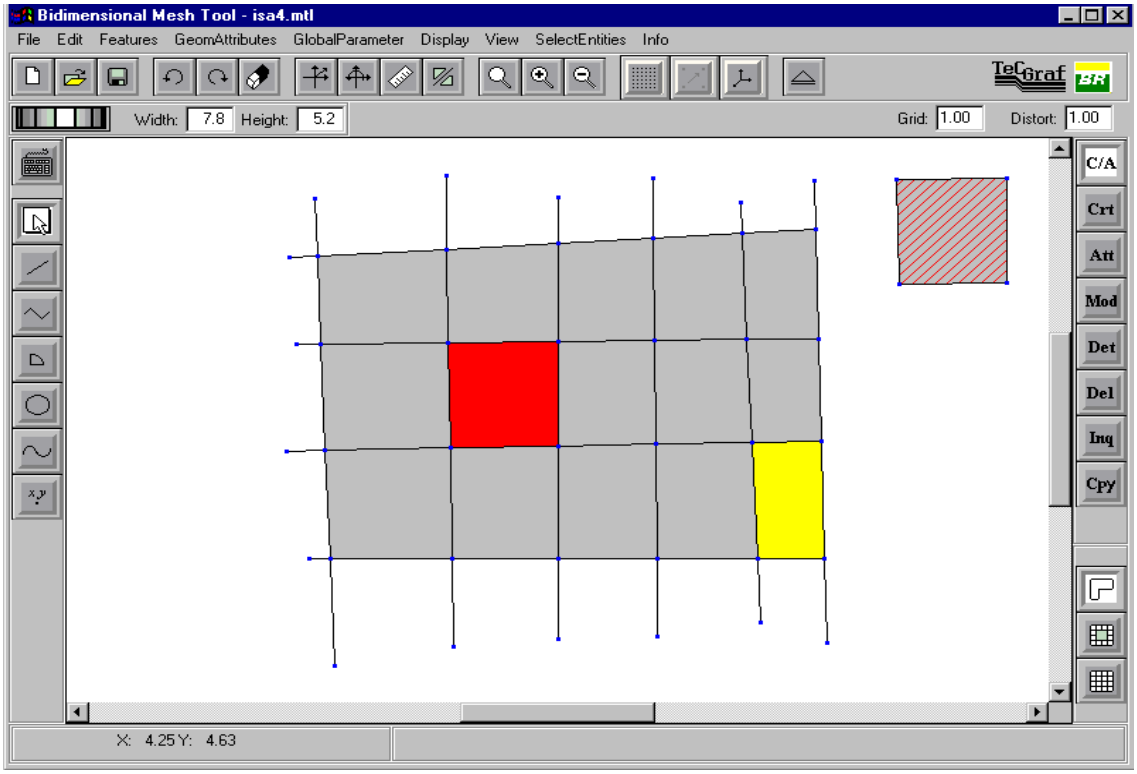


Figura 5.6. (b) Movimentação de face interna no MTool

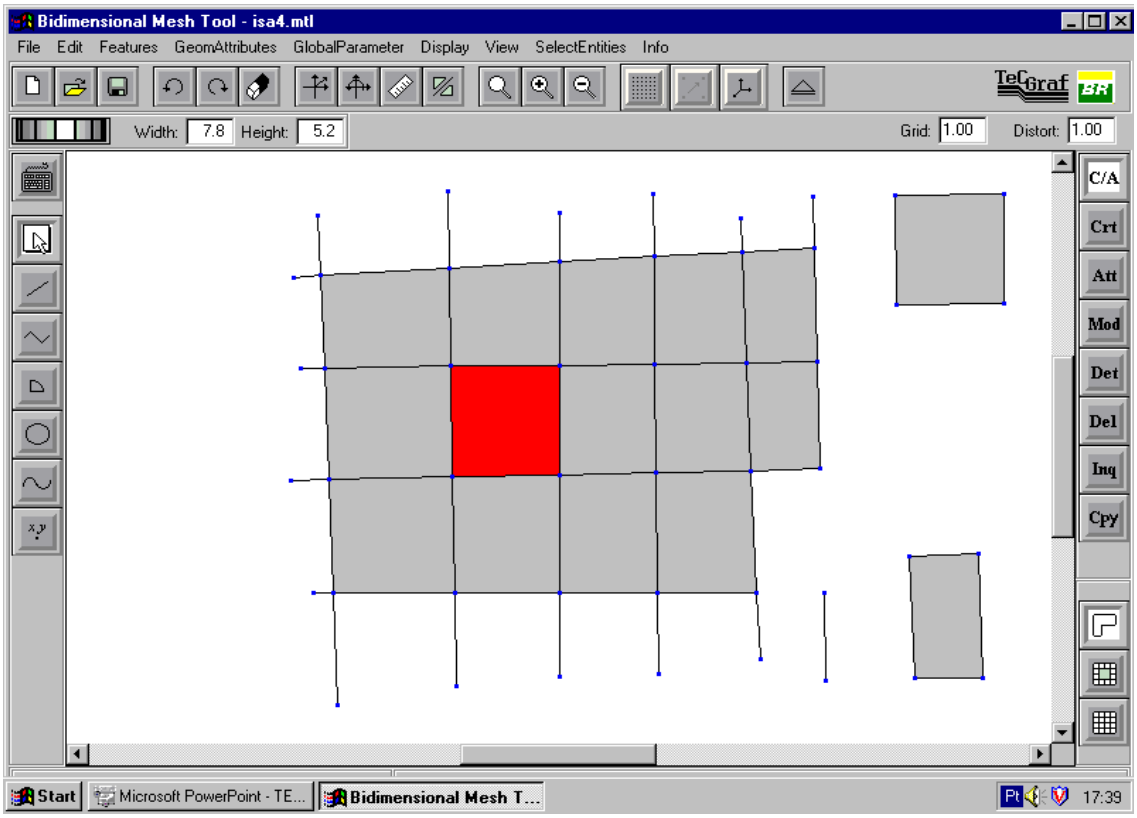


Figura 5.6. (c) Movimentação de face externa no MTool

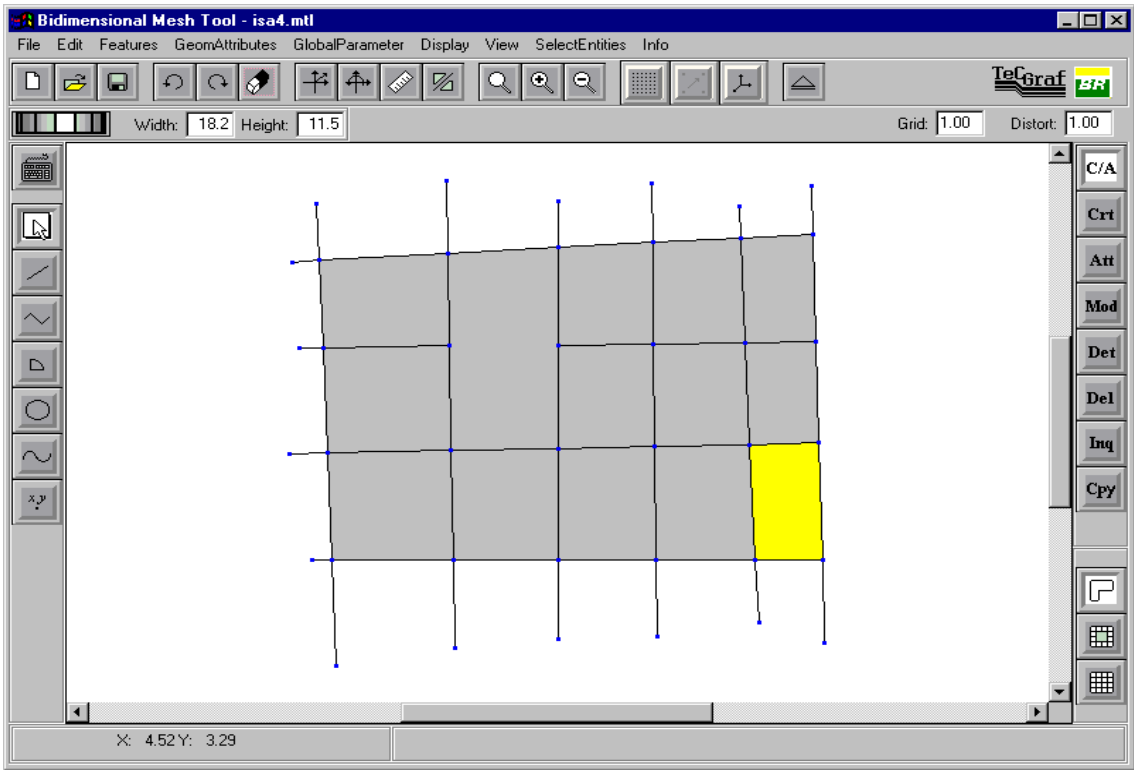


Figura 5.6. (d) Apagamento de uma aresta interna no MTool

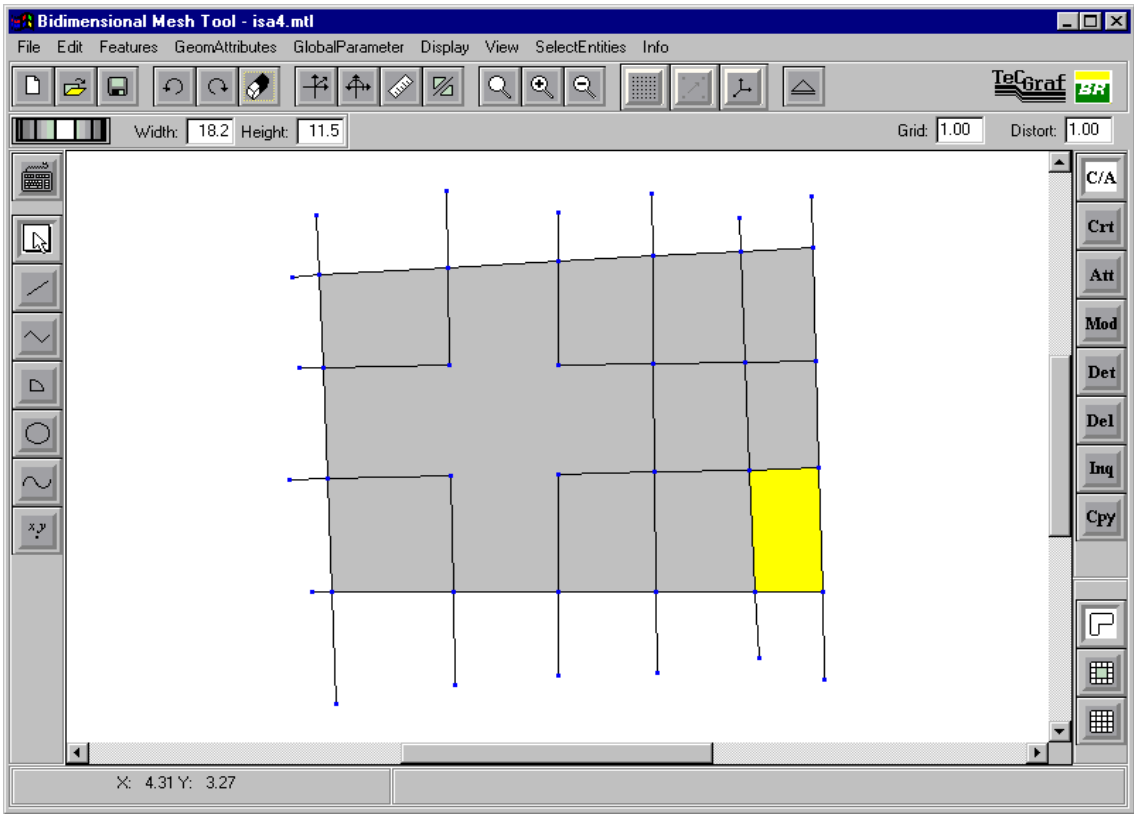


Figura 5.6. (e) Apagamento de três arestas adjacentes no MTool

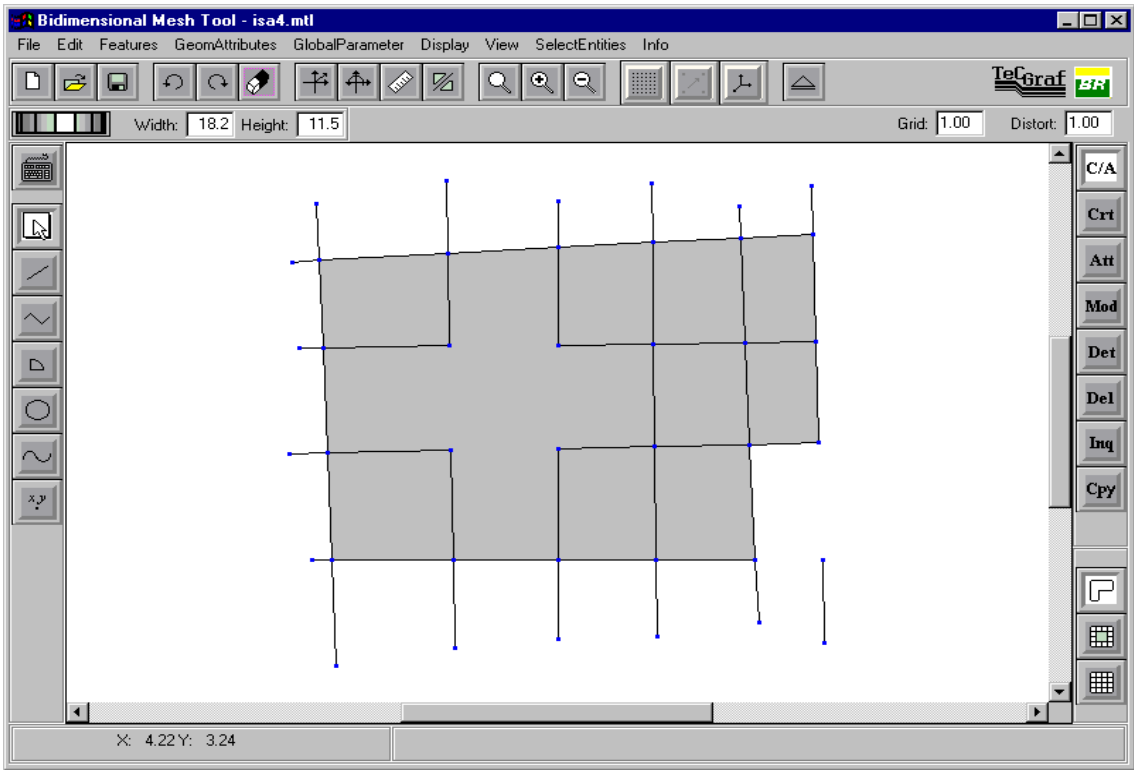


Figura 5.6. (f) Apagamento de face externa no MTool

A análise teórica apresentada nos capítulos anteriores permitiu tratar adequadamente o contexto de uso da LV (conteúdo e recursos expressivos), no ambiente de SMG. A partir dessa análise foram estabelecidos os requisitos básicos para uma notação de auxílio à especificação de LVs. Neste capítulo é apresentado o formato básico da STAG, analisando-se sua atuação com relação aos requisitos estabelecidos. A análise demonstra para quais requisitos a notação atua adequadamente e para quais ela atua com restrições. A STAG é orientada para o projeto e análise de LVs.

6.1. Aspectos Gerais

> *"We aim to capture the notion of regularity or consistency. Consistency is difficult to define and therefore difficult to measure, but it is informally recognized to be a major determinant of learnability. The advantages of consistency lie in facilitating generalizations by the user, who having learned some parts of the system can then infer others."* [Payne and Green 86, p. 95].

A Semiotic TAG (STAG) é uma notação para auxílio à especificação de LVs por MD, que orienta a sistematização de escolhas expressivas. Ela foi proposta como uma extensão da TAG (Task-Action Grammars). Parte dos problemas da TAG decorre especificamente de sua aplicação à LV, outra parte é intrínseca à própria estrutura da gramática. A proposta é que a notação complemente a TAG, superando alguns pontos de deficiência, para aplicação às LVs.
Com base na análise teórica e no estudo de caso apresentados nos capítulos anteriores, a seguir são definidos: (a) os requisitos básicos para uma notação de auxílio à especificação de LVs, (b) o formato básico da STAG, (c) o desempenho da STAG com relação aos requisitos básicos especificados e (d) as restrições da STAG com relação aos mesmos requisitos. No Capítulo 7 será discutida a implementação da STAG através de um ambiente para criação de signos visuais, de forma a orientar o projetista, automaticamente, na especificação da LV.

6.2. Enumeração de Requisitos Necessários a uma Gramática para Tratamento de LVs

Nesta seção vamos enumerar os requisitos necessários a uma gramática de apoio à especificação de LVs. A enumeração de requisitos será feita em comparação com a TAG e a D-TAG. Os requisitos para a STAG foram estabelecidos considerando os seguintes aspectos: (i) a notação deve estar apoiada em uma base teórica sólida, neste caso instanciada em um enfoque semiótico, e (ii) deve atender às necessidades específicas das LVs. Na proposta da STAG entendemos que não é o caso que o projetista da LV seja um semioticista. Por isto, a notação incorpora implicitamente alguns princípios básicos da semiótica, de forma que estes princípios sejam automaticamente respeitados, pela simples utilização da mesma. A STAG deve auxiliar o projetista a definir a linguagem de interface dentro da proposta semiótica, observando orientações específicas para LVs, que favoreçam um bom resultado.
A STAG deverá atender as necessidades de projeto e análise. Quando utilizada para projeto, ela estará modelando a concepção do projetista sobre o problema. Quando utilizada para análise, ela estará registrando o processo sígnico do usuário, ou seja, seus interpretantes, nem sempre previstos pelo projetista.
A STAG é uma notação de auxílio, ela não se propõe a determinar o que deve ser usado. Mesmo sendo implementada através de um aplicativo, ela não prescreve soluções para o projetista. Ela apenas sugere soluções mais adequadas, de acordo com suas heurísticas, baseadas nas DOCLV. A decisão final a respeito da LV será sempre do projetista.
Porque a STAG é uma extensão da TAG, ela incorpora os requisitos da TAG, como por exemplo refletir a estrutura de raciocínio do usuário, em função da tarefa. Ela incorpora também os requisitos da D-TAG, que é uma extensão da TAG, com o objetivo de registrar aspectos estáticos da expressão do sistema. Portanto, estes requisitos já estão considerados, sem que haja necessidade de reespecificá-los. Os requisitos abaixo foram definidos para a STAG considerando a natureza da TAG. Os requisitos de 6.2.a a 6.2.c são fundamentais no sentido que evocam uma reestruturação de alguns aspectos da TAG. Os requisitos seguintes atendem a questões mais específicas da LV ou a casos particulares, característicos de determinados ambientes.

6.2.a. Auxílio ao processo de lexicalização

Um dos pontos mais importantes no processo de lexicalização da LV é a segmentação dos contínuos de conteúdo. Em termos da TAG este processo é tratado como uma categorização dos elementos semânticos das tarefas. A TAG, segundo os autores, é construída sobre unidades de tarefas relativas aos componentes semânticos que caracterizam o domínio do sistema, "*... a featural description of that simple task in terms of semantic components which categorize the entire task world*" [Payne & Green 86, p. 111]. No entanto, não é definido como obter estes componentes semânticos, nenhuma regra ou orientação são fornecidas que auxiliem a captura dos componentes básicos e da estrutura semântica do sistema.
Em Payne & Green (86, p. 101) é apresentado um exemplo relativo à tarefa de mover objeto. Naquele exemplo, a tarefa "mover o cursor um caracter à frente" é caracterizada por duas variáveis, a variável Direção, valorada em "à frente", e a variável Unidade, valorada em "caracter". As instâncias de valoração são mapeadas em um determinado elemento expressivo; no caso, "a frente" é mapeado na tecla "CTRL" e "caracter" na letra "c". Portanto, a tarefa "mover o cursor um caracter à frente" é expressa por "CTRL C". O objetivo da TAG é que todas as operações de movimento de cursor sejam caracterizadas pelas mesmas variáveis, Direção e Unidade, e que as intâncias destas variáveis sejam mapeadas apenas uma vez em elementos expressivos. Com isto todas as operações semelhantes seriam mapeadas em uma mesma forma de ativação.

No exemplo, as variáveis caracterizadas como componentes semânticos da operação, unidade e direção, são elementos facilmente caracterizáveis por serem instâncias físicas de movimento. Mas, nem sempre esta facilidade existe. Normalmente não existem referências claras que auxiliem a segmentação adequada do contínuo de conteúdo. Na TAG não é apresentado um argumento que justifique esta escolha de forma sistemática, porque isto transcende os objetivos dos seus autores.

Sobre a Definição do Conteúdo de Representação

Não havendo uma orientação clara para uma segmentação sistemática do contínuo semiótico, o processo de lexicalização fica sujeito a soluções circunstanciais. Através da operação de seleção de objetos gráficos, vamos exemplificar o tipo de problema relacionado a uma segmentação circunstancial. Conceitualmente, algumas operações de edição gráfica, como por exemplo uma movimentação de objeto, operam sobre conjuntos de objetos selecionados, os quais podem ser unitários.
Cada tipo de seleção, unitária ou de grupo, pode ser expressa de diferentes formas. A seleção de uma única entidade é geralmente feita por apontamento direto. A seleção de um conjunto de entidades pode ser feita através de uma janela (bounding-box) ou através da definição de uma primitiva semântica. No segundo caso o sistema pode selecionar, por exemplo, "todas as arestas". A seleção também pode ser feita através dos atributos semânticos associados às primitivas. Por exemplo, o sistema pode selecionar "todas as áreas em concreto". Ou ainda, pode ser feita por caracterização de um objeto semântico, como por exemplo "selecionar todas as primitivas que compõem a pilastra da ponte". A relação de pertinência do conjunto pode ser definida de forma extensional, por enumeração, ou de forma intensional, por regra.
No Exemplo 6.1 apresentamos uma operação de mover objeto gráfico, incluindo a ação de seleção, representada através da TAG. Na especificação da operação "mover objeto" pela TAG, poderia ser caracterizada a variável Caso, conforme R1, a qual seria diretamente mapeada em uma ação do usuário.

Regras Gramaticais

R1. Mover objeto (Efeito = mover, Caso) --> ação-seleção [Caso] + destino = valor-objetivo

R2. ação-seleção [Caso = um-objeto-físico] --> Tipo = apontamento + Lugar1 = valor-objetivo

R3. ação-seleção [Caso = vários-objetos-físicos] --> Tipo = janela + Lugar1 = valor-objetivo +
Lugar2 = valor-objetivo

R4. ação-seleção [Caso = objetos-semânticos] --> Tipo = apontamento +
lugar = menu-atributos + "pressionar-botão-mouse"

Exemplo 6.1 (a) Especificação de uma operação de movimentação, através da TAG

Embora correta em termos da TAG, esta classificação não reflete o modelo semântico do sistema. Neste exemplo a forma de expressão escolhida está associada ao tipo de objeto selecionado, quando deveria estar associada ao critério da seleção, uma vez que o recurso de expressão é escolhido em função do critério de seleção. O conteúdo associado à variável Caso, deveria ser o critério de definição do conjunto (por enumeração ou por regra) sobre o qual a movimentação é executada. Por exemplo na R3, a instanciação vários-objetos-físicos para a variável Caso, não esclarece o tipo de seleção que será feita. Diferentes objetos físicos podem ser selecionados por circunscrição de um espaço, se eles estiverem fisicamente agrupados, ou por indicações individuais, se eles estiverem fisicamente dispersos ou misturados com outros. A instanciação objetos-semânticos também não esclarece o critério de seleção. Os objetos semânticos podem ser agrupados por indicação um a um, ou podem ser agrupados por especificação textual de um atributo semântico, entre outras. Na interação com o sistema, o usuário expressa critérios de agrupamento. Ele não diz "vou agrupar primitivas semânticas"; diz "vou agrupar primitivas semânticas por definição de um atributo". O Exemplo 6.1 (b) apresenta uma especificação alternativa.

Regras Gramaticais

R1. Mover objeto (Efeito = mover, Caso) --> ação-seleção [Caso] + destino = valor-objetivo

R2. ação-seleção [Caso = um-objeto-físico] --> Tipo = apontamento + Lugar1 = valor-objetivo

R3. ação-seleção [Caso = objetos-físicos-agrupados] --> Tipo = janela +
Lugar1 = valor-objetivo + Lugar2 = valor-objetivo

R4. ação-seleção [Caso = objetos-físicos-dispersos] --> "pressionar-shift" +

Tipo = apontamento + Lugar1 = valor-objetivo

R5. ação-seleção [Caso = objetos-atributos-semânticos] --> Tipo = apontamento + lugar = menu-atributos + "pressionar-botão-mouse"

Exemplo 6.1 (b) Especificação alternativa de uma operação de movimentação, através da TAG

A gramática deveria refletir a seguinte estrutura semântica: (i) operação de movimentação também opera sobre conjuntos, (ii) a operação de seleção usa uma variável que indica o tipo de especificação do conjunto, (iii) o conteúdo desta variável deverá ser associado ao recurso expressivo. O importante é detectar corretamente a natureza do elemento determinante da forma de expressão, e neste aspecto o critério de segmentação do contínuo do conteúdo tem um papel fundamental.

Em Payne (91), a respeito da D-TAG, o autor propõe que as unidades de tarefas sejam especificadas em função de suas características, considerando os dispositivos. Para as tarefas de criar objeto, alterar forma, rotacionar e espelhar objeto ele exemplifica com as seguintes características: tipo-objeto, efeito, tipo-alteração e alteração, reproduzidas na Tabela 6.1. O exemplo apresentado em Payne (91, p.146) tem por objetivo comparar tratamento de texto com objetos geométricos. Este não é o nosso caso. Portanto, não foram reproduzidas as operações referentes a texto.

Simple Tasks			Features	
	Obj-type	Effect	Change-type	Change
Create geometric object	geometric	create		
Change object shape	geometric	modify	continuous	shape
Rotate object	any	modify	discrete	rotate
Flip object	any	modify	discrete	flip

2b. Task [Effect=modify, Obj-type=geometric, Change-type=discrete] ->
select object
modify [Change-type, Change]

3. modify [Change-type = discrete] ->
action (point, display-item ([Change],
[type=menu-bar])),
action (drag, display-item ([Change],
[type=menu-bar;
type=pull-down-menu;
location=below;
type=item]))

Tabela 6.1. Reprodução parcial de "*A partial D-TAG for MacDraw*" [Payne 91, p.146]

Através dessa classificação Payne busca os componentes semânticos que caracterizam as operações, e que permitem criar regras de mais alto nível na gramática. No caso, as alterações do tipo discreto ele irá associar a uma expressão do tipo apontamento (point) para a barra de menu principal e arrasto (drag) para um submenu. Este menu conteria os vários tipos de alterações discretas que pudessem ser executadas sobre objetos geométricos (no caso rotação, espelhamento e preenchimento). Portanto, as alterações de tipo discreto seriam mapeadas em uma expressão do tipo point e drag sobre o menu principal. Essa classificação permite criar regras de alto nível, onde todas as operações que obedecem determinada característica são mapeadas em uma mesma ação. O autor está trazendo à tona componentes de classificação, que sejam de alguma forma ligados à natureza do tratamento gráfico, e, em função disso, sejam determinantes da forma de expressão. Ou seja, o sistema gráfico, nos dispositivos onde é operado, permite operações de modificação do tipo discretas e contínuas, e cada uma delas será mapeada em uma forma de expressão diferente. A abordagem é correta porém falha no referencial utilizado para classificação.

Tarefas Simples	Conteúdo expresso			
	Efeito	Tipo objeto	Instância objetos	Qualificador
Criar objeto posicionamento	criar	primitiva		
Mudar forma objeto atuação	modificar	qualquer	critério-escolha	critério-
Mover objeto atuação	modificar	qualquer	critério-escolha	critério-

Rodar objeto	modificar	qualquer	critério-escolha	critério-atuação
Espelhar objeto	modificar	qualquer	critério-escolha	critério-atuação

R1. Tarefa [Efeito=criar, Objeto] -->
selecionar-ferramenta [Objeto] +
gerar-primitiva [Objeto]
R2. Tarefa [Efeito=modificar, Critério-escolha, Critério-atuação] -->
selecionar-objeto [Critério-escolha] +
executar-modificação [Critério-atuação]

R3. selecionar-objeto [Critério-escolhac= único] -->
Primitivas ( ) +
Intenção (indicar) +
Estado-seleção (ligado)
R4. selecionar-objeto [Critério-escolha = grupo] -->
Primitivas ( ) +
Intenção (indicar-grupo) +
Estado-seleção (ligado)
R5. selecionar-objeto [Critério-escolha = atributo] -->
Atributos ( ) +
Intenção (indicar) +
Estado-seleção (ligado)
R6. executar-modificação [Critério-atuação = física] -->
Plano-interação (plano) +
Intenção (indicar)
R7. executar-modificação [Critério-atuação = discursiva] -->
Opções-textuais ( ) +
Intenção (indicar)

Tabela 6.2. Especificação alternativa para operações de edição básica

Uma abordagem semiótica da questão, nos leva a resultado semelhante através de outro caminho. Embora o tipo de alteração (discreta ou contínua) seja uma característica de fato dessas operações, não é esse conteúdo que o usuário expressa. Na Tabela 6.2 as mesmas operações foram caracterizadas com base no conteúdo que o usuário deve expressar na interação. Para todas as operações que atuam sobre instâncias de primitivas já criadas, o usuário deverá expressar um critério de escolha, para seleção das primitivas objeto da ação, e um critério de atuação, para definir o tipo de atuação que ele fará sobre a primitiva.

Quanto ao critério de escolha, o usuário poderá selecionar as primitivas a partir do que chamamos critérios sintático ou semântico. No critérios sintáticos ele atua diretamento sobre as primitivas, indicando, de forma individual ou em grupo, aquelas que deseja selecionar. No critério semântico ele indica, através de um atributo semãntico, quais primitivas ele deseja selecionar. Cada tipo de critério será mapeado em uma forma de expressão específica. Por exemplo, na regra R3 o conteúdo expresso na seleção individual é uma indicação sobre as primitivas traçadas, essa indicação será mapeada em um clique. Na regra R5 o conteúdo expresso na seleção por atributo é uma indicação de um atributo. Os atributos serão expressos através de menus, e a indicação, mais uma vez é mapeada em um clique.

Da mesma forma ocorre com relação ao critério de atuação. Operações de rotação, por exemplo, podem ser implementadas como movimentos contínuos ou discretos. Nenhum dos dois aspectos caracteriza necessariamente a rotação. Ao executar uma rotação, o usuário não expressa uma característica do movimento, mas uma forma de atuação. Se ele quiser atuar fisicamente sobre o objeto (por MD), ele irá manipular um elemento de apoio como, por exemplo, pequenas coordenadas no plano, conforme especificado na regra R6. Se ele quiser atuar através de recurso discursivo, ele irá escolher alguma opção através de um menu ou especificar um valor para rotação. Neste caso, o fator determinante da expressão do usuário será sua opção por uma atuação física ou a nível de discurso sobre o objeto.

Sobre as Categorias Ontológicas

O objetivo da linguagem de interação é expressar corretamente o comportamento do objeto computacional. Ela deve expressar as abstrações sobre a interação sobre o objeto computacional. Como foi visto no Capítulo 2, o usuário raciocina em termos de categorias. Portanto, se a linguagem retratar adequadamente as categorias ontológicas do objeto, ele será mais facilmente interpretado pelo usuário. Por exemplo, a representação visual de

um polígono sugere que as propriedades que caracterizam um polígono sejam válidas. Se o polígono visual é composto por linhas separadas, ele não está na mesma categoria de um polígono de fato. A categoria visualmente sugerida será incompatível com as categorias internas, de acordo com as quais a representação será tratada. Se, por exemplo, o usuário traçar um polígono através de linhas, pedindo em seguida para agrupar o conjunto de primitivas, e o sistema oferecer uma sinalização de que está tratando todo o conjunto como um objeto só, o usuário pode supor que houve uma alteração de categoria ontológica, de linhas para polígono. Em função disto, o usuário pode supor que o objeto passe a se comportar como um polígono, quando então ele poderá, por exemplo, ser preenchido. O agrupamento dos segmentos de reta que formam o polígono potencializa a expectativa de que, não só ele possa operar o conjunto como um elemento único, mas que passem a valer as propriedades características deste elemento único. Deve haver harmonia entre as categorias com as quais o sistema opera internamente, e as categorias que são expressas na interface, para que o processamento interno do sistema não cause uma ruptura na consistência da percepção visual do usuário.
A ontologia pode se estender a níveis relativos ao modelo semântico. Por exemplo, em uma estrutura de uma ponte, se o usuário vê um conjunto de elementos visuais como "o pé da ponte", ele deveria ter meios de assinalar esta visão e trabalhar com ela.

6.2.b. Tratamento distinto entre planos da expressão e do conteúdo, de usuário e sistema

Este problema está profundamente ligado ao problema anterior. Parte da dificuldade de caracterização dos elementos gramaticais tem como pano de fundo a não distinção dos planos da expressão e do conteúdo, conforme enfoque semiótico. No Exemplo 6.2 é apresentada uma reprodução parcial da especificação apresentada em Payne & Green (86, p. 108), na qual este problema pode ser constatado. Este exemplo foi escolhido por ser um caso bastante comum, presente na maioria dos editores de textos. Embora não sendo um exemplo da TAG sobre uma operação por MD, é bastante característico em termos da proposta da gramática. O exemplo foi parcialmente reproduzido, mantendo a numeração original das regras. Na regra 6.10 é descrita uma ação do tipo "apontar", que é uma ação conceitual do usuário. No mesmo nível gramatical é descrito o ícone disponível para escolha, que é uma forma de expressão do sistema, para um determinado conteúdo. E ainda, no mesmo nível, é apresentada a forma de expressão do usuário para a ação conceitual de escolha. A Tabela 6.3, executada nos moldes das tabelas de dados apresentadas no Anexo A, esclarece esta situação, distinguindo elementos de conteúdo e expressão, do usuário e sistema. Para a construção adequada da linguagem a gramática deve distinguir claramente estes elementos.

Tarefa-simples
Criar novo objeto (Efeito = criar, Caso = regular)
Regras gramaticais
6.1 Tarefa [Efeito = criar, Caso] -> selecionar-ferramenta +
apontar-para-2-lugares [Caso, Lugar1 = valor-do-objetivo,
Lugar2 = valor-do-objetivo]
6.5 apontar-para-2-lugares [Caso = regular, Lugar1, Lugar2] -->
ação [Tipo = apontar, Lugar1] + arrastar-para-lugar [Lugar2]
6.10 selecionar-ferramenta --> ação [Tipo = apontar, Lugar = ícone-ferramenta] +
"pressionar-botão-mouse
[Payne & Green 86, p. 108, reprodução parcial]

Exemplo 6.2. Especificação de uma operação de criação, através da TAG
No Exemplo 6.3 é apresentada uma nova especificação para a regra 6.10. Nela, o que está sendo apontado pelo nome simbólico Objeto é o conteúdo de sistema oferecido ao usuário, são as primitivas de interação conceituais, disponíveis para criação de um novo objeto. Não está sendo registrado na regra gramatical a forma de expressão destas primitivas. Da mesma forma o que está sendo apontado pelo nome Tipo é o conteúdo do usuário, cujo valor foi trocado de "apontar" para "indicar", no sentido de indicar uma escolha. Neste caso trata-se apenas de um ajuste do termo, uma vez que apontar pressupõe algum tipo de ação física, enquanto a idéia de indicação é mais conceitual. O que o usuário expressa é a indicação de uma ação de escolha entre as primitivas de interação disponíveis. A nova especificação da regra 6.10 registra a transformação da intenção do usuário em tipos conceituais, sem registrar as formas de expressão.

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	operação	menu textual		

2			indicar	clique

Tabela 6.3. Tabulação dos dados contidos na regra 6.10 do Exemplo 5.2.

Para um tratamento sistemático da linguagem a gramática deve distinguir elementos de tradução semântica (mapeamento entre a intenção do usuário e os procedimentos do sistema) de elementos de tradução articulatória (ativação do sistema). Os elementos envolvidos na tradução semântica são os tipos do conteúdo, e na tradução articulatória são os tipos da expressão. Portanto, tipos do conteúdo e tipos da expressão devem ser claramente diferenciados.

No exemplo original da gramática, a intenção do usuário de criar um objeto é traduzida para uma ação de seleção de uma ferramenta e uma ação de informação de dados. Por sua vez a ação de seleção é traduzida para uma ação de apontamento para um ícone. Na especificação alternativa, a tradução da intenção do usuário em ações de seleção e informação de dados foi mantida. Porém, a tradução da ação de seleção foi feita para uma ação de indicação de uma ferramenta (primitivas-de-interação), onde indicação e primitivas-de-interação são, respectivamente, tipos de conteúdo do usuário e do sistema. O objetivo foi tornar ambos os conteúdos independentes do recurso expressivo utilizado. As primitivas de interação do sistema podem, por exemplo, ser expressas por ícone, por menu textual ou por botões, sem que isto afete a gramática. Esta especificação da TAG fundamenta-se na proposta de que a alteração da forma de expressão do sistema não afeta a tradução semântica do usuário. A nova especificação da TAG preserva a especificação do conteúdo do usuário associado ao conteúdo do sistema.

6.10. selecionar-ferramenta --> ação [Objeto = primitiva-de-interação, Tipo = indicar]

Exemplo 6.3. Especificação alternativa para um componente da operação de criação

No exemplo para a D-TAG apresentado em Payne (91), parcialmente reproduzido na Tabela 6.1, observamos problema semelhante. Na regra 3, para modificações do tipo discretas, são especificadas duas ações, uma ação de apontamento (point) para a barra de menu e uma ação de carregamento (drag) para o submenu. Este tipo de especificação traz para a gramática o uso do recurso expressivo textual (barra de menu principal e submenu) misturado ao conteúdo.

6.2.c. Especificação do código a nível de tipo e instância

Por não caracterizar claramente os planos do conteúdo e da expressão, a TAG não permite dois níveis de associação entre eles, associação por tipo e associação por instância. Por exemplo, um tipo do conteúdo como Material pode ser associado a um tipo expressivo como cor. Isto significa que as diferentes instâncias de material serão expressas através de diferentes cores. Será feita também a associação por instância. Por exemplo, o concreto poderá ser expresso pela cor amarela, o ferro pela cor vermelha, e assim por diante. A distinção entre tipo e instância garante estruturalmente uma regularidade maior na linguagem. Quando a associação é feita apenas por instância, soluções ocasionais ficam facilitadas. Por exemplo, o concreto poderia ser expresso por uma textura pontilhada, enquanto o ferro fosse expresso pela cor vermelha.

Este problema não fica evidente quando a gramática é utilizada para linguagens textuais, porque neste caso a expressão do sistema é feita através de um único tipo de recurso, o texto, não existindo necessidade de escolha entre tipos expressivos em função de tipos do conteúdo.

O código correlaciona conteúdo e expressão. Se a notação orientar o projetista no sentido de que a correlação seja feita por tipo e tipo/instância, ela estará garantindo uma regularidade estrutural maior na linguagem. A associação entre tipos, do conteúdo e da expressão, deverá ser feita previamente à associação entre instâncias.

6.2.d. Suporte à verificação de regularidade

Nas LVs para SMG os seguintes níveis de atuação devem ser observados no que se refere à regularidade do código, o intra-código e o inter-código. Neste ambiente, sobre a mesma representação visual o usuário pode interagir através de códigos diferentes, de forma alternada. Por exemplo, num sistema pré-processador o usuário pode interagir alternadamente com base em um modelo de engenharia (sobre a geometria do objeto) ou com base em um modelo de elementos finitos (sobre a malha de elementos finitos). Para que a linguagem como um todo seja consistente, é necessário que os protocolos dos diferentes códigos sejam semelhantes. O mesmo nível de regularidade deve ser observado em todos os aspectos da linguagem. Portanto, a notação deve oferecer suporte à verificação de regularidades intra-código e inter-código.

No que se refere ao intra-código, sendo a LV especificada de acordo com os tópicos 6.2.a a 6.2.c, na busca de regularidade a notação deve tratar a linguagem desde o nível lexical até o nível de unidade tópica de discurso.

6.2.e. Sobreposição de recursos expressivos

Conforme foi visto no Capítulo 3, nas LVs verificamos a ocorrência de sobreposição de recursos expressivos sobre o recurso básico forma. Algumas vezes os recursos se sobrepõem causando interferência mútua danosa, como no caso de duas cores sendo utilizadas simultaneamente. Outras vezes não há interferência, como por exemplo, uma cor e uma textura sendo utilizadas simultaneamente. Outras vezes ainda, os recursos não são usados

simultaneamente, como por exemplo duas cores sendo usadas sobre a mesma forma, em diferentes unidades tópicas do discurso.
Portanto, é fundamental que a notação ofereça formas de verificação sobre as sobreposições, não apenas a nível de recurso expressivo associado a cada forma, mas no contexto em que esses recursos são utilizados.

6.2.f. Interação com outros códigos

A notação deve auxiliar o projetista a demarcar a fronteira de atuação do código visual e registrar a atuação de código complementar. Por exemplo, em uma operação de criação de primitiva gráfica o usuário deve informar o objeto a ser criado, antes da especificação dos dados, para que o sistema saiba interpretar os dados de entrada. Neste caso, algum tipo de menu será necessário para que o usuário escolha entre as primitivas de interação disponíveis no sistema. Este menu poderá ser textual, icônico, por botões ou outro. Qualquer que seja o código expressivo adotado será utilizado complementarmente ao código visual, na traçado da primitiva por MD.
A notação deve registrar também a redundância na utilização de código visual. A redundância de código ocorre quando um conteúdo semântico idêntico é expresso por dois códigos diferentes. Por exemplo, a mesma operação expressa por código visual e textual. Por exemplo, um objeto que estivesse representado em vermelho por estar selecionado, em conjunto com um texto indicando "objeto selecionado". A redundância de código pode ser propositadamente utilizada em várias situações, por exemplo, no sentido de chamar a atenção do usuário.
A notação também deve auxiliar a verificação de abuso de código. Estes são os casos em que o projetista lança mão do mesmo tipo de recurso visual para expressar tipos de conteúdos diferentes. Um caso muito frequente de abuso de código é no uso da cor. Porque a cor é um recurso de grande flexibilidade de uso, ela se presta a expressar conteúdos de diferentes naturezas, os projetistas inadvertidamente podem utilizá-la em excesso. Por exemplo, o projetista pode utilizar a cor para expressar propriedades e estados de um objeto, podendo ocorrer de o estado mascarar a propriedade, ou vice-versa.

6.2.g. Tratamento de relações de dependência entre tipos e instâncias, do conteúdo e da expressão

Algumas instâncias expressivas do sistema, quando referenciadas pelo usuário, têm uma instância expressiva de retorno padrão. Por exemplo, a instância expressiva "botão", quando referenciada pelo usuário (clicada pelo usuário), oferece como retorno a instância expressiva "botão sombreado", representando um botão pressionado. As instâncias expressivas "botão" e "botão sombreado" podem estar associadas a instâncias de conteúdos diferentes. Por exemplo, vamos supor um caso em que uma primitiva de interação do sistema, como linha (instância de conteúdo), esteja associada à instância expressiva "botão". Isto significa que a disponibilidade dessa primitiva de interação está sendo expressa pelo sistema através do recurso expressivo "botão". Quando esta primitiva é ativada pelo usuário, o sistema entra em estado-de-criação (de linhas). Neste momento, ele deve retornar ao usuário a informação de que o estado-de-criação está ativo. A informação de estado-de-criação ativo é associada à instância expressiva "botão-sombreado". Neste exemplo, duas instâncias de conteúdo diferentes estão associadas a instâncias interligadas de um mesmo recurso expressivo.
No sentido de manter a regularidade da linguagem é interessante que a notação registre essa interligação entre as diferentes instâncias de recurso expressivo. Isto garante o mesmo retorno, automaticamente, sempre que a instância expressiva for utilizada na linguagem. Com isso, poderão ser evitadas situações como, por exemplo, um menu que oferece diferentes sinalizações de retorno à ativação, dependendo do contexto em que é ativado (como brilho ou contraste). Deve-se observar que no caso do menu o retorno pode estar significando apenas "esta opção foi escolhida". Esta é uma comunicação a nível do modelo de interação do sistema, não é uma comunicação a nível do modelo funcional, como no exemplo anterior. Estas comunicações não exigirão do projetista uma nova associação entre conteúdo e recurso expressivo, como no caso da ativação do estado-de-criação.
Esta associação direta será tratada como encapsulamento do código expressivo. O encapsulamento permite a associação automática entre instâncias expressivas. A forma de associação deverá distinguir casos como os acima exemplificados. Em alguns casos o encapsulamento aumenta a granularidade da notação, na medida em que está sendo especificado automaticamente o código de ativação do sistema e seu retorno.

6.2.h. Tratamento de elementos de customização

É possível prever um certo nível de customização na LV, como por exemplo a codificação de novos atributos. No aplicativo MTool é encontrado um exemplo bastante significativo. O sistema oferece um conjunto de cores para associação a novos materiais, para serem utilizados na especificação das estruturas dos artefatos de engenharia. Se a LV estiver adequadamente estruturada, fazendo associações de códigos por tipo e instância, a notação poderá averiguar a consistência do código a nível de tipo. Uma vez que as instâncias de Material só serão especificadas pelo usuário em tempo de execução, as averiguações sobre sobreposição de recursos visuais, por exemplo, poderão ser feitas para todo o conjunto de cores oferecido como recurso expressivo para o tipo Material.

6.3. Apresentação da STAG

A STAG tem quatro componentes: (a) a declaração de tipos e instâncias, (b) o mapeamento entre tipos e instâncias de conteúdo e expressão, (c) um dicionário de tarefas simples, opcional e (d) um conjunto de regras que chamaremos de gramática, para distinguí-lo do mapeamento. Este conjunto de componentes será utilizado dentro

de uma estrutura hierárquica, que permite gerar uma linguagem com seções paralelas, entre as quais o usuário poderá alternar durante o uso da linguagem. A estrutura e cada um dos quatro componentes serão detalhados a seguir.

6.3.1. A Estrutura da Notação

A estrutura hierárquica da notação permite especificar seções paralelas em diferentes níveis de granularidade. Seções paralelas são segmentos da linguagem entre os quais o usuário alterna durante a interação com o aplicativo, ou seja, esses segmentos não são utilizados simultaneamente. O uso não simultâneo ocorre por diferentes motivos, de acordo com o nível de granularidade da seção. As seções paralelas de mais alto nível, às quais chamaremos de contextos paralelos de discurso (CPD), permitem a alternância entre interações com base em diferentes modelos semânticos. Por exemplo, nos pré-processadores o usuário interage com o aplicativo de forma alternada entre um modelo de engenharia e um modelo de elementos finitos (conforme apresentado no Capítulo 1). Elas permitem também a alternância entre diferentes formas de manipulação da representação visual, por exemplo uma manipulação voltada à representação ou à construção (Capítulo 1). As seções paralelas de nível mais baixo na hierarquia equivalem a unidades tópicas de discurso (UTD). A Figura 6.1 ilustra a estrutura hierárquica proposta para a STAG.

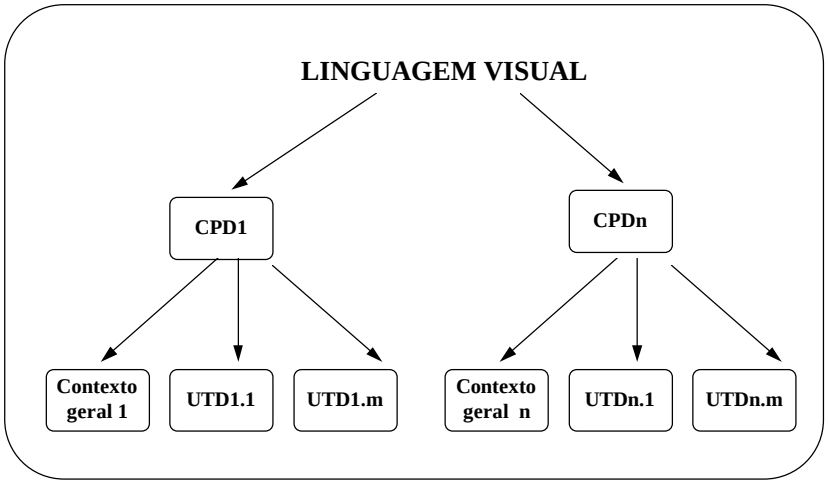


Figura 6.1. Estrutura Hierárquica da STAG

A UTD equivale a um pequeno núcleo da linguagem, dentro do CPD, onde um contexto dialógico particular pode ser instaurado. A UTD permite ao projetista selecionar os conteúdos expressos a cada momento, no interesse específico da tarefa. Para cada UTD o conteúdo de sistema expresso através da representação visual pode ser totalmente alterado, sendo substituído por um outro, mais significativo em função da tarefa. Deve haver um cuidado especial do projetista em sinalizar adequadamente ao usuário a entrada e a saída de cada UTD. Ao entrar em uma UTD um contexto interativo específico substitui o contexto corrente, ao sair, este contexto é automaticamente recuperado. Por exemplo, para alterar a forma de um objeto representado, em geral, é necessário selecionar este objeto. Em um aplicativo onde a seleção seja expressa através de cor, informações anteriores, como propriedades do objeto, também expressas através de cor, serão mascaradas. Neste caso, a seleção do objeto sinaliza a entrada em uma UTD, onde o contexto interativo corrente será mascarado durante as várias operações de edição que podem ser executadas, em sequência, sobre o objeto selecionado. Este contexto será automaticamente restaurado quando o objeto for desselecionado. Neste exemplo, durante a execução da tarefa de edição da forma, as propriedades associadas não são relevantes, podendo ser mascaradas, o que permite ao projetista fazer um uso adicional do recurso expressivo cor.

A UTD pode ser especificada em diferentes níveis de granularidade, podendo atender a uma meta ou a uma tarefa. A distinção que estamos fazendo entre meta e tarefa é apenas didática. Entendemos que a meta do usuário é uma tarefa em um grão maior, ou seja, ela pode ser desdobrada em várias tarefas. De acordo com essa classificação, a UTD especificada para uma meta tem um grão maior do que aquela especificada em função de uma única tarefa, mas ambas têm as mesmas características. A UTD é um mecanismo fundamental no sentido de otimizar a capacidade expressiva da linguagem, restrita por natureza, em função das limitações dos recursos visuais e da condição de sobreposição dos mesmos sobre o recurso básico forma (Capítulos 3 e 4). Um mesmo recurso, por exemplo cor, pode ser utilizado para expressar conteúdos diferentes dentro e fora das UTDs, desde que sejam respeitadas as orientações de consistência e o limite de utilização em função da capacidade cognitiva do usuário. As DOCLV foram apresentadas na Seção 4.3.4.

No CPD o léxico da linguagem é particular a cada contexto, uma vez que a representação visual está sendo interpretada de acordo com diferentes modelos semânticos, e o léxico da linguagem é construído a partir da segmentação do contínuo de conteúdo referente ao modelo semântico. Portanto, cada CPD terá suas próprias declarações de tipos e instâncias, seu próprio mapeamento e sua própria gramática. Quando o usuário alterna entre

diferentes CPDs ele alterna entre diferentes LVs. A alternância ocorre em uma mesma seção interativa com o aplicativo e com uma frequência significativa, dependendo da tarefa em desenvolvimento. Portanto, as LVs têm de ser consistentes entre si, deve haver consistência entre os protocolos de interação de cada CPD. Por exemplo, deve haver similaridade entre os traços semânticos dos conteúdos associados a um mesmo recurso expressivo, nos diferentes CPDs. Por exemplo, a cor vermelha não pode ser usada em um protocolo para expressar o conteúdo "faça" e, no outro, para expressar "não faça". Este nível de consistência tratamos por consistência inter-códigos.

Nas UTDs de cada CPD o léxico da linguagem é o mesmo, apenas são alternados os conteúdos expressos, em função da tarefa em desenvolvimento. Neste nível deve haver consistência no código, conforme foi postulada no Capítulo 4. Este nível de consistência tratamos por consistência intra-código.

Como foi dito, cada CPD terá seu próprio conjunto de componentes: a declaração de tipos e instâncias, o mapeamento e a gramática. Para possibilitar um controle visual do projetista e um tratamento automático da notação, os CPDs deverão ser numerados em sequência, a partir de 1. As UTDs, por utilizarem o mesmo léxico, serão restritas à gramática. Elas também deverão ser numeradas em sequência, a partir de 1, respeitando a numeração do CPD a que pertencem. Portanto, a numeração de uma UTD será sempre de dois números, na forma UTDn.m, onde n equivale ao número do CPD a que pertence e m é seu número de sequência. No que se refere às UTDs é importante observar que sempre existirá um contexto geral, ou seja, em uma determinada LV pode não ocorrer nenhuma UTD particular. Portanto, para manter a homogeneidade da numeração, o contexto geral será tratado como uma UTD número zero. À exceção da UTD número zero, os números sequenciais, tanto dos CPDs, como das UTDs, têm função apenas de identificação, sem nenhuma outra conotação particular, como por exemplo, a precedência de uma UTD sobre outra.

Os exemplos 6.4 e 6.5 ilustram a declaração de um CPD e de uma UTD, respectivamente. Neles, a declaração de cada componente é apresentada parcialmente. Um exemplo detalhado de especificação da STAG é apresentado no Anexo B. A inclusão da numeração do CPD em todos os seus componentes é opcional, pode ser colocada ou não, conforme seja mais confortável para o projetista. Trata-se de uma questão de controle visual. A numeração, que no exemplo foi apresentada apenas nos tópicos de cada componente, poderia se estender a todas as ocorrências de cada tópico. Ou seja, todas as linhas da declaração poderiam receber a numeração do CPD. Neste exemplo, assim como no Anexo B, entendemos que apenas a numeração dos tópicos seria suficiente para compreensão do leitor. Nos outros exemplos, ao longo deste trabalho, esta numeração foi desconsiderada, por se tratar quase sempre de exemplos parciais. No ambiente para geração de signos visuais a preocupação com a numeração deixa de existir, pois o controle será feito automaticamente à medida em que o projetista declare a numeração do CPD.

Deve-se observar que no Exemplo 6.5 as operações de movimentação e apagamento foram colocadas na mesma UTD, porque ambas desfrutam do mesmo contexto de seleção de objeto.

CPD 1 — Exemplo 6.4

1. Tipos de Dados do Sistema
    primitiva
1. Instâncias de Dados do Sistema
    primitiva: vértice, aresta, face
1. Tipos de Expressão do Sistema
    forma
1. Instâncias de Expressão do Sistema
    forma: asterisco, traço

1. Mapeamento de Tipos do Sistema
    primitiva <=> forma

1. Mapeamento de Instâncias do Sistema

    primitiva (vértice) <=> forma (asterisco)

1. Regras Gramaticais

UTD 1.0 — Exemplo 6.5

    1.0.R1. Criar [Operação = criar, Objeto] -->
        selecionar-ferramenta [Objeto] +
        gerar-primitiva [Instância-Objeto] +
        Estado-criação (inativo)
    1.0.R2. gerar-primitiva [Objeto = aresta] -->
        plano-interação (plano) +
        Intenção (indicar) +
        Estado-captura (ativo) +
        plano-interação (plano) +

Intenção (indicar) +
Estado-captura (inativo) +
Primitiva (aresta) +
Estado-seleção (ligado)

UTD 1.1.

1.1.R.1. Mover [Operação = mover, Critério-escolha] -->
selecionar-objeto [Critério-escolha] +
Intenção (movimentar) +
Estado-seleção (desligado)

1.1.R2. Apagar [Operação = apagar, Critério-escolha] -->
selecionar-objeto [Critério-escolha] +
Operação-edição (apagar) +
Intenção (indicar)

1.1.R3. selecionar-objetos [Critério-escolha = único] -->
Primitiva ( )
Intenção (indicar) +
Estado-seleção (ligado)

6.3.2. A Declaração de Tipos e Instâncias

A declaração de tipos é feita de acordo com a segmentação dos contínuos do conteúdo e da expressão, de usuário e sistema. Para cada tipo é feita a declaração de instâncias. Alguns tipos podem ter apenas uma instância. A declaração de tipos e instâncias obedece a forma abaixo. A especificação do tipo em separado (Exemplo 6.6) não é obrigatória para o processamento da notação, uma vez que a declaração de tipo está embutida na declaração de instâncias (Exemplo 6.7). Ela é apresentada em separado no sentido de facilitar a leitura.

Declaração de tipos
cor Exemplo 6.6

Declaração de instâncias
cor: azul, amarela, verde. Exemplo 6.7

Dois pontos fundamentais devem ser respeitados no que se refere à definição dos tipos: (a) os critérios utilizados na segmentação dos contínuos e (b) a definição do elemento do conteúdo a ser representado. Nos tópicos a seguir serão tratados cada um destes pontos.

6.3.2.1. Critérios para Segmentação

Os critérios para segmentação foram discutidos em detalhes no Capítulo 4. Aqui serão apenas citados em função da notação.

Definição de Tipos do Conteúdo do Sistema

O critério adotado para segmentação do contínuo do conteúdo do sistema foi o modelo ERA [Chen 76, Batini et alii 92]. Com base nele são declaradas as entidades (Exemplo 6.8), correspondentes às primitivas semânticas do sistema, seus atributos (Exemplo 6.9) e relacionamentos (Exemplo 6.10). Os relacionamentos e estados têm algumas particularidades, que serão apresentadas na Seção 6.3.2.2 sobre decisões sobre o conteúdo de representação.

Declaração de tipos de dados do sistema
primitivas Exemplo 6.8
força Exemplo 6.9
hierarquia Exemplo 6.10

Declaração de instâncias de dados do sistema

Primitivas: vértice, aresta, face
Força: peso, vento
Hierarquia: 1{vértice:0, aresta:1, face:1}
Relações de sobreposição
Primitiva <-- Força

Também são declaradas como entidades do sistema as primitivas de interação (Exemplo 6.11).

Declaração de instâncias do conteúdo do sistema
Primitivas-interação: marca, linha, polilinha, arco, círculo, área Exemplo 6.11

Além da declaração de tipos de dados do sistema, devem ser declaradas as operações, as variáveis de estados das entidades, as variáveis de estados correntes do sistema, as variáveis de estados de execução, os sinalizadores e as relações de sobreposição (no Anexo B encontra-se um exemplo detalhado de aplicação da STAG).
A relação entre o tipo de entidade Primitivas e o tipo de atributo Força deve ser declarada, embora não seja visualmente expressa, e, por isso, não será mapeada em um recurso expressivo. A declaração é necessária para apoiar a verificação de sobreposição de recursos expressivos. Ela será declarada à parte, no item Relações de Sobreposição.
Uma questão importante deve ser observada no que se refere às relações, elas podem ocorrer entre tipos, entre instâncias, ou entre tipos e instâncias. Por exemplo, na relação de sobreposição um tipo de uma variável de estado pode atuar sobre tipos ou instâncias de entidades. Por exemplo, o tipo Estado-seleção atua sobre o tipo Primitivas. Ou seja, ele pode atuar sobre todas as instâncias do tipo Primitivas. Porém, o tipo de atributo Força somente pode atuar sobre duas instâncias do tipo Primitivas: Primitivas (face) e Primitivas (aresta).

Definição de Tipos da Expressão do Sistema

Na segmentação do contínuo expressivo do sistema, os tipos são estabelecidas em função da análise do dispositivo de comunicação, o vídeo, apresentada no Capítulo 4 (Exemplo 6.12).

Declaração de tipos da expressão do sistema
forma Exemplo 6.12
cor
textura

Definição de Tipos do Conteúdo do Usuário

A segmentação assenta-se basicamente sobre os tipos de ação que o usuário pode expressar, considerando interfaces por menus e por MD, e a metáfora de extensão da mão, relativa ao dispositivo de mouse. Por exemplo, neste ambiente verificamos intenções de indicação, movimentação, ativação e interrupção. Por exemplo, uma indicação está ligada a um contexto de menus, botões ou quaisquer outros elementos de interface, que ofereçam um determinado conteúdo ao usuário, sobre o qual ele possa expressar escolha. Exemplos de tipos e instâncias do conteúdo do usuário são apresentados no Exemplo 6.13.

Declaração de instâncias do conteúdo do usuário Exemplo 6.13
intenção: indicar, movimentar, ativar, interromper

Definição de Tipos da Expressão do Usuário

Na segmentação do contínuo expressivo do usuário foram classificados os sinais emitidos pelo mouse (Exemplo 6.14). Como foi definido no Capítulo 4, neste trabalho nos restringimos ao tratamento de um mouse com dois botões.

Declaração de instâncias da expressão do usuário Exemplo 6.14

sinal: clique, pressão-e-arrasto, clique-duplo, clique-botão-direita

---

6.3.2.2. Definições sobre o Conteúdo de Representação

A variáveis de estado são elementos do conteúdo do sistema com características peculiares. Elas agem sobre as entidades, em alguns casos determinando suas características, em outros atuando como um protocolo de acesso para certas operações. Por exemplo, no primeiro caso, variáveis de estados do sistema podem agir como determinantes dos atributos das entidades, no momento de criação. Por exemplo, uma linha pode ter sua espessura e sua cor determinadas pelo valor corrente de variáveis de estados do sistema. Em nossa experiência constatamos que, em casos como este, a maioria dos sistemas não expressa as variáveis de estado atuantes num determinado momento, nem seus valores correntes. Em princípio, esta não é uma solução de projeto adequada, porque não sinaliza claramente o modelo funcional do sistema para o usuário.

No segundo caso, por exemplo, uma entidade pode ser sujeita a uma determinada variável de estado, que atue como um protocolo de liberação para alteração de forma. Esta entidade, a cada momento, estará liberada ou bloqueada para alteração de forma, em função do valor corente da variável. Obviamente os estados ativos, ou ligados, devem ser devidamente sinalizados pelo sistema. Em alguns casos as variáveis de estados funcionam como protocolo de entrada e saída para unidades de conversação, onde, entre a abertura e o fechamento do protocolo, várias operações podem ser executadas.

Na notação são declarados os tipos e instâncias das variáveis de estado (Exemplos 6.15 e 6.16). De forma semelhante à relação entre entidades e atributos, são declaradas as relações entre entidades e variáveis de estado, sendo especificado sobre quais entidades cada variável de estado atua (Exemplo 6.17). As declarações de relação entre entidades e variáveis de estado serão declaradas em separado, no item Relações de Sobreposição, porque não terão entrada no mapeamento. Essas relações não são visualmente expressas. No entanto, sua declaração é necessária para a verificação de sobreposição de recursos expressivos. Somente as instâncias de variáveis de estado terão entrada no mapeamento, sendo que, em geral, só o estado ligado é mapeado para um recurso expressivo (Exemplo 6.18), a representação do estado desligado é nula.

A declaração correta das variáveis de estado é um elemento extremamente forte no sentido do projetista expressar claramente o modelo funcional do sistema.

Declaração de variáveis de estado das entidades

estado-seleção Exemplo 6.15

Declaração de instâncias de variáveis de estado das entidades

estado-seleção: ligado, desligado Exemplo 6.16

Relações de sobreposição

primitiva <-- estado-seleção Exemplo 6.17

Mapeamento

estado-seleção <=> cor Exemplo 6.18

estado-seleção (ligado) <=> cor (amarelo)

Quanto ao relacionamento, existem dois tipos que devem ser declarados através da notação, os que não são espressos e os que são expressos através da linguagem. O primeiro caso, já comentado junto ao Exemplo 6.10 e novamente ilustrado no Exemplo 6.20, refere-se às declarações de relacionamento entre entidade e atributo. Neste caso, o relacionamento não é um conteúdo a ser expresso, os conteúdos expressos são as instâncias de primitivas e

as instâncias do atributo densidade. Esses relacionamentos são declarados em separadamente, nas Relações de Sobreposição. A declaração é necessária para a verificação de sobreposição de recursos expressivos. Na ocorrência de mais de um atributo incidindo sobre a mesma entidade, se ambos tiverem representação visual, estará ocorrendo sobreposição de recursos.
Quanto ao segundo caso, exemplificamos com um relacionamento entre entidades (Exemplo 6.19). O relacionamento hierárquico entre as entidades foi definido através de índices associados às mesmas, onde a entidade vértice (índice 0) é hierarquicamente superior às entidades aresta e face (índice 1). Este relacionamento pode ser expresso, por exemplo, através do recurso brilho, com um brilho mais intenso para a representação da entidade de mais alto nível hierárquico. Embora a declaração do tipo de relacionamento apresente as entidades envolvidas na hierarquia, o conteúdo a ser expresso com base nesta declaração não são as entidades, mas os níveis hierárquicos presentes no relacionamento. Neste caso a notação é indireta, uma vez que na declaração de tipos está sendo declarado o relacionamento (no Exemplo 6.19 é "hierarquia"), mas no mapeamento os recursos expressivos estão sendo associados aos níveis hierárquicos (Exemplo 6.21). Neste caso, os índices hierárquicos 0 e 1 são associados às instâncias expressivas, brilho forte e brilho normal.

Declaração de instâncias de dados do sistema
hierarquia: 1{vértice:0, aresta:1, face:1} Exemplo 6.19
Relações de atribuição
primitiva <-- densidade Exemplo 6.20
Mapeamento
hierarquia (1{vértice:0, aresta:1, face:1}) <=> brilho (forte:0, normal:1)Exemplo 6.21

Cada tipo de relacionamento a ser expresso é peculiar no que se refere ao elemento que será associado ao recurso expressivo. No caso do relacionamento hierárquico são expressos os diferentes níveis, por isso, quanto mais níveis de profundidade tiver a árvore hierárquica, mais difícil será expressá-los. No caso, por exemplo, de um relacionamento de grupo, bastará o uso de um único recurso que possa ser aplicado a todas as entidades do grupo. Por exemplo, mais uma vez pode ser usado o recurso brilho, quando todas as entidades pertencentes ao grupo terão um brilho forte.
A declaração de tipos e instâncias atende os requisitos 6.3.a a 6.3.d estabelecidos para a STAG. O requisito 6.3.e também será tratado através da declaração de tipos e instâncias.

6.3.3. O Mapeamento entre Elementos do Conteúdo e da Expressão

O mapeamento faz a correlação entre os planos do conteúdo e da expressão. A partir da declaração de tipos é feita a correlação entre as instâncias do conteúdo e as instâncias da expressão. Em geral, a correlação deve ser feita por um conjunto de regras que geram os tipos gerais da linguagem. Na LV, por ser uma linguagem bastante pobre em articulação (Capítulo 3), a correlação é de praticamente um para um, entre conteúdo e recurso expressivo. Neste sentido esta correlação torna-se um mapeamento.
O mapeamento associa conteúdo e expressão pelo sinal <=>, e tanto pode associar tipos do conteúdo e da expressão (Exemplo 6.22), como instâncias do conteúdo e da expressão (Exemplo 6.23). Todos os símbolos utilizados pela notação estão descritos na Tabela 6.4. Semelhante à declaração de tipos, a associação entre tipos não é obrigatória quanto ao aspecto de processamento da notação, uma vez que ela está implícita na associação entre tipos/instância. Mas ela facilita a averiguação visual do projetista, em termos de consistência. As instâncias são declaradas entre parênteses, ao lado do tipo.

Mapeamento entre tipos
material <=> cor Exemplo 6.22
Mapeamento entre instâncias
material (concreto) <=> cor (amarela) Exemplo 6.23

Atendendo ao requisito 6.3.d, a notação registra redundância de código e interação com outros códigos, permitindo a verificação de uso abusivo do código. Na codificação redundante é feita a associação do mesmo conteúdo a mais de um recurso expressivo, estes ligados pelo sinal <> (Exemplo 6.24). O Exemplo 6.25 mostra de que forma a notação registra a interação com outros códigos. O Exemplo 6.26 mostra indícios de uso abusivo de código.

Mapeamento entre instâncias
material (concreto) <=> cor (amarela) <> textura (pontilhada) Exemplo 6.24
primitiva-interação (polilinha) <=> forma-composta (botão) <> texto (rótulo) Exemplo 6.25
Mapeamento entre tipos
material <=> cor Exemplo 6.26
resistência <=> cor
agrupamento <=> cor

6.3.4. O Dicionário de Tarefas Simples

O dicionário de tarefas simples lista as tarefas qualificadas por elementos do modelo adotado para segmentação (modelo ERA). Cada elemento do modelo que particulariza a tarefa é denotado por um termo descritivo. O sinal = associa o termo descritivo a tipos/instâncias do conteúdo. O conjunto de informações vem dentro de colchetes, e as informações vêm separadas por vírgulas (Exemplo 6.27).

Dicionário de tarefas simples
Criar-vértice [Operação = criar, Objeto = vértice] Exemplo 6.27

6.3.5. As Regras Gramaticais

Cada regra contém um elemento simples do lado esquerdo. Cada lado esquerdo é um termo consistindo de um rótulo arbitrário e um conjunto de características contidos em colchetes, os quais podem conter qualquer número de informações que particularizam a tarefa, valoradas ou não valoradas. O lado esquerdo da regra é separado do lado direito por uma seta --> a qual denota reescritura. O lado direito da regra contém uma sequência ordenada de termos (exatamente no mesmo formato do lado esquerdo) e símbolos terminais (Exemplo 6.28).

Regras Gramaticais
Criar-vértice [Operação = criar, Objeto = vértice] -> Exemplo 6.28
selecionar-ferramenta [Objeto] +
gerar-primitiva [Objeto] +

Para expandir a regra, qualquer informação particularizante não valorada deve ser associada a valores. Quando a associação estiver completa, cada termo deve denotar um objeto gramatical único. Se um termo se refere a uma tarefa simples (dando o rótulo genérico de tarefa) então as características valoradas vão especificar exatamente uma das entradas no dicionário de tarefas simples. Aonde o termo se refere a um símbolo não terminal, este termo não terminal aparecerá, ele próprio, como o lado esquerdo de uma regra.
A gramática é gerativa, a saída da gramática é uma lista de tipos e instâncias do conteúdo utilizados por usuário e sistema durante a execução de uma tarefa. Estes tipos e instâncias do conteúdo são mapeados em tipos e instâncias da expressão. De forma semelhante à TAG, a STAG tem capacidades estritamente livres de contexto, apesar de uma certa parametrização dos constituintes gramaticais. Cada regra ordinária pode ser expandida em um número finito de regras de reescritura de níveis simples (livres de contexto), simplesmente associando valores às características em todos os caminhos possíveis. Ao final deste capítulo a Tabela 6.4 apresenta os símbolos utilizados pela STAG e a Tabela 6.5 apresenta o formato geral da STAG.

6.4. Avaliação sobre o Potencial da STAG

Um dos pontos de maior dificuldade na codificação linguística é a expressão correta do conteúdo do sistema. Na notação proposta o ponto de apoio mais forte neste sentido é o tratamento independente entre conteúdo e expressão, conforme postula a teoria semiótica. O tratamento do conteúdo de forma independente facilita a identificação do elemento a ser expresso e a expressão adequada. O ponto mais delicado neste processo é a segmentação do contínuo de conteúdo. Como em toda linguagem, o critério adotado na segmentação tem um forte

impacto no resultado final da linguagem. Todo sistema gramatical é determinado pelo tipo de segmentação de mundo que seu léxico reflete.
Neste tópico vamos avaliar de que forma a notação pode auxiliar o projetista neste processo. De modo geral podemos dizer que o grande apoio que a notação oferece é forçar o projetista a especificar os tipos expressivos e do conteúdo, do usuário e sistema. Mesmo que a notação não dê conta de controlar automaticamente as possíveis inadequações, o simples fato de o projetista ser forçado a pensar a interface como uma linguagem e ser forçado a especificar os elementos básicos desta linguagem, fará com que ele próprio se dê conta de boa parte dos problemas. O enfoque sistemático da linguagem de interface é o grande ganho do projetista, no uso da notação.

6.4.1. Suporte à Expressão do Modelo Funcional

Dois pontos de dificuldade podem ser ressaltados no que se refere à expressão do modelo funcional: (a) a identificação do conteúdo a ser expresso, (b) a expressão adequada do conteúdo. A seguir vamos comentar de que forma a notação pode auxiliar em cada caso.

6.4.1.a. A identificação do conteúdo a ser expresso

A identificação do conteúdo a ser expresso depende em muitos casos da sensibilidade e do cuidado do projetista. Como foi apresentado no tópico 6.4.a, quando o usuário deseja fazer uma seleção de objetos ele não expressa "fazer uma seleção", ele expressa "fazer uma determinada seleção", uma vez que ele pode usar diferentes critérios para selecionar conjuntos. Ao formular sua intenção o usuário decide se o objeto da seleção é único ou é um conjunto, podendo lançar mão de critérios intensionais, por exemplo todas as arestas, ou extensionais físicos, por exemplo objetos localizados em uma determinada área. O Exemplo 6.29 ilustra a expressão do estado de seleção (expressão do sistema) e da ativação da seleção (expressão do usuário). A diferença deste exemplo para o Exemplo 6.1 (b) é que este já está no formato da notação, enquanto o anterior estava mais próximo do formato da TAG.

Tipos de Dados do Sistema Exemplo 6.29
    primitivas
Instâncias de Dados do Sistema
    primitivas: vértice, aresta, face
Declaração de Variáveis de Estado das Entidades
    estado-seleção
Declaração de Instâncias de Variáveis de Estado das Entidades
    estado-seleção: ligado, desligado
Relações de Atribuição
    primitivas <-- estado-seleção
Instâncias da Expressão do Sistema
    cor: amarela, verde, azul, vermelha
Mapeamento do Sistema
    estado-seleção ( ) <=> cor ( )
    estado-seleção (ligado) <=> cor (amarela)

Tipos do Conteúdo do Usuário
    intenção
Instâncias do Conteúdo do Usuário
    intenção: indicar, indicar-grupo
Tipos do Conteúdo do usuário
    sinal
Instâncias da Expressão do Usuário
    sinal: clique, pressão-e-arrasto
Mapeamento do Usuário
    intenção <=> sinal
    intenção (indicar) <=> sinal (clique)
    intenção (indicar-grupo) <=> sinal (pressão-e-arrasto)

Regras Gramaticais
    ação-seleção [Caso = uma-entidade-física] ---> intenção (indicar) + estado-seleção (ligado)
    ação-seleção [Caso = entidades-físicas-dispersas] --> intenção (indicar-grupo) +

estado-seleção (ligado)

Neste exemplo pode ser observado que a expressão do estado de seleção está sendo feita pelo sistema através do recurso cor, e a expressão da operação de seleção está sendo feita pelo usuário através de manipulações por cliques ou pressão-e-arrasto, conforme o objeto seja único ou conjunto. A gramática está mapeando a intenção do usuário em tipos/instâncias de conteúdo adequados.

Se não fossem definidos tipos de conteúdo específicos para cada intenção, a gramática iria mapear a intenção do usuário em um único tipo/instância de conteúdo, por exemplo indicação. Este tipo/instância de conteúdo deveria ser associado a diferentes formas de expressão no sentido de atender às diferentes formas de ativação do usuário, o que não seria uma estruturação adequada da linguagem. A notação não pode determinar ao projetista como definir seus tipos de conteúdo, mas ela auxilia no sentido de que tipos mal definidos provocam inconsistências no mapeamento, que neste caso específico poderiam inclusive ser verificadas automaticamente.

Vamos analisar um segundo exemplo, onde a notação pode auxiliar o projetista a esclarecer a estrutura de sua linguagem. Este exemplo foi observado no aplicativo MsDraw, cujos dados de análise estão apresentados no Anexo A. Dois aspectos particulares foram observados com relação a este aplicativo. O primeiro deles refere-se à definição da cor como conteúdo, além de recurso expressivo. No MsDraw, os atributos aplicados sobre as primitivas (como cor e tipo de linha) são estéticos. É diferente, por exemplo, do MTool, onde a cor é um recurso expressivo do sistema, que pode ser utilizado para expressar atributos do objeto, com um estado, um material ou uma propriedade. No caso do MsDraw a cor enquanto um elemento abstrato é ela própria um atributo, expresso pelo elemento físico cor. Por exemplo, um polígono amarelo é uma primitiva (polígono) com um atributo estético (cor amarela). A mesma primitiva poderia ter outros atributos estéticos, como por exemplo espessura de linha largura 2, expresso por uma linha de tantas polegadas de espessura. No Exemplo 6.30 a cor é definida como atributo e como recurso expressivo.

Outra particularidade com relação aos atributos do MsDraw é que os atributos não são associados às primitivas como um todo, mas a componentes das primitivas (interior e contorno), os quais têm atributos específicos. Neste caso, a declaração da relação dos atributos com as entidades exige que o projetista defina interior e contorno com entidades independentes. A notação não prevê uma forma de especificação exclusiva para esta situação, onde interior e contorno são um desdobramento da entidade polígono, específico para associação de atributos. No entanto, para análise de sobreposição do recurso expressivo é necessário definir algum tipo de relação entre estas entidades. Mais uma vez, as exigências da notação vão alertar o projetista sobre as ocorrências na sua linguagem visual. No Exemplo 6.30, foi declarada uma relação de atributo, entre as primitivas originais e seu desdobramento. Esta declaração atende apenas aos objetivos de conferência, ela não será associada a recurso expressivo.

Curiosamente no aplicativo em questão a idéia de interior e contorno também está associada à primitiva poligonal. Se uma poligonal for traçada com opção de preenchimento (Fill) o sistema define uma fronteira ligando o primeiro e o último ponto da poligonal e preenche a figura (Figura A.2). Portanto, por coerência com a realidade do sistema, no Exemplo 6.30 a poligonal também foi associada às entidades interior e contorno.

Tipos de Dados do Sistema — Exemplo 6.30

- primitivas
- interior
- contorno
- cor

Instâncias de Dados do Sistema

- primitivas: linha, polígono, poligonal

Relações de Atribuição

- primitiva (polígono) <-- interior
- primitiva (polígono) <-- contorno
- primitiva (poligonal) <-- interior
- primitiva (poligonal) <-- contorno
- interior <-- cor
- contorno <-- cor

Tipos da Expressão do Sistema

- forma
- cor

Mapeamento

- primitivas ( ) <=> forma ( )
- cor ( ) <=> cor ( )

Vamos exemplificar ainda uma terceira situação onde a notação pode auxiliar o projetista a expressar adequadamente o modelo funcional do sistema. O exemplo trata a questão das categorias ontológicas. Como foi dito, neste trabalho partimos do pressuposto de que a linguagem de interação deve expressar corretamente o comportamento do objeto gráfico. Vamos exemplificar com o desenho de um polígono, através de linhas. Em alguns sistemas, como por exemplo o MsDraw [Word 97], a representação terá somente a aparência visual de um polígono, sendo tratada como um conjunto de linhas. Em outros sistemas, como por exemplo o CorelDraw [CorelDraw 5.0], as linhas traçadas são unidas automaticamente, formando um polígono visual, com um comportamento coerente.
Se o usuário estiver trabalhando conforme o exemplo acima, no MsDraw, e solicitar uma operação de agrupamento, o sistema poderá responder com dois comportamentos. Um deles, seria um agrupamento visual para efeito de operações de edição (como mover ou apagar o grupo), mas mantendo um comportamento do objeto de acordo com seu traçado original (um conjunto de linhas). Neste caso, se o usuário tentasse preencher o polígono ele não conseguiria. O outro, seria o sistema agrupar as primitivas iniciais, transformando a representação gráfica em um polígono de fato. Neste caso, a operação de agrupamento produziria uma nova entidade, a partir de um conjunto de primitivas linha ela produziria uma primitiva polígono. Em qualquer caso, em função da notação, o projetista deverá definir claramente o tipo de primitiva está sendo tratada. Isto o forçará a especificar claramente o comportamento da representação visual em cada nível de categorização.

6.4.1.b. A expressão adequada do conteúdo

Através das UTDs, a notação permite o tratamento de unidades expressivas especiais. A declaração das variáveis de estado do sistema e sobre quais entidades elas atuam, permite algum nivel de averiguação automática de sobreposição de recursos e inconsistência na linguagem. Porém, essa averiguação não pode ser completa se a unidade da linguagem, durante a qual o estado está atuando, não for tratada de forma particular, como um contexto temporário. Uma heurística que faça uma avaliação simples de atribuição de estado a uma entidade, sem considerar que o estado pode estar ocorrendo em uma unidade de diálogo particular, está sujeita a concluir erroneamente uma sobreposição de recursos expressivos.
Como já foi dito, o estado de seleção funciona como um protocolo de abertura para uma unidade de discurso, dentro da qual todo o contexto pode ser adaptado, mascarando temporariamente algumas informações. Também há caso de estados que atuam sobre uma mesma primitiva mas não simultaneamente. Através da UTD estes casos serão automaticamente registrados.

6.4.2. Suporte à Verificação de Inconsistências Intra-Código

Na especificação da LV o usuário dispõe de diretrizes de orientação para definir as associações entre conteúdo e expressão. Através da notação, diferentes tipos de inconsistências podem ser verificados sobre a utilização dos recursos expressivos. Para isto, o usuário dispõe de heurísticas de verificação para serem aplicadas sobre a STAG. Ambas, as diretrizes de orientação e as heuristicas de verificação, podem ser implementadas em um ambiente de auxílio à especificação de LVs por MD. Além de auxiliar a especificação da LV, a notação deve permitir algumas verificações automáticas de inconsistências. Obviamente as heurísticas de verificação são definidas com base nas diretrizes de orientação. Por exemplo, uma das regras de orientação recomenda que não se utilize o recurso cor para expressar as instâncias de um conjunto de alta cardinalidade. A notação não impede que o projetista infrinja esta recomendação, mas as heurísticas de verificação podem assinalar sua ocorrência.
Através das heurísticas, podem ser verificados automaticamente o abuso de código e a sobreposição de recursos visuais, considerando os casos em que a sobreposição causa interferência e os casos em que não causa. Recursos visuais utilizados sobre a mesma forma, mas não simultaneamente, podem não causar interferência mútua. O Exemplo 6.31 ilustra uma ocorrência de abuso de código e de sobreposição. O abuso decorre do uso excessivo do recurso visual cor. A sobreposição ocorre porque a entidade face é expressa por cor, assim como o material associado a ela. Quando o material é associado à face, ou quando ela é selecionada, um código de cor anula o outro. As heurísticas para verificação da STAG são apresentadas no Capítulo 7.

Mapeamento de Tipos do Sistema
primitivas (face) <=> cor ( ) Exemplo 6.31
estado-seleção ( ) <=> cor ( )
material ( ) <=> cor ( )

Outro tipo de verificação possível é a regularidade no uso do código. A notação provê uma forma de associação direta, que pode ser utilizada entre instâncias expressivas interligadas (botão e botão-sombreado), tratada como encapsulamento. No Exemplo 6.32 é apresentado o encapsulamento entre a instância da expressão forma-composta (botão) e sua instância da expressão interligada forma-composta (botão sombreado). A instância primitiva-interação (linha) é expressa através da instância expressiva forma-composta (botão). A instância estado-criação (ativo) é expressa através de sua instância expressiva interligada forma-composta (botão-sombreado). Isto significa

que quando o botão é clicado, o sistema entra em estado de criação de primitiva. A utilização da notação (botão #) associada ao estado-criação garante que será usado o recurso expressivo interligado. O encapsulamento garante maior regularidade no código linguístico visual e simplificação da notação.

Declaração de instâncias expressivas do sistema

forma-composta: botão # botão sombreado Exemplo 6.32

Mapeamento

primitiva-interação (linha) <=> forma-composta (botão)

estado-criação (ativo) <=> forma-composta (botão #)

6.4.3. Suporte à Verificação de Inconsistências Inter-Códigos

Uma das premissas sobre as quais este trabalho foi proposto era que a linguagem de interface deveria atender as etapas de desenho e modelagem sucessivas do objeto em foco. Nos SMG, o processo de geração do modelo geométrico se divide entre tarefas de criação e edição do objeto. Neste processo interativo o usuário estará frequentemente alternando entre o desenho e a construção do objeto, tanto na criação como na edição. Esta é uma questão importante porque de acordo com a abordagem semiótica, esta alteração, que em uma abordagem tradicional seria parte do contexto de uso (pragmática), passa a fazer parte do conteúdo semântico do sistema, ou seja, do conteúdo a ser transmitido. Portanto, a linguagem de interface, deve permitir a expressão da alternância entre processos.

Outro ponto que também já foi abordado é a coexistência de dois modelos semânticos sob a mesma representação visual. Durante a interação o usuário também estará alternando entre dois modelos semânticos, e esta alternância também deverá ser transmitida pela linguagem de interface. Neste caso, a alternância poderá ser sinalizada através das entidades do sistema. Quando o usuário muda de um sistema semântico para outro, embora o traçado na tela permaneça o mesmo, as primitivas do modelo semântico são outras, e portanto cada traço passa a ser interpretado de forma diferente, sofrendo um tratamento diferenciado em função do modelo semântico ativo. Por exemplo, uma aresta no modelo de engenharia equivale ao lado de um elemento no modelo de elementos finitos.

No que se refere a esta alternância de modelos na interação, o que se espera da linguagem de interface é a comunicação ao usuário sob qual modelo ele está interagindo, além da regularidade entre os protocolos utilizados em cada modelo. Utilizando o exemplo acima, se o usuário puder movimentar uma aresta no modelo de engenharia, e puder movimentar o lado de um elemento no modelo de elementos finitos, espera-se que os protocolos de movimentação sejam semelhantes, para que o usuário possa inferir e prever as ações de interação com o sistema. Ao mesmo tempo, através da linguagem, o projetista deve sinalizar claramente ao usuário quando ele está interagindo com uma aresta e quando está interagindo com o lado de um elemento. A estrutura hierárquica da notação, através dos CPD, permite a especificação de diferentes linguagens interativas sobre a mesma representação visual. A notação permite que seja averiguada a consistência entre os diferentes CPDs.

6.4.4. Suporte à Interação com Outros Códigos

A verificação de interação com outros códigos pode ser feita em dois pontos. Um deles, através das redundâncias declaradas no mapeamento, onde o projetista declara a redundância do código visual com outro código. Outro ponto, através das regras gramaticais, onde os tipos expressivos visuais não forem suficientes será necessário complementar a regra através de outro código. Este tipo de verificação é visual, não é possível executar esta verificação automaticamente. O Exemplo 6.33 ilustra esta situação. Na regra gramatical, a ação de selecionar-ferramentas é desdobrada em uma indicação de uma primitiva de interação. Neste exemplo, o tipo discreto primitiva-interação é mapeado em um código textual, através de botões. Também a ação de apagar objeto é desdobrada, entre outros, em uma instância de operação-edição, a qual é mapeada em um código textual, através de menus.

Mapeamento Exemplo 6.33

primitivas-interação (linha) <=> forma-composta (botão) <> texto (rótulo)

operação-edição (apagar) <=> texto (menu-hierárquico)

Regras gramaticais

apagar [Operação = apagar, Critério-escolha] --> selecionar-objeto [Critério-escolha] +
Operação-edição (apagar) +
Intenção (indicar)

6.4.5. Suporte à Geração de Ambientes Customizáveis

A codificação dinâmica é um recurso utilizado nos aplicativos gráficos, em situações onde o conteúdo a ser expresso não pode ser antecipado pelo projetista. Um exemplo é a codificação em tempo de execução, quando o aplicativo oferece ao usuário a possibilidade de definir novas instâncias de um determinado tipo de atributo e a sua associação ao recurso expressivo. Por exemplo, o usuário pode especificar novas instâncias de tipos de solo e expressá-las através de cor, onde cada região de solo será traduzida em uma cor diferente, na representação gráfica do artefato de engenharia. Este é um caso em que as instâncias de regiões de solo não podem ser antecipadas. Somente o usuário poderá definí-las no momento de construção da sua representação gráfica. Se a representação visual desta informação for relevante, o projetista deve oferecer ao usuário uma forma de codificá-la durante o uso do aplicativo. Uma solução possível é o projetista oferecer ao usuário um mostruário de cores, para que ele as associe a cada nova instância de solo utilizado.

Esta solução está sujeita a alguns problemas, como por exemplo a insuficiência do número de cores oferecidas no mostruário, ou a reutilização de cores pelo usuário. Esta última ocorrência pode ser evitada a partir de um controle do aplicativo, mas com isto a responsabilidade sai da competênica da linguagem de interface, passando para o aplicativo. Em tempo de especificação da linguagem de interface, o que pode ser avaliado é se as cores disponibilizadas pelo projetista não vão causar nenhuma inconsistência. Através da notação pode ser avaliado se aquele conjunto de instâncias expressivas pode causar algum tipo de inconsistência, independente das instâncias do conteúdo às quais serão associadas. A avaliação é feita sobre o mapeamento declarado entre os tipos da expressão e do conteúdo, e pelos relacionamentos do tipo do conteúdo com outros tipos do conteúdo.

Como a codificação dinâmica é um caso particular com relação às interfaces por MD, sua notação específica não foi colocada no formato geral da notação. Os Exemplos 6.34 e 6.35 mostram o mapeamento entre conteúdo e recurso expressivo. O Exemplo 6.35 mostra a associação entre um conjunto dinâmico de instâncias do conteúdo, a ser definido em tempo de execução, representado pelo sinal @, e um conjunto de instâncias expressivas.

Mapeamento de tipo

solo <=> cor Exemplo 6.34

Mapeamento de instância

solo (@) <=> cor (verde, amarelo, azul, branco) Exemplo 6.35

De modo geral, a notação pode funcionar como um controlador de consistência da linguagem de comunicação em ambientes extensíveis pelo usuário. Ela permite controles que podem orientar o projetista, no sentido de manter a consistência da linguagem em suas extensões.

6.4.6. Uso da Notação para Análise da Linguagem Visual

Nesta Seção vamos fazer um breve comentário sobre a atuação da notação, quando usada para análise da LV. Serão avaliadas duas situações particulares, que foram apontadas no Estudo de Caso apresentado no Capítulo 5. Uma delas refere-se a usos do aplicativo não previstos pelo projetista. Como foi dito, o sistema implementado registra o interpretante do projetista sobre o problema. Se, durante a análise de um aplicativo implementado, for detectada uma utilização não prevista, a análise estará registrando um interpretante do usuário, não previsto pelo projetista. Por exemplo, o aplicativo MTool (97) não oferece

operações de alteração de forma, portanto sua linguagem de interface não oferece esta operação. Porém, através de um uso alternativo do sistema, o usuário pode simular a alteração de forma, movimentando um vértice. Este uso alternativo equivale a um interpretante do usuário, sobre a forma como o sistema resolve o problema de modelagem geométrica. Esse interpretante do usuário é registrado quando a notação é utilizada para análise da LV em uso.
A outra situação avaliada, também para o aplicativo MTool, refere-se ao uso da opção "leave original" nas operações de movimentação. No interpretante do projetista o que deveria ser uma intenção de "copiar e mover a cópia" foi abreviado para uma intenção de "mover uma cópia", e invertido para "mover deixando o original". O aplicativo permite outras operações utilizando a mesma opção, como rotação e espelhamento. Neste caso também ocorre uma simulação, uma vez que a operação ativada corresponde a uma sequência de duas operações. A diferença neste caso é que o uso simulado não é um interpretante do usuário; é um interpretante do projetista. Em princípio esta solução significaria um aumento na carga cognitiva imposta ao usuário, porém, aparentemente, esta forma de raciocínio é bem absorvida pelos usuários.
Na especificação da LV, a regra gramatical deveria mapear a intenção de "mover" em uma operação simples de movimentação; e deveria mapear a intenção de "mover deixando o original" em uma dupla de operações, uma cópia e uma movimentação. Mas, a intenção do usuário (que distingue as operações) será expressa através da alteração do valor corrente de uma variável de estado do sistema. Ou seja, no momento de ativação da movimentação, o usuário deverá "ligar" a opção de "leave original". Com a opção ligada, ao final da operação o sistema mostrará uma duplicata do objeto, movimentada, expressando a ocorrência de uma cópia. É interessante reforçar que ao solicitar a movimentação o usuário expressa somente o critério para definição do objeto a ser movimentado, a operação de cópia não é expressa pelo usuário. Internamente ao programa, a intenção do usuário é traduzida em "mover" ou "copiar e mover", em função do valor corrente da variável de estado.
Ambos os casos serão detalhados no Anexo C, onde é apresentado um exemplo de aplicação da STAG para análise de interfaces.

6.5. Avaliação sobre as Restrições da STAG

6.5.a. Sobre o que é necessário expressar
A notação auxilia o projetista a tratar a linguagem de interface de forma sistemática, auxiliando-o a perceber eventuais falhas na linguagem. Porém, a notação não tem mecanismos de detectar da adequação do conteúdo expresso. Por exemplo, se a manipulação de uma primitiva está sujeita a determinadas variáveis de estado, do sistema ou da própria primitiva, é conveniente que estas variáveis sejam sinalizadas para o usuário. Mas a notação não tem mecanismos de conferência dessas ocorrências. Como foi dito, ela auxilia no sentido de que o tratamento sistemático facilita ao projetista a percepção de eventuais faltas. Neste exemplo, a notação auxilia o projetista a perceber a especificação incompleta, embora essa incompletude não possa ser apontada automaticamente.

6.5.b. Segmentação dos contínuos
Embora já tenha sido citado em alguns pontos, vale relembrar que a STAG não obriga o projetista a nenhuma solução. Em função de sua estrutura, ela apenas facilita uma especificação consistente da LV. Apesar dos recursos estruturais da notação, ela é bastante dependente do tipo de segmentação que o projetista fará dos contínuos, especialmente o contínuo de conteúdo de sistema, já que nos contínuos de expressão a segmentação é mais constante.
Vamos exemplificar o problema através da segmentação adotada. Classificamos os sinais do mouse em um tipo "sinal" com as instâncias "clique, pressão-e-arrasto, clique-duplo e clique-botão-direita" (Exemplo 6.36). Não existe nenhum critério claro que justifique a classificação dos sinais do mouse desta forma ou em quatro tipos com uma instância cada um (Exemplo 6.37). No mapeamento as instâncias de sinal são associadas a instâncias do conteúdo como, por exemplo, indicação (unária), indicação (enária), movimento (movimento) e outras. Caso tivéssemos classificado cada sinal do mouse como um tipo, para manter a regularidade da linguagem, no mapeamento estes tipos deveriam ser associados a tipos do conteúdo. Portanto, um único sinal do mouse deveria ser associado ao tipo indicação. Se o sinal clique fosse associado ao tipo indicação, ambas indicações unária e enária seriam expressas por cliques. Mas o mais adequado é que a indicação unária seja expressa por clique e a indicação enária por pressão-e-arrasto, quando pode ser utilizado um retorno de bounding-box. A classificação do Exemplo 6.36 é mais adequada ao tratamento posterior que será dado sobre o tipo.

Instâncias da expressão do usuário — Exemplo 6.36

sinal: clique, pressão-e-arrasto, clique-duplo, clique-botão-direita

Instâncias da expressão do usuário — Exemplo 6.37

clicar: clique
pressionar-e-arrastar: pressão-e-arrasto

clicar-duplo: clique-duplo
clicar-botão-direita: clique-botão-direita

De modo geral, a notação busca incorporar na linguagem de interação categorizações pré-existentes e conhecidas pelo usuário, como por exemplo, através o uso do modelo ERA para segmentar o contínuo de conteúdo do sistema. Esta categorização, que pode ser reconhecida pelo usuário, torna a linguagem mais adequada em termos cognitivos.

6.5.c. Interfaces por eventos assíncronos

A STAG não é adequada para interfaces por eventos assíncronos. Para este tipo de interfaces não é possível fazer o controle da interação, uma vez que o retorno nem sempre é imediato à ativação. Nestes casos, a gramática pode atuar em pequenos núcleos onde seja garantido o sincronismo.

6.5.d. Controle de Iteração

Por ser livre de contexto a STAG não registra determinadas iteração enárias que ocorrem durante alguns processos, como, por exemplo, a criação de uma polilinha para a qual o usuário especifica vários pontos. A restrição não causa problemas com relação ao objetivo da notação, uma vez que este tipo de controle não é fundamental em termos de averiguação da consistência do código. O objetivo da notação é registrar de que forma sistema e usuário se expressam a cada iteração. Se a expressão se mantém, o número de iterações não faz diferença.

Tabela 6.4.

Especificação dos Sinalizadores Utilizados na STAG

Sinal de Instanciação

( ) - Expressa uma relação de pertinência de conjunto, onde uma instância pertence a um conjunto de um determinado tipo, do conteúdo ou da expressão
- Lê-se: "contém uma instância do tipo que o precede"
- Forma geral: tipo (instância)
- Exemplo: cor (amarelo)

Sinal de Correlação

<=> - Associa membro precedente a membro sucessor, membro precedente será tipo ou tipo(instância) do conteúdo, membro sucessor será tipo ou tipo(instância) da expressão
- Lê-se: "expresso por"
- Forma geral:
  tipo-conteúdo (instância-conteúdo) <=> tipo-expressivo (instância-expressiva)
- Exemplo: material (concreto) <=> cor (amarelo)

Sinal de Atribuição

<-- - Atribui membro sucessor a membro precedente, membro sucessor será tipo ou tipo-instância de atributo, membro precedente será tipo ou tipo-instância de expressão
- Lê-se: "tem como atributo"
- Forma geral:
  tipo-entidade (instância-entidade) <-- tipo-atributo (instância-atributo)
- Exemplo: face (face) <-- densidade (média)

Sinal de Conjunção

<> - Conjuga dois tipos expressivos, ou dois tipos(instâncias) expressivos. Utilizado para expressar redundância de código expressivo
- Lê-se "e"
- Forma geral: tipo-conteúdo ( ) <=> tipo-expressivo1 ( ) <> tipo-expressivo2 ( )
- Exemplo: vértice (vértice) <=> forma (linha) <> cor (amarelo)

Sinal de Atribuição de Nível

: - Atribui um índice a um tipo ou tipo(instância) do conteúdo, referente a um determinado nível em uma relação, por exemplo, hierárquica
- Lê-se: "índice"
- Forma geral: tipo(instância):n, onde n é o número do nível
- Exemplo: vértice(vértice):0

Sinal de Delimitação

{ } - Delimita conjuntos de tipos ou tipos(instâncias) de conteúdo ligados por um determinado tipo de relacionamento, o qual é definido pela indexação dos elementos do conjunto.
- Lê-se: "contém os elementos correlacionados"
- Forma geral: relacionamento1 {tipo1:0, tipo2:1, tipo3:1}
- Exemplo: relacionamento {vértice(vértice):0, aresta(aresta):1, face(face):1)}

Sinal de Ligação Obrigatória

- Associa instância de expressão de retorno a um tipo(instância) expressivo
- Interpreta-se: "sempre que o tipo(instância) precedente for utilizado, a instância sucessora também será utilizada, automaticamente". Ambas instâncias, precedente e sucessora, devem pertencer ao mesmo tipo, uma vez que este sinal é utilizado na declaração de instâncias.
- Forma geral: tipo: instância1 # instância2
- Exemplo: forma-composta: botão # botão sombreado

Sinal de Substituição de Conjunto

@ - Representa um conjunto abstrato de instâncias do conteúdo, a serem especificadas e associadas ao conjunto de instâncias expressivas, um para um, em tempo de execução
- Interpreta-se: "conjunto abstrato de instâncias, do tipo do conteúdo precedente"
- Forma geral:
  tipo-conteúdo (@) <=> tipo-expressão (instância1, instância2, instância3)
- Exemplo: material (@) <=> cor (verde, amarelo, azul, branco)

Tabela 6.5.
Formato Geral da STAG

CPD 1
Declaração de Tipos e Instâncias do Conteúdo

.....
1. Tipos de Dados
    <entidades>
    <atributos>
    <relacionamentos {:0, ... :n}>
1. Instâncias de Dados
    <entidade: instância-1, ... instância-n>
    <atributo: instância-1, ... instância-n>
    <relacionamento: 1{entidade:0, ... entidade:n}, 2{entidade:0, ... entidade:n}>
.....
1. Tipos de Operações
    <tipo-operação>
1. Instâncias de Operações
    <tipo-operação: instância-1, ... instância-n>
.....
1. Tipos de Variáveis de Estados das Entidades
    <estado>
1. Instâncias de Variáveis de Estados das Entidades
    <estado: instância-1, ... instância-n>
.....

.....
1. Tipos de Variáveis de Estados Correntes do Sistema
<estado>
1. Instâncias de Variáveis de Estados Correntes do Sistema
<estado: instância-1, ... instância-n>
.....
1. Tipos de Variáveis de Estados de Execução do Sistema
<estado>
1. Instâncias de Variáveis de Estados de Execução do Sistema
<estado: instância-1, ... instância-n>
.....
1. Tipos de Sinalizadores de Apoio
<sinalizador>
1. Instâncias de Sinalizadores de Apoio
<sinalizador: instância-1, ... instância-n>
.....
1. Relações de Atribuição
<entidade <-- atributo>
.....
1. Relações de Sobreposição
<entidade <-- estado>
.....

Declaração de Tipos e Instâncias da Expressão

.....
1. Tipos da Expressão Visual
<tipo-visual>
1. Instâncias da Expressão Visual
<tipo-visual: instância-1, ... instância-n>
.....
1. Tipos da Expressão Manual
<tipo-manual>
1. Instâncias da Expressão Manual
<tipo-manual: instância-1, ... instância-n>
.....

Mapeamento de Tipos e Instâncias

.....

1. Mapeamento de tipos
<tipo-conteúdo <=> tipo-expressão>
1. Mapeamento de instâncias
<tipo-conteúdo (instância-conteúdo) <=> tipo-expressão (instância-expressão)>
<tipo-relacionamento ({entidade:0, ... entidade:n}) <=>
tipo-expressão ({instância-expressão:0, ... instância-expressão:n})>
.....

Regras Gramaticais

.....

1. Regras Gramaticais
UTD 1.0
1.0 <Tarefa [tipo-operação, tipo-dado, tipo-sinalizador] -->
tipo-operação + tipo-dado + tipo-sinalizador
.....

Um dos objetivos deste trabalho foi oferecer a projetistas de interfaces um ambiente que permita a especificação automática de LVs, de acordo com princípios semióticos, sem exigir a competência de um semioticista. Este capítulo apresenta uma proposta para um ambiente de geração de signos visuais (AGSV), o qual incorpora as diretivas de orientação para codificação da linguagem visual (DOCVL) e as heurísticas de avaliação da linguagem visual (HALV). Esta proposta não esgota o projeto da ferramenta, ela está sendo apresentada no sentido de antecipar a viabilidade da implementação.

7.1. Proposta para Aplicação da STAG

Este capítulo apresenta uma proposta para um ambiente de geração de signos visuais (AGSV), através da utilização automática da STAG. A STAG se propõe a auxiliar o projetista a especificar LVs por MD para SMG, através de um tratamento sistemático dos recursos expressivos, objetivando expressar da melhor maneira possível o modelo funcional do sistema. A notação incorpora princípios semióticos, de tal forma que através de seu uso o projetista estará automaticamente projetando de acordo com estes princípios. O AGSV propõe a implementação da notação em conjunto com as diretivas de orientação para codificação da linguagem visual (DOCLV) (Seção 4.3.4) e as heurísticas de avaliação da linguagem visual (HALV). O objetivo é oferecer ao projetista uma verificação de consistência da LV durante a fase de projeto.

Como foi dito, a STAG não obriga soluções; através de suas heurísticas ela apenas adverte para possíveis inadequações, advertências que o projetista poderá ignorar. Mesmo que a notação não permita controlar automaticamente todas as possíveis inadequações, o simples fato de o projetista ser forçado a pensar a interface como uma linguagem e ser forçado a especificar os elementos básicos da linguagem de forma sistemática, fará com que ele próprio se dê conta de boa parte dos problemas. O enfoque sistemático da linguagem de interface é o grande ganho do projetista, no uso da notação.

O propósito inicial deste trabalho foi uma abordagem teórica ao problema de especificação sistemática de LVs por MD. Como resultado desta análise teórica foi a proposta a STAG. A proposta para o AGSV, que implementa a notação, é apenas um complemento ao propósito inicial. Neste sentido, foge ao escopo deste trabalho sua implementação. Neste capítulo está sendo apresentada uma proposta parcial para um ambiente para geração de signos visuais (AGSV). A apresentação contempla apenas as telas principais do ambiente proposto, sem a preocupação de ser uma especificação definitiva. O objetivo é demonstrar ao leitor deste trabalho de que forma o AGSV reflete a base teórica sobre a qual foi proposto.

De acordo com a abordagem semiótica sobre o qual se assenta, o AGSV deve orientar o projetista em três aspectos: (a) a segmentação dos contínuos de conteúdo e da expressão, para usuário e sistema, (b) a especificação dos sistemas de códigos e (c) a associação entre os sistemas de códigos do conteúdo e da expressão. No que se refere à verificação de consistência, os códigos linguísticos do usuário e do sistema serão tratados de forma integrada. No processo de especificação da LV, frequentemente o projetista deverá retornar de uma etapa mais avançada, para uma etapa inicial. Por exemplo, em função do não determinismo do tipo de segmentação que pode ser executada, na etapa de geração de código o projetista pode ter necessidade de retornar e alterar a segmentação, no sentido de obter uma melhor codificação das correlações entre os tipos/intâncias do conteúdo e da expressão. Portanto, o AGSV deve permitir ao projetista alternar entre as etapas de segmentação e codificação, quantas vezes for necessário. Sempre que solicitado, poderão ser aplicadas as HALV sobre o código gerado.

Nas seções seguintes serão apresentados: um resumo das diretrizes teóricas e de especificação da linguagem sobre as quais o AGSV foi definido, a proposta para o AGSV propriamente dito e as heuristicas de avaliação da LV.

7.2. Resumo das Orientações de Base para o AGSV

Como foi dito, o AGSV reflete a base teórica estabelecida para especificação das LVs, assim como implementa as DOCLV. Portanto, antes de apresentarmos as principais telas projetadas para a implementação dessa ferramenta, vamos relembrar resumidamente os aspectos mais importantes relativos a estes dois pontos. A base da definição de uma boa linguagem é o léxico. A simples utilização da STAG auxilia o processo de lexicalização através: da segmentação dos contínuos semióticos, da distinção entre planos da expressão e do conteúdo, e da correlação destes planos a nível de tipo e instância. Não existem regras de orientação para que o projetista possa executar a segmentação dos contínuos de maneira determinística. No entanto, a definição da tipologia de conteúdo deve estar assentada em algum critério coerente de segmentação, de forma que a estruturação das sentenças através da gramática tenha algum sentido cognitivo para o usuário. Lembramos que os recursos expressivos disponíveis são função da capacidade dos dispositivos de comunicação (por exemplo, mouse de dois botões e vídeo colorido).

Como foi visto (Capítulo 3), a LV é pobre em articulação. Portanto, o sistema de códigos será bastante simples, no sentido de que em geral não serão produzidos tipos mais complexos a partir dos tipos básicos. Em função disto, na LV o mapeamento é feito praticamente de um para um entre os tipos do conteúdo e os tipos expressivos. Os recursos visuais expressivos são limitados e a codificação deve ser cuidadosa. Na especificação do código visual do usuário, o projetista deve observar as metáforas atuantes (Seção 4.2.2 e 4.2.3). A ruptura das metáforas com codificações ocasionais tende a aumentar a carga cognitiva imposta ao usuário. O projetista deve respeitar as DOCLV. entre elas estão: associar os recursos expressivos considerando as características do conteúdo; assinalar cuidadosamente as relações de sobreposição; respeitar os níveis de dependência na utilização do código visual; observar limites para a cardinalidade do conjunto de conteúdos a serem associados a um determinado recurso expressivo e outras. Parte desses aspectos poderão ser verificados através das HALV (Seção 7.4).

Pelas limitações do código visual, geralmente não é possível representar toda informação necessária através de recursos visuais. Nestes casos, o projetista deve priorizar o que será representado através de cada código, lançando mão de um código complementar (por exemplo textual) para o restante. A notação registra o uso de código complementar, mas não o controla automaticamente, o controle deverá ser feito pelo projetista.

Através da STAG o projetista poderá contextualizar o código visual. A estrutura hierárquica da notação permite a contextualização do código visual em diferentes níveis de granularidade. Os níveis mais altos da hierarquia correspondem aos CPDs, que atendem a diferentes condições de interpretação da representação visual. Ou seja, o usuário poderá interagir com os SMG alternando entre diferentes interpretações sobre a mesma representação

visual. Isto vai exigir códigos visuais adequados a cada modelo, porém consistentes entre si. Não pode haver incompatibilidade no conjunto de traços semânticos de conteúdos expressos de forma semelhante. Entre contextos paralelos na hierarquia a utilização dos recursos visuais deve ser consistente.
Os níveis mais baixos na hierarquia correspondem às UTDs, que atendem a contextos específicos relacionados a unidades tópicas de discurso. Nestas unidades não necessariamente haverá mudanças substanciais no código visual, porém conteúdos de interesse específico poderão ser expressos, sobrepondo-se a outros desnecessários no contexto. As UTDs permitem a expressão de conteúdos de interesse específico da tarefa, sem prejuízo da linguagem como um todo.

7.3. Proposta Parcial para o Ambiente para Geração de Signos Visuais

O AGSV foi projetado com base em uma estrutura hierárquica de janelas, objetivando individualizar as etapas de segmentação e mapeamento, para usuário e sistema. A Figura 7.1 (final da seção) oferece uma tela inicial, através da qual podem ser acessadas janelas para tratamento de cada uma das etapas de especificação do código visual. As três primeiras janelas (barra inferior da tela) são utilizadas para especificação do léxico do usuário. Através da primeira delas é especificado o sistema de códigos do conteúdo do usuário, através da segunda é especificado o sistema de códigos expressivos e através da terceira é feito o mapeamento, correlacionando tipos/instâncias do conteúdo e da expressão do usuário. A especificação do sistema de códigos do conteúdo e do sistema de códigos expressivos compreende a segmentação dos contínuos do conteúdo e da expressão, respectivamente, seguida da estruturação dos tipos discretos. Considerando que o sistema de códigos do conteúdo do usuário é definido em função de suas intenções, as quais são formuladas sobre as interfaces, e que o sistema de códigos expressivos do usuário é definido em função do dispositivo, geralmente um mouse de dois botões, esses tipos se manterão constantes em diferentes LVs. Portanto, através do AGSV, eles poderão ser editados de especificações já existentes, no sentido de facilitar o seu trabalho de segmentação.
As três segundas janelas (também na barra inferior da tela) são utilizadas para especificação do léxico do sistema, num processo equivalente ao anterior. Neste caso, quanto ao sistema de códigos do conteúdo do sistema nada pode ser oferecido em termos de tipos discretos, uma vez que a segmentação é executada em função do domínio do sistema. O AGSV pode apenas oferecer uma lista de componentes a serem segmentados, considerando, por exemplo, o modelo ERA. Neste caso o AGSV indicará ao projetista que ele pode segmentar o contínuo de conteúdo em entidades, relacionamentos, atributos, variáveis de estado do sistema e outros. Mais uma vez, o projetista deverá ter liberdade de ignorar a oferta e executar a segmentação com base em outro modelo.
É fundamental esclarecer em que aspectos o AGSV impõe um comportamento ao projetista e em que aspectos ele apenas oferece auxílio. Quanto aos primeiros, a interação através de janelas para segmentação dos contínuos, janelas de mapeamento e janela de definição da gramática, o projetista será forçado a especificar a LV de acordo com a abordagem semiótica. Ele estará distinguindo conteúdo de recurso expressivo, para sistema e usuário. Quanto aos aspectos de auxílio, nos casos em que o contínuo a ser segmentado pode ser comum a várias linguagens (por exemplo um mouse), o AGSV deverá oferecer uma segmentação prévia, uma vez que os tipos discretos tenderão a ser constantes. Apesar dessa tendência, a oferta de uma segmentação prévia para qualquer contínuo é considerada um aspecto de auxílio, podendo o projetista sempre optar por uma segmentação diferente. Portanto, quaisquer ofertas que o AGSV fizer ao projetista, referentes à segmentação, devem ser passíveis de substituição.
O AGSV poderá oferecer segmentações de contínuos expressivos referentes a diferentes dispositivos de comunicação. A oferta pode auxiliar projetistas mais experientes, acelerando seu trabalho. A oferta também pode auxiliar projetistas menos experientes, exemplificando uma proposta de segmentação coerente com a abordagem teórica adotada. Porém, ela não deve forçar a solução do projetista sobre a LV em nenhum sentido. Ele deve ter liberdade total quanto aos critérios de especificação da LV.
Ao expandir uma das quatro janelas para segmentação dos contínuos o projetista terá um desdobramento de seu conteúdo em tipos e instâncias (Figura 7.2), de tal forma que ele sempre será forçado a especificar uns e outros. Ao expandir as janelas de mapeamento (Figura 7.3) ele terá um desdobramento de seu conteúdo em tipos e instâncias, semelhante à anterior. Porém, neste caso, ele não poderá ser forçado a fazer o mapeamento entre instâncias respeitando a classificação em tipos. Por exemplo, o projetista poderá expressar uma instância de propriedade através de cor e outra instância de propriedade através de textura. Embora, dentro da proposta geral deste trabalho, esta não seja uma decisão adequada, em situações particulares pode ser necessária. De qualquer forma o projetista sempre deverá ter direito a ela.
Ao especificar o léxico do usuário e do sistema o projetista deverá especificar o contexto paralelo de discurso (CPD). O primeiro contexto será assumido automaticamente pelo AGSV. A última janela especificará a estrutura hierárquica da linguagem. Para mudar de contexto o projetista poderá assinalar outro ponto da estrutura.
Finalmente, através da última janela na barra inferior da tela o projetista poderá especificar a gramática (Figura 7.4). Na gramática ele poderá especificar diferentes unidades tópicas de discurso (UTD). Através da numeração estas unidades serão assinaladas na estrutura hierárquica. A especificação da gramática não é obrigatória.

Customização do AGSV para Linguagens Visuais Particulares

Embora o AGSV não obrigue as soluções, as heurísticas estarão todo o tempo avisando ao projetista, caso a LV especificada esteja em desacordo com suas diretivas de orientação. As medidas de qualidade embutidas nas diretivas de orientação são medidas experimentais, adotadas em função de um usuário genérico, logo, não são medidas absolutas. Orientações particulares poderão ser adotadas pelos projetistas, de forma a gerar LVs personalizadas, características de determinados ambientes, determinados grupos ou determinados trabalhos.
Para atender a esses casos o AGSV deverá implementar as heurísticas com base em um sistema de ponderações, através do qual as diretrizes de orientação poderão ser alteradas. Por exemplo, serão atribuídos pesos às regras embutidas nas heurísticas, e os avisos ao projetista só serão deflagrados quando um determinado peso for atingido. Através da alteração dos pesos o projetista poderá customizar o AGSV de acordo com suas necessidades.

7.4. Heurísticas de Avaliação da Linguagem Visual

As heurísticas oferecem indícios de que certas irregularidades, ou soluções inadequadas podem estar ocorrendo. A heurística obviamente não é absoluta, sua indicação serve como um suporte ao projetista. As heurísticas aqui apresentadas foram escritas em linguagem natural, em forma procedimental. Todos os procedimentos especificados nas heurísticas, para execução sobre a declaração de tipos e instâncias, e sobre o mapeamento, devem ser executados para usuário e sistema.
Algumas verificações não podem ser feitas automaticamente, devem ser feitas por observação, pelo projetista. Mas, a notação oferece formas de especificação que facilitam o trabalho. Por exemplo, a STAG prevê formas de especificação explícita para a redundância de código e para interação do código visual com outros códigos. Outras verificações nem mesmo podem ser auxiliadas pela notação. Por exemplo, as DOCLV orientam no sentido de que o conteúdo deve ser associado ao recurso expressivo em função das características do mesmo. Por exemplo, um atributo dinâmico poderia ser melhor expresso através do brilho do que um atributo estático. O atendimento a este tipo de orientação depende exclusivamente do projetista.

7.4.1. Verificação de Sistematicidade na Codificação das Correlações

Esta heurística tem o propósito de averiguar se a codificação das correlações entre conteúdo e recurso expressivo está respeitando a segmentação estabelecida, em tipos e instãncias. São feitas as seguintes verificações. (a) É verificado se existe uma declaração básica de forma. (b) É verificado se as associações entre instâncias estão respeitando a associação entre os tipos a que pertencem. Por exemplo, duas instâncias de conteúdo de um mesmo tipo não podem ser associadas a instâncias de expressão de tipos diferentes. Por exemplo, uma instância de material expressa por cor e outra expressa por textura. (c) É verificado se as associações estão sendo feitas entre tipos ou entre instâncias, não devendo ser associados tipos a instâncias, ou vice-versa. Obviamente será necessário verificar se todos os tipos e instâncias foram declarados, e será conveniente verificar se existem tipos ou instâncias declarados sem utilização. A verificação deverá ser feita em todos os contextos definidos pela hierarquia da notação.
A heurística verifica também se as instâncias declaradas de forma encapsulada foram utilizados no mapeamento. O projetista deve ser alertado sobre isto.

Varrer o mapeamento verificando se existe alguma entidade associada a forma. Se sim,

> Varrer o mapeamento de tipos, verificando se todos os tipos mapeados (do conteúdo e da expressão) foram corretamente declarados, e assinalando na declaração os tipos mapeados. Se sim,
>
> > Varrer o mapeamento de instâncias, verificando se todas as intâncias (do conteúdo e da expressão) mapeadas foram corretamente declaradas, e assinalando na declaração as instâncias mapeadas. Se sim,
> >
> > > Varrer a declaração de tipos e de instâncias (do conteúdo e da expressão), verificando se algum tipo ou instâncias declarados não foi assinalado (não utilizado no mapeamento).
> > >
> > > Varrer o mapeamento de tipos e de instâncias verificando se todas as associações são feitas entre tipos ou entre instâncias. Se sim,
> > >
> > > > Varrer o mapeamento de tipos e de instâncias, verificando se todas as associações entre instâncias respeitam as associações entre tipos.

Varrer a declaração de tipos e instâncias expressivos, verificando as instâncias declaradas encapsuladas, varrer o mapeamento de tipos e instâncias verificando a ocorrência de utilização das instâncias encapsuladas.

Esta heurística de verificação é básica. Todas as outras verificações só deverão ser feitas caso esta tenha sido executada com sucesso.

7.4.2. Verificação de Abuso de Código

Esta heurística tem o propósito de averiguar se um mesmo recurso visual está sendo excessivamente utilizado, podendo confundir o usuário ou, até mesmo, descaracterizá-lo como recurso expressivo. Por exemplo, é inadequada a utilização do recurso cor para expressar diferentes materiais, diferentes propriedades ou ainda para colorir, simultaneamente. São feitas as seguintes averiguações: (a) Se o mesmo tipo de recurso expressivo (exceto forma) é utilizado para expressar diferentes tipos de conteúdos, num mesmo contexto. No caso da forma a avaliação deve ser feita a nível de instância, ou seja, se uma determinada forma está sendo associada a instâncias de conteúdo diferentes; (b) Se um determinado tipo de recurso expressivo é associado a um tipo de conteúdo, cujas instâncias formem um conjunto de cardinalidade muito alta. Neste caso é usado como referência o número mágico de Miller (56), sete mais ou menos dois.

A utilização de um mesmo recurso expressivo para dois tipos de conteúdo, em um mesmo contexto, não significa necessariamente um erro. Se o conjunto das intâncias de conteúdo, para os dois tipos, for bastante baixo, a utilização é possível, desde não cause sobreposição dos recursos expressivos e perda da informação. Por exemplo, a cor pode ser utilizada para expressar duas instâncias de materiais e duas instâncias de propriedades.

Na verificação de abuso de código devem ser considerados os contextos particulares especificados através das UTDs. A heurística de verificação de abuso de código deve ser aplicada a cada UTD. Em princípio, a reutilização de um recurso não deve ser considerada abuso de código se estiver sendo feita dentro de uma UTD. Mas, se um mesmo recurso estiver sendo reutilizado em várias UTDs o projetista deverá ser alertado sobre isto.

Varrer o mapeamento de tipos, verificando os tipos do conteúdo que não são expressos por forma. Para estes, varrer a declaração de instâncias do conteúdo, verificando a cardinalidade do conjunto de instâncias, para cada tipo do conteúdo. Se a utilização for adequada,

> Varrer o mapeamento de tipos, assinalando os tipos de recurso utilizados. Se a frequência de utilização de um mesmo recurso exceder um certo limite, verificar a cardinalidade dos conjuntos de instâncias do conteúdo, referentes aos tipos envolvidos.

Varrer o mapeamento de instâncias, para os tipos de conteúdo expressos por forma, assinalando as instâncias de forma utilizadas. Verificar se a frequência de utilização está em um limite adequado.

7.4.3. Verificação de Sobreposição de Código

A sobreposição de código é uma situação de difícil avaliação, porque depende de uma declaração específica na notação e porque está sujeita a questões de contexto e temporalidade. Quanto ao primeiro motivo, para que ela seja bem avaliada é necessário que o projetista tenha declarado corretamente todas as relações de atribuição e sobreposição. Não é difícil que o projetista se esqueça de declará-las, uma vez que elas não são visualmente expressas. Elas são declaradas apenas para efeito de conferência da sobreposição de código. Portanto, antes da aplicação desta heurística, pode ser conveniente que o aplicativo solicite ao projetista a confirmação de que essas relações foram declaradas.

Quanto ao segundo motivo, a heurística não tem todas as informações necessárias para avaliar as condições em que a sobreposição ocorre. Há casos em que a sobreposição não é simultânea. Por exemplo, a sobreposição de recursos expressivos dentro de uma UTD pode mascarar temporariamente uma ocorrência expressiva. A heurística deve avaliar cada UTD em separado, assinalando caso o mesmo recurso visual tenha sido excessivamente utilizado entre as UTDs. Em geral, a heurística executa uma avaliação "sintática" e não "semântica". No contexto geral da gramática, a heurística pode acusar a sobreposição de recursos visuais, porém sem nenhum critério de avaliação sobre a gravidade da ocorrência.

Para cada componente do lado esquerdo da declaração de relação, deve ser feita nova verificação nas declarações de relação, de modo a detectar sobreposições sucessivas. No exemplo abaixo, as representações de atributo, relacionamento e estado estarão se sobrepondo à representação da entidade. Quando o componente da esquerda não for especificado em nenhuma outra relação, ele deverá estar sendo mapeado no tipo forma. Isto porque todas as relações de sobreposição devem ocorrer sobre a forma. No exemplo abaixo, as entidades são expressas através de forma.

entidade <-- atributo
atributo <-- relacionamento
entidade <-- estado

Quando o componente do lado esquerdo da declaração de relação for um tipo, exemplo abaixo, todas as instâncias correspondentes devem ser marcadas como sofrendo sobreposição.

primitivas <-- atributo

Para avaliação das sobreposições aceitáveis é necessário um controle sobre os níveis de dependência na utilização dos recursos visuais. Se todas as sobreposições utilizarem o mesmo recurso expressivo, por exemplo cor, pode se caracterizar uma situação de abuso de código, exceto para uma cardinalidade muito pequena (conforme heurística de verificação de abuso de código). Se diferentes recursos expressivos forem utilizados podem ocorrer situações como, por exemplo, uma cor aplicada sobre a textura para qualificá-la. Ou seja, por exemplo, uma propriedade expressa por textura e um atributo dessa propriedade expresso por cor, logo a cor atuará sobre a textura.

Varrer as declarações de relações de atribuição e de sobreposição, marcando as instâncias de conteúdo que recebem qualquer tipo de representação sobre si.

Verificar se as instâncias terminais que recebe sobreposição são mapeadas no recurso expressivo forma. Se sim,

> Varrer as instâncias de conteúdo marcadas, verificando se as sobreposições utilizam recursos expressivos diferentes.
>
> Se sim,
>
> > Verificar se a utilização do recursos expressivos está coerente com os níveis de dependência na utilização de recursos visuais.
>
> Senão,
>
> > Verificar abuso de código.

7.4.4. Verificação Inter-Códigos

A verificação inter-códigos (entre diferentes CPDs) deverá ser feita basicamente comparando-se os tipos/instância de conteúdo e os tipos/instâncias expressivos. Por exemplo, é possível comparar se as entidades declaradas em diferentes CPDs estão sendo expressas pelo mesmo recurso, por exemplo forma. É possível também comparar se as instâncias de sinalizadores estão sendo expressas por recurso semelhante. Por exemplo, o estado-seleção, uma instância de variável de estado das entidades, em um CPD pode estar sendo expresso por cor (vermelha), em outro por forma (pontos de manipulação), ou ainda, no segundo pode estar sendo expressa por cor (verde). A expressão do estado-seleção por cor e forma, em diferentes CPDs, pode dificultar a compreensão do usuário. Porém, a expressão por cores diferentes pode confundir totalmente o usuário, na medida em causa uma ruptura na capacidade do usuário inferir a codificação.
É importante salientar que esse tipo de conferência depende totalmente dos nomes utilizados pelo projetista para identificar os tipos e instâncias na notação. Se em um CPD um determinado estado for denominado estado-criação e em outro for denominado estado-execução, a heurística não terá como avaliar que são a mesma coisa.
CONFERIR A NUMERAÇÃO

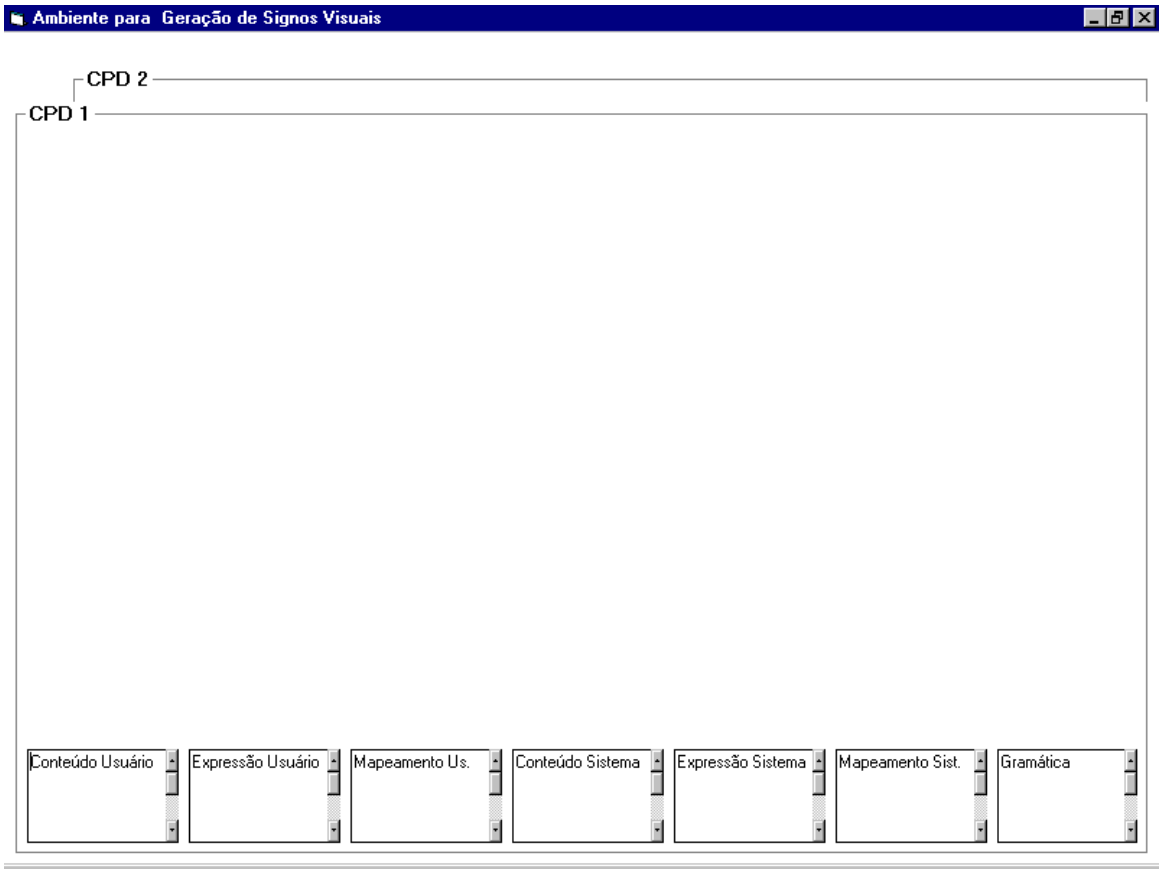


Figura 7.1. Tela de abertura do AGSV

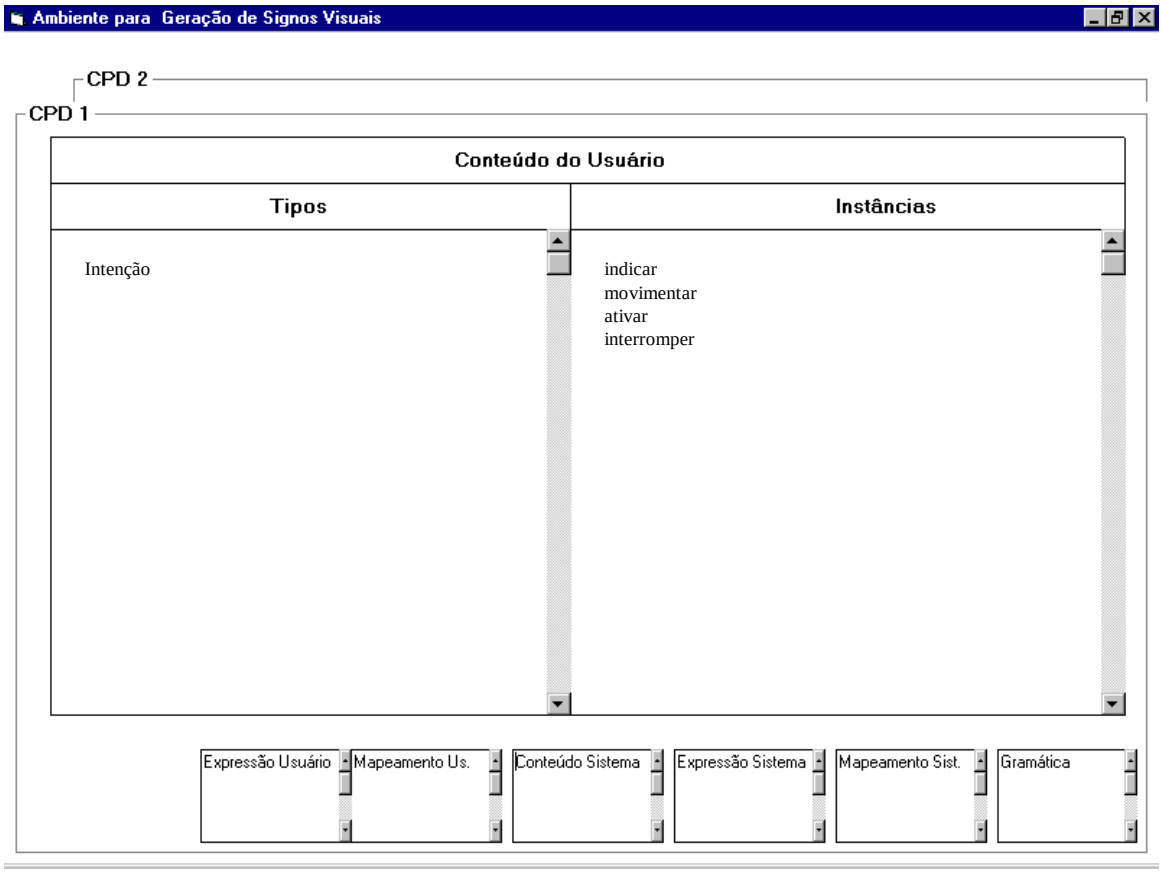


Figura 7.2. Tela para segmentação dos contínuos

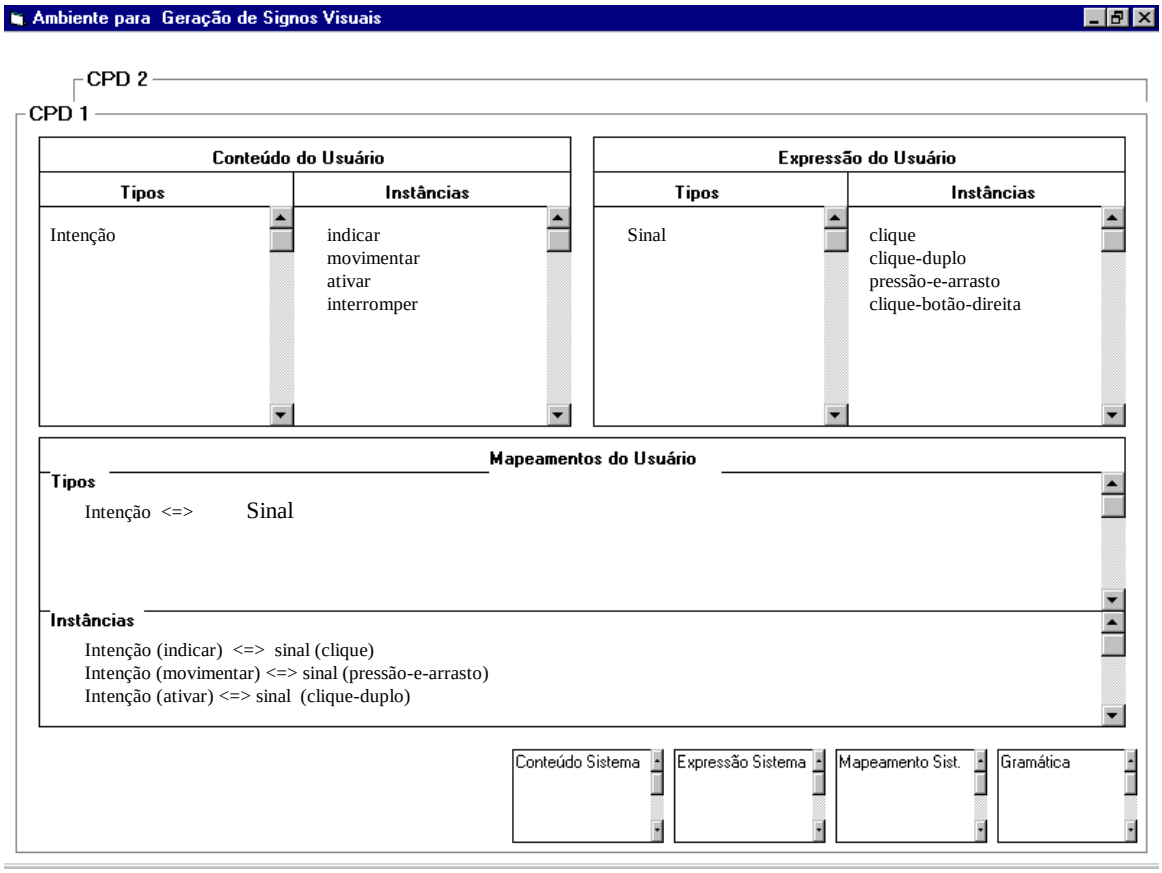


Figura 7.3. Tela para especificação do mapeamento

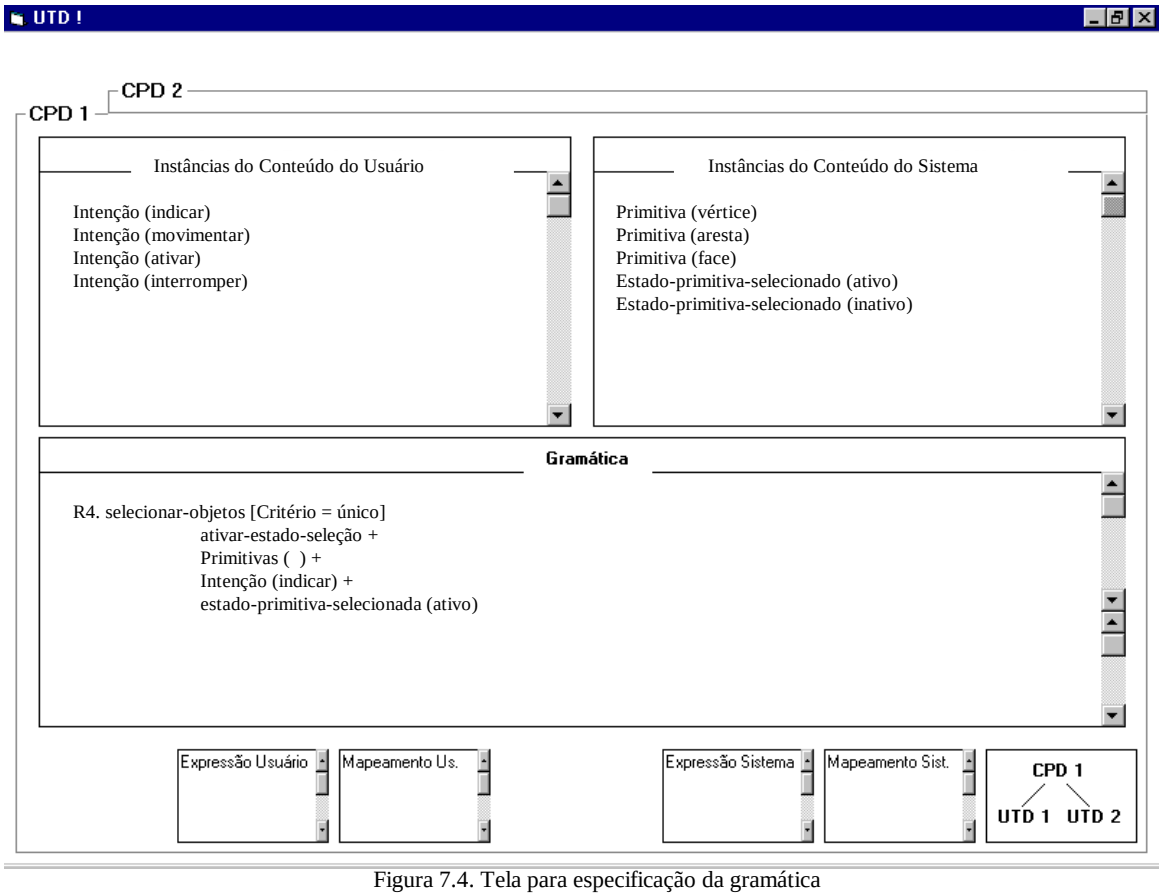


Figura 7.4. Tela para especificação da gramática

Os recursos de hardware disponíveis para interação usuário-sistema estão em constante evolução. A tendência é que as interfaces utilizem cada vez mais recursos gráficos, associados a outras mídias, em interfaces multimodais. Parte das atuais interfaces gráficas pode evoluir no sentido da realidade virtual. Qualquer que seja o caminho evolutivo para as atuais interfaces gráficas, os problemas básicos em termos de linguagem de comunicação serão semelhantes. Portanto, o tratamento sistemático dos recursos expressivos é fundamental para as interfaces atuais e para as futuras. Em qualquer sentido que essa evolução ocorra, uma plataforma metodológica básica será necessária. Qualquer passo no sentido de um tratamento sistemático das atuais interfaces gráficas, será a base de apoio para trabalhos de maior porte, em futuro próximo.

8.1. Sobre a Motivação para o Trabalho

A proposta deste trabalho surgiu da constatação, junto a equipes de desenvolvimento de software gráfico, da dificuldade dos projetistas em desenvolverem interfaces gráficas para SMG, de forma sistemática e controlada. De modo geral, as interfaces de sistemas gráficos são desenvolvidas por especialistas na área de computação gráfica, cujo conhecimento sobre linguagens de interface é relativamente superficial. Em decorrência, algumas vezes observamos, por exemplo, interfaces que deixam transparecer determinações e estruturas internas dos programas. Este procedimento não se restringe apenas à computação gráfica, ocorrendo também em outras áreas. Porém, na área de computação gráfica o problema se agrava, dadas as especificidades do recurso visual expressivo. O que se observa são interfaces construídas com base em soluções ocasionais, as quais podem acarretar ônus de projeto (como interfaces que não suportam expansões) ou sobrecargas cognitivas para o usuário (como este não conseguir inferir o comportamento do sistema).

Obviamente, esta situação não é verdade absoluta. Em equipes de grande porte, em grandes desenvolvedoras de software, encontram-se profissionais voltados exclusivamente para projetos de interfaces, dispondo de tempo e recursos adequados à execução desta tarefa. Porém, em equipes de trabalho de menor porte, observamos que estes profissionais carecem de ferramentas que os orientem na produção de projetos consistentes, com base em requisitos teóricos sólidos, mas sem um esforço adicional deles próprios.

Este trabalho foi desenvolvido com base nestas premissas, buscando dar mais um passo no sentido de superar a barreira de dificuldades relativas ao uso sistemático de LVs. O objetivo foi obter subsídios para uma ferramenta capaz de apoiar decisões de projeto, destinadas a possibilitar a interação usuário-sistema através de LVs por MD em SMG. As linguagens deverão atender as etapas de desenho e modelagem sucessivas do objeto gráfico. O trabalho contemplou aspectos teóricos e práticos. Quanto aos teóricos, ele apresenta: (a) um quadro analítico sobre interfaces gráficas por MD para SMG e (b) uma notação para sistematização de escolha expressivas, a Semiotic TAG (STAG).

O quadro teórico foi construído sobre dois eixos, um semiótico e outro cognitivo, de forma complementar (Capítulo 2). Nesse quadro a interface é observada sob duas óticas. Na primeira, ela é um artefato de troca de mensagens entre usuário e sistema. A engenharia cognitiva [Norman 86] esclarece o processo de contrução da mensagem usuário-sistema, através do golfo de comunicação. A relevância dessa proposta está na distinção entre aspectos mentais de formulação da solução e aspectos físicos de ativação do sistema. A segunda ótica de análise é a da engenharia semiótica [Souza 93b], que complementa a proposta da engenharia cognitiva. Ela trata a interface como um artefato de metacomunicação, distinguindo dois níveis de mensagens veiculadas. Nela, a interface é uma via unidirecional através da qual trafegam mensagens do projetista para o usuário, ao mesmo tempo em que ela própria emite e recebe mensagens no processo interativo com o usuário.

Com relação ao processo de produção de mensagens, as duas óticas foram mantidas. Através de uma abordagem semiótica [Eco 76] foi estudado o processo de produção sígnica, fundamental para a composição das mensagens. Através de uma abordagem cognitiva foram estudadas questões de percepção e compreensão [Jackendoff 83], bem como o uso de metáforas [Lakoff & Johnson 80]. Considerando o constante processo evolutivo a que as interfaces gráficas estão sujeitas, a base teórica foi adotada com o intuito de atender não somente as necessidades atuais, mas oferecer suporte a expansões futuras, para ambientes e interfaces mais complexas. O enfoque semiótico utilizado no tratamento dos recursos visuais, oferece suporte à expansão da interface para outras modalidades de recursos expressivos.

Quanto aos aspectos práticos, este trabalho apresenta: uma proposta para um ambiente de geração de signos visuais (AGSV), como uma ferramenta de auxílio ao projeto de interfaces. A proposta do AGSV foi direcionada especialmente aos projetistas da área de computação gráfica, que desenvolvem interfaces sem serem especialistas.

8.2. Sobre o Trabalho Desenvolvido

Os SMG, como se pode dizer de todos os sistemas computacionais [Eco 88], são sistemas essencialmente semióticos (criadores de signos), na medida em que representam objetos computacionais, sobre os quais os usuário interage. Como tal, eles devem apoiar os usuários na tarefa de (a) selecionar e instituir formas significativas (primárias e secundárias), (b) prover mecanismos sistemáticos para a interpretação de tais formas quando usadas em atos de comunicação/representação.

No conceito de usabilidade apresentado por Adler & Winograd (92), o usuário é um componente ativo, capaz de entender o sistema, aprender sobre ele e utilizá-lo de formas criativas. Nesta abordagem, o foco de preocupação está em "como o projetista pode expressar melhor ao usuário o que o sistema faz". O projeto de interfaces deve orientar-se na busca da melhor forma de expressar o modelo funcional do sistema, potencializando ao usuário utilizá-lo plenamente. De acordo com os princípios semióticos, quanto melhor o projetista puder expressar o modelo funcional, maiores chances existem de que os interpretantes dos usuários sejam bastante próximos do seu próprio interpretante. A consistência da linguagem de interface, enquanto um sistema semiótico, é fundamental no processo de comunicação.

O projeto de uma LV consistente exige do projetista o conhecimento do potencial e das limitações do recurso visual expressivo, no sentido de usá-lo adequadamente. A LV tem características próprias, exigindo um tratamento específico, diferenciado daquele adotado para a linguagem textual. Com base no postulado semiótico [Eco 76], foi averiguado o potencial expressivo dos recursos visuais em ambiente computacional (Capítulo 3). Nessa análise algumas particularidades foram apontadas para a LV, como:: escassez de recursos expressivos; baixo nível de articulação dos recursos expressivos; sobreposição de recursos expressivos sobre o recurso forma. Foi registrada a

presença de metáforas no processo de significação [Lakoff & Johnson 80]. Além disso foi apontada uma instabilidade da linguagem no processo de produção de signos visuais, uma vez que o projetista pode codificar a mensagem de diferentes formas, em função de seu próprio interpretante.
A partir da análise do recursos expressivos visuais em ambiente computacional, foi proposta a STAG (Capítulo 6). A STAG é uma extensão da TAG - Task-Action Grammars [Payne 86], uma gramática orientada à tarefa. O fato de a TAG ser uma metodologia já conhecida, facilita o uso da notação. A STAG busca corrigir alguns aspectos da TAG, fazendo uma adaptação para tratamento dos recursos visuais. Por exemplo, um dos problemas identificados para a TAG é a não distinção entre conteúdo e recurso expressivo. De acordo com o enfoque semiótico, o código linguístico é resultante de uma associação entre um sistema de códigos do conteúdo e um sistema de códigos expressivos. A STAG busca resolver esta questão (a) especificando a segmentação dos contínuos semióticos em tipos e instâncias, (b) criando um mapeamento que associa conteúdo a recurso expressivo, a nível de tipo e instância e (c) baseando sua gramática apenas no conteúdo veiculado entre usuário e sistema.
A notação incorpora os pressupostos teóricos, de tal forma que, através de sua simples utilização, o projetista estará trabalhando de acordo com os princípios semióticos. Ela exige do projetista a segmentação dos contínuos semióticos (conteúdo e recurso expressivo, do usuário e sistema), a estruturação dos sistemas de códigos a partir dos elementos discretos e, finalmente, o mapeamento (correlação entre conteúdo e recurso expressivo). O mapeamento define os eixos de lexicalização da linguagem. Portanto, ele auxilia a regularidade da linguagem, uma vez que o conteúdo só é associado ao recurso expressivo uma única vez.
Quanto à gramática da notação, semelhante à TAG, ela busca mapear as tarefas do usuário em ações do sistema. A gramática da STAG descreve o conteúdo informacional trocado entre usuário e sistema, em torno da tarefa. Embora essa estruturação em torno da tarefa seja semelhante à TAG, na STAG ela desempenha um papel mais importante [Martins et alii 94/95]. Ela tem um papel fundamental na superação de parte das deficiências da linguagem, geradas pelas limitações dos recursos expressivos. Com base na tarefa, a LV pode ser estruturada em segmentos hierárquicos, gerando os contextos paralelos de discurso (CPDs), e as unidades tópicas de discurso (UTDs). Em especial as UTDs permitem a reutilização dos recursos expressivos da linguagem, em contextos particularizados em torno da tarefa, melhorando seu aproveitamento. Os CPDs permitem diferentes formas de interação (diferentes modelos interpretativos) sobre a mesma representação visual. Uma vantagem adicional pode ser obtida da estrutura de linguagem gerada pelas CPDs e UTDs. A estrutura da gramática pode ser usada pelo projetista, como um indicativo para a estruturação de suas telas e menus, uma vez que a gramática gira em torno das tarefas.
Dois pontos importantes devem ser ressaltados sobre a notação. Um deles é que ela não obriga as soluções do projetista, apenas orienta. O outro é que ela atende ao projeto e análise de LVs. Quando usada no projeto de linguagens ela registra o interpretante do projetista, quando usada na análise de linguagens já existentes ela registra os interpretantes dos usuários.
Para permitir um uso automatizado da notação, foi proposto o ambiente para geração de signos visuais (AGSV) (Capítulo 7). Esse ambiente implementa a STAG, em conjunto com as diretrizes de orientação para codificação da linguagem visual (DOCLV) e as heurísticas de avaliação da linguagem visual (HALV). O AGSV permite que o projetista trabalhe fazendo refinamentos sucessivos da LV.

Pontos fortes referentes à abordagem teórica utilizada

Pela sua abordagem ampla, a teoria semiótica oferece uma série de vantagens. Uma delas é que a semiótica trata as mais diferentes disciplinas (por exemplo matemática, psicologia e outras) como um conjunto de sistemas de códigos de significação. Dessa forma, ela permite incorporar às regras de formação do signo, elementos de várias disciplinas. Entre as regras que o código provê para gerar o signo podem estar componentes cognitivos, psicológicos, sociais, linguísticos e outros. Este enfoque permite incorporar ao código linguístico a cultura estabelecida na área.
Outra vantagem é que a semiótica reúne dois níveis de significado tradicionalmente utilizados na linguagem, não distinguindo entre semântica e pragmática. Isto permite incorporar ao sistema de significação todas as particularidades de contextos que antes eram atribuídas à pragmática. Além disso, a semiótica enfoca o processo de comunicação como algo dinâmico, permitindo uma retro-alimentação do sistema de significação, na medida em que a cultura na área se reorganiza. Ela ainda oferece suporte ao tratamento de diferentes recursos expressivos, o que significa a possibilidade de interfaces multimodais. Este aspecto é importante considerando-se as restrições do recurso expressivo visual e as interfaces mais complexas, que virão no futuro.
Finalmente, através das noções de contínuos semióticos, a semiótica distingue claramente o conteúdo a ser expresso do recurso expressivo (tradução semântica e articulatória), e distingue a expressão do usuário da expressão do sistema. O processo de segmentação dos contínuos força uma categorização do conteúdo expressivo, que reflete os aspectos cognitivos relativos ao uso que está sendo feito da linguagem.

8.3. Vantagens Esperadas com a Utilização da STAG

A STAG permite um tratamento sistemáticos no projeto de LVs, a partir do qual espera-se obter algumas vantagens, como as relatadas a seguir.

Quanto a aspectos de sistematização do trabalho

Quanto aos aspectos de sistematização do trabalho, podem ocorrer ganhos individuais (a nível de projeto) e ganhos culturais (a nível da comunidade de projetistas de interfaces gráficas). Quanto aos individuais, a STAG força o projetista a uma sistematização, que, pelo simples uso, vai denunciar irregularidades e inconsistências na LV. Por exemplo, ao segmentar o contínuo expressivo e explicitar os elementos discretos, através da declaração de tipos e instâncias, o projetista se dará conta da sua real capacidade em termos de recursos visuais expressivos (que é pequena). Outro exemplo, ao especificar o conteúdo expresso em uma unidade dialógica da gramática, o projetista terá clara consciência sobre a informação que ele está prestando ao usuário em termos do modelo funcional do sistema. Lembramos que na especificação da gramática o projetista não se preocupa com forma de expressão, apenas com conteúdo.
Quanto aos ganhos culturais, a STAG abre a possibilidade de instauração de uma nova cultura entre os projetistas, a cultura de que é possível tratar a LV de forma sistemática. O tratamento sistemático faz com que o projetista se dê conta de que ao projetar uma interface está projetando uma linguagem, passando a usar os recursos visuais como recursos expressivos com características linguísticas. O AGSV é fundamental para isso, porque ele libera o projetista do esforço de aprender a notação. Ele auxilia a especificação, através das DOCLV, ao mesmo tempo que oferece um retorno da qualidade da linguagem especificada, através das HALV.
O uso da notação garante uma base de trabalho sólida para especificação de interfaces de maior porte, uma vez que a sistematização é necessária para todo tratamento mais complexo em ambiente computacional.

Quanto aos aspectos de usabilidade

Interfaces consistentes permitem um melhor desempenho nos aspectos de usabilidade, o que significa interfaces mais fáceis de entender, de aprender e mais produtivas. Um dos aspectos mais importantes nesse processo é a consistência entre o significado pretendido (pelo projetista) e o significado percebido (pelo usuário). Essa consistência é potencializada quando o projetista, através da LV, expressa adequadamente o modelo funcional do sistema. Na medida em que o usuário identifica o interpretante do projetista, ele compreende o funcionamento do sistema. Como foi dito no Capítulo 1, o usuário formula a intenção de operação a partir de sua percepção sobre a interface. A consistência da percepção visual do usuário é atingida quando ele pode manipular "intuitivamente" a representação visual. Se ele tenta e não consegue executar a operação formulada, ocorre uma quebra na consistência da sua percepção visual. Porém, se o projetista puder expressar adequadamente o modelo funcional do sistema, o interpretante do usuário tenderá a ser bastante próximo do interpretante do projetista e suas intenções de operações serão coerentes com o sistema, não ocorrendo conflitos.
Como também foi visto, a regularidade da LV é um aspecto importante na expressão do modelo funcional. Esta regularidade é propiciada pela notação.

Quanto aos Aspectos de Projeto

Embora a STAG não obrigue as soluções do projetista (ela apenas o orienta), ela reflete a qualidade do sistema de comunicação que ele está criando para dialogar com o usuário. A notação permite que a medida de qualidade da interface venha à tona, durante o projeto, sem um esforço extra do projetista. Ao abordar os recursos visuais como recursos expressivos, o projetista pode perceber melhor seu potencial e suas limitações. Interfaces projetadas com conhecimento dos problemas inerentes à LV, permitem um melhor aproveitamento dos recursos e uma melhor integração com recursos de outra natureza.
O uso da notação também permite projetos de interfaces mais sólidos, não apenas para o consumo imediato, uma vez que o projetista consegue prever os problemas e projetar soluções mais seguras e mais abrangentes. Além do que, interfaces projetadas de forma sistemática, oferecem suporte a expansões futuras, o que não se verifica em projetos com base em soluções ocasionais. É sabido que grande parte dos custos de projetos deve-se a soluções inadequadas, que exigem grandes manutenções, e principalmente a soluções que não suportam alterações ou expansões. Portanto, com base nas diretrizes de orientação, o projetista pode trabalhar de forma mais segura, com a possibilidade de um resultado melhor e projetos mais baratos.
A notação permite também uma avaliação sistemática de interfaces já implementadas, viabilizando correções controladas, onde o efeito das alterações pode ser testado e avaliado. Durante a análise de uma interface implementada pode ser detectado um interpretante do usuário, como por exemplo um uso alternativo da interface, não previsto pelo projetista. Neste caso pode ser averiguado o que está levando o usuário a esse interpretante, uma necessidade não prevista em fase de projeto ou uma má comunicação do modelo funcional do sistema. Alterações podem ser feitas, testadas e avaliadas.

Quanto aos Aspectos Tecno-Culturais da Comunidade de Usuários de SMG
O uso da STAG potencializa a geração de uma nova cultura entre os projetistas, através um trabalho sistemático, evitando a instauração de soluções não adequadas, que são difundidas no mercado comercial e se perpetuam, transformando-se em cultura na área de aplicativos gráficos.

8.4. Trabalhos Futuros

Em termos de objetivos imediatos, a proposta é (a) a implementação do AGSV e (b) a avaliação de seu desempenho em ambientes de desenvolvimento de projetos e na avaliação de interfaces já implementadas. Quanto ao primeiro objetivo, a implementação do AGSV exigirá um grande esforço de programação, especialmente no sentido de criar estruturas de dados ágeis e potentes. A especificação da LV vai exigir um tratamento de verificação dos itens lexicais para os tipos e instâncias definidos, os quais serão identificados pelos nomes. As heurísticas de avaliação tratarão os tipos discretos de conteúdo e expressão em termos de seus nomes. Extensas verificações deverão ser executadas, em toda a especificação, a cada alteração do usuário.

Quanto à avaliação de desempenho, uma vez implementado, o AGSV deve ser utilizado por equipes de desenvolvimento de software gráficos, em tarefas de projeto e análise de interfaces com desempenho problemático. A avaliação do AGSV deve ser feita em dois sentidos: (a) sua eficiência em termos de projeto de linguagens e (b) sua agilidade como uma ferramenta de suporte aos projetistas. O uso do AGSV dará dois retornos importantes à proposta da notação: em que medida a qualidade da LV projetada justifica o uso da notação, e em que medida os projetistas sentem-se "à vontade" com o uso da ferramenta. Para que o segundo ponto possa ser avaliado (o aspecto do projetista) é necessário que os mesmos percebam uma alteração de qualidade real e positiva, nos projetos de LVs desenvolvidos com base na ferramenta. Porém, mesmo sendo constatado um acréscimo de qualidade no projeto da LV, se a ferramenta não for de fácil interatividade, seu uso será oneroso e, por consequência, ineficiente.

Utilizando-se a STAG em tarefas de análise pode-se obter resultados mais palpáveis em termos de eficiência da ferramenta, por serem resultados comparativos. Pode-se avaliar o desempenho das interfaces antes e depois de correções executadas com base na análise desenvolvida.

A utilização do AGSV em ambientes de projeto reais vai gerar um retorno, que permitirá ajustes na proposta atual da ferramenta. Eventuais ajustes na ferramenta implementada devem respeitar a proposta inicial: oferecer suporte aos projetistas para desenvolvimento das interface, oferecendo orientação quanto à qualidade do produto gerado, ser de fácil interatividade (ágil para uso), de forma que sua utilização não represente um ônus e sim um ganho.

Uma vez obtidas as validações necessárias para esta proposta inicial da STAG, e executados os ajustes necessários, ela deverá ser expandida para o tratamento de ambientes mais complexos envolvendo as mesmas mídias (interfaces multimodais) e equipamentos mais complexos. Alterações da proposta inicial da notação, para tratamento de equipamentos mais complexos, como por exemplo um mouse em 3D, em princípio não causarão nenhuma alteração estrutural, apenas um aumento da complexidade da especificação. Todo o processo de segmentação dos contínuos semióticos e organização em sistemas de códigos permanecerá de forma bastante similar.

Outro aspecto a ser avaliado é a expansão da ferramenta para outras mídias (interfaces multimídia), para associação de outras mídias ao recurso gráfico. Esta possibilidade irá exigir um estudo aprofundado em termos do tratamento semiótico à matéria linguística de outras mídias.

REFERÊNCIAS BIBLIOGRÁFICAS

[Abelson & Sussman 85] Abelson, Harold & Sussman, Gerald Jay Structure and Interpretation of Computer Programs. The Massachusetts Institute of Technology, 1985.

[Adler & Winograd 92] Adler, Paul S. & Winograd, Terry A. The Usability Challenge, In Usability: Turning Technologies into Tools, Edited by Paul Adler & Terry Winograd, Oxford University Press, 1992.

[Andersen 90] Andersen, P. B. A Theory of Computer Semiotics. Cambridge University Press. Cambridge, 1990.

[Andersen 93] Andersen, P. B. A Semiotic Approach to Programming. In The computer as Medium, Cambridge, University Press, 1993.

[Austin 62] Austin, John L. How to Do Things with Words. Harvard University Press, Cambridge, Massachussetts, 1962.

[AutoCAD 96] AutoCAD Release 13, Autodesk, Inc, 1996.

[Batini et alii 92] Batini, Carlo & Ceri, Stefano & Navache, Shamkant B. Conceptual Database Design - An Entity-Relationship Approach. The Benjamin/Cunnings Publishing Company, Inc, 1992.

[Berry 97] Berry, Dan Palestra sobre Métodos Formais no Desenvolvimento de Software. Departamento de Informática, PUC-Rio, Outubro/1997.

[Booth 89] Booth, Paul A. An Introduction to Human-Computer Interaction. Lawrence Erlbaum Associates, Publishers, 1989.

[Brennan 90] Brennan, Susan E. Conversation as Direct Manipulation: An Iconoclastic View, In Brenda Laurel, The Art of Human-Computer Interface Design, Addison-Wesley Publishing Company, 1990.

[Card et alii 83] Card, S. & Moran T. & Newell, A. The Psychology of Human Computer Interaction. Lawrence Erlbaum Associates, New Jersey, 1983.

[Carroll et alii 88] Carroll, John M. & Mack, Robert L. & Kellogg, Wendy A. Interface Metaphors and User Interface Design, in Handbook of Human-Computer Interaction, M. Helander (eds.), North-Holland, 1988.

[Carroll 91] Carroll, John M. Designing Interactions - Psychology at the Human-Computer Interface, Cambridge University Press, 1991.

[Chen 76] Chen, P.P The Entity-Relationship Model: Toward a Unified View of Data. In ACM Transactions on Database System 1, no. 1, March 1976, 9-37.

[Cohen 91] Cohen, P. R. The Role of Natural Language in a Multimedia Interface. Technote 514, SRI International, Menlo Park, 1991.

[Dondis 73] Dondis, Donis A. (1973) La Sintaxis de La Imagem - Introducion al Alfabeto Visual. Tradução em Espanhol Editora Gustavo Gili, Barcelona, 1976.

[Eco 76] Eco, Umberto Tratado Geral de Semiótica. Editora Perspectiva, 2a. edição, 1976.

[Eco 88] On Truth: A Fiction. In Meaning and Mental Representations, 41-60. Eco, U. & Santambrogio, M. & Violi, P. Editors. Indiana University Press, 1988.

[Erickson 90] Erickson, Thomas D. Working with Interface Metaphors, In Brenda Laurel, The Art of Human-Computer Interface Design, Addison-Wesley Publishing Company, 1990.

[Franzke 95] Franzke, Marita Turning Research into Practice: Characteristics of Display-Based Interaction, in Proceedings of CHI'95, May 7-11, 1995.

[Foley et alii 93] Foley, James D. & van Dam, Andries & Feiner, Steven K. & Hughes, John F. Computer Graphics Principles and Practice. Addison-Wesley Publishing Company, Second Edition, 1993.

[Gallagher 75] Gallagher, R. H. Finite Element Analysis - Fundamentals. Prentice Hall, Inc, 1975.

[García 95] García, Laura Sánchez LINX: Um Ambiente Integrado de Interface para Sistemas de Informação Baseados em Conhecimento, Tese de Doutorado, Departamento de Informática, Puc-Rio, Maio/1995.

[George 91] George, P.L. Automatic Mesh Generation - Application to Finite Element Methods. John Wiley & Sons, 1995.

[Grudin 89] Grudin, Jonathan The Case Against User Interface Consistency. in Communication of the ACM, October 1989, volume 32, number 10, 1164-1173.

[Hallaz & Moran 82] Hallaz, Frank & Moran, Thomas P. Analogy Considered Harmful. in Proceedings of the Conference on uman Factors in Computer Systems. Gaithersburg, 15-17 March, 1982.

[Harel 95] Harel, David On Visual Formalisms. In Diagrammatic Reasoning - Cognitive and Computacional Perspectives, Chandrasekaran, B. & Glasgow, J. & Narayanan, N. H. editors, MIT Press, 1995.

[Hartson & Hix 92] Hartson, H. Rex & Hix, Deborah Advances in Human-Computer Interaction, Volume 3. Ablex Publishing Corporation, Norwood, New Jersey, 1992.

[Helander 88] Helander, Martin Handbook of Human-Computer Interaction. North-Holland, 1988.

[Howes & Payne 90] Howes, A. & Payne, S. J. Display-Based Competence: Towards User Models for Menu-Driven Interfaces, in International Journal of Man-Machine Studies, Vol. 33(6) 637-655, 1990.

[Hutchins et alii 86] Hutchins, E. L. & Hollan, J. D. & Norman, D. A. Direct Manipulation Interfaces. In Norman & Draper, User Centered System Design. Hillsdale, NJ. Lawrence Erlbaum Associates, 1986.

[Jackendoff 83] Jackendoff, R. Semantic and Cognition. MIT Press, 1983.

[Jackendoff 87] Jackendoff, R. Consciousness and the Computational Mind. MIT Press, 1987.

[Jackendoff 91] Jackendoff, R. Semantic Structure. MIT Press, 1991.

[Kammersgaard 88] Kammersgaard, J. Four Different Perspectives on Human-Computer Interaction. In International Journal of Man-Machine Studies, no. 28, pp 343-362, 1988.

[Lakoff & Johnson 80] Lakoff, G. & Johnson, M. Metaphors We Live By. The University of Chicago Press, 1980.
[Lee 93] Lee, Geoff Object-Oriented GUI Application Development. PTR Prentice -Hall, Inc., Englewood Cliffs, New Jersey, 1993.
[Lohse 93] Lohse, Gerald Lee A Cognitive Model for Understanding Graphical Perception, in Human-Computer Interaction, Volume 8, p.353-388, 1993.
[Lyons 77] Lyons, John Semantics, Volume 1, Cambridge University Press, 1977.
[Marcus 92] Marcus, A. Graphic Design for Eletronic Documents and User Interfaces. New York, NY, ACM Press, 1992.
[Marr 82] Marr, David Vision - A Computational Investigation into the Human Representation and Processing of Visual Information. W. H. Freeman and Company, New York, 1982.
[Martins 90] Martins, I. H. O Perfil da Comunicação Homem-Máquina em Linguagem Natural para Sistemas com Base em Conhecimento. Dissertação de Mestrado, Departamento de Informática, PUC-Rio, 1990.
[Martins et alii 94] Martins, I. H. & Souza, C. S. de & Gattass, M. Orientação Semântica *Versus* Orientação Sintática na Captura de Dados para Aplicações Computacionais pelo Método de Elementos Finitos. Monografias em Ciência da Computação no. 31/94, PUC-Rio, Departamento de Informática, 1994.
[Martins et alii 95] Martins, I. H. & Souza, C. S. de & Gattass, M. Analytical Frameworks and Enhanced Modeling for Finite-Element Pre-Processors with Direct Manipulation Interfaces, Anais do XXII Semish - Seminário Integrado de Software e Hardware, 31/jul a 4/ago, Canela, RS, 1995.
[Maybury 93] Maybury, M. T. Intelligente Multimedia Interfaces. AAAI Press / The MIT Press, Menlo Park, California, 1993.
[MGE 96] MGE - Modular Geographic Information System (GIS) Environment - Versão 6.0.2, Intergraph Corporation, 1996.
[Microsoft Corporation 95] The Windows Interface Guidelines for Software Design - Microsoft Press, 1995.
[Miller 56] Miller, G.A. The magic mumber seven plus or minus two: some limits of our capacity for information processing, Psychological Review, 63(2), 81-87, 1956.
[Moll-Carrillo et alii 95] Moll-Carrillo, H.J. & Salomon, Gitta & Marsh, Mattew & Suri, Fulton Jane & Spreenberg, Peter Articulating a Metaphor Through User-Centered Design, in Proceedings of CHI'95, May 7-11, 1995.
[MTool 92] MTool - Bidimensional Mesh-Tool - Versão 1.0 - Manual do Usuário. Grupo de Tecnologia em Computação Gráfica - TeCGraf/PUC-Rio, Grupo de Geotecnia do SEDEM/DIPREX, Convênio CENPES-PETROBRÁS/PUC-Rio, Maio de 1992.
[MTool 97] Mtool - Bidimensional Mesh Tool - Versão 1.3 - Manual do Usuário. Grupo de Tecnologia em Computação Gráfica - TeCGraf/PUC-Rio, Grupo de Geotecnia do SEDEM/DIPREX, Convênio CENPES-PETROBRÁS/PUC-Rio, Maio de 1997.
[Mullet & Sano 95] Mullet, Kevin & Sano, Darrell Designing Visual Interfaces. Sun Microsystems, Inc, 1995.
[Nadin 88] Nadin, Mihai Interface Design and Evaluation - Semiotic Implications, In Advances in Human-Computer Interaction, Volume 2, Edited by H. Rex Hartson & Debora Hix, Ablex Publishing Corporation, Norwood, New Jersey, 1988.
[Norman 86] Norman, D. A. Cognitive Engineering. In Norman & Draper, User Centered System Design. Hillsdale, NJ. Lawrence Erlbaum Associates, 1986.
[Norman & Draper 86] Norman, D. A. & Draper, S.W. User Centered System Design. Hillsdale, NJ. Lawrence Erlbaum Associates, 1986.
[Norman 88] Norman, D. A. The Psychology of Everyday Things, New York, Basic Books, 1988.
[Norman 91] Norman, K. L. The Psychology of Menu Selection: Designing of Cognitive Control of the Human-Computer Interface. Ablex Publishing Corporation, Norwood, New Jersey, 1991.
[Ostrower 90] Ostrower, Fayga Acasos e Criação Artística. Editora Campus, 1990.
[Payne & Green 86] Payne, Stephen J. & Green T.R.G. Task-Action Grammars: A Model of the Mental Representation of Task Languages. in *Human-computer Interaction*, Volume 2, pp 93-133, 1986.
[Payne 91] Payne, Stephen J. Interface Problems and Interface Resources. In Designing Interactions, Edited by John Carrol, Cambridge University Press, 1991.
[Peirce 31] Peirce, Charles Sanders Collected Papers. Cambridge: Harvard University Press, 1931. Tradução Brasileira. Semiótica, São Paulo, Perspectiva, 1995.
[Piaget 42] Piaget, Jean Classes, Relations et Nombrés - Essai sur les Groupements de la Logistique et sur la Reversibilité de la Penseé, Librairie Philosophique J. Vrin, 1942.
[Piaget & Inhelder 59] Piaget, Jean & Inhelder, Bärbel Gênese das Estruturas Lógicas Elementares, Zahar Editores, 1959.
[Piaget & Chomsky 79] Piaget, Jean & Chomsky, Noam Théories du Langage - Théories de l'Apprentissage, Éditions du Seuil, 1979.
[Pimenta & Faust 97] Pimenta, Marcelo Soares & Faust, Richard Eliciting Interactive Systems Requirements in a Language-Centred User-Designer Collaboration - A Semiotic Approach. SIGCHI Bulletin, Volume 29, Number 1, 61-65, January 1997.
[Preece et alii 94] Preece, Jenny & Rogers, Yvonne & Sharp, Helen & Benyon, David & Holland, Simon & Carey, Tom Human-Computer Interaction, Addison-Wesley, 1994.

[Rheinfrank 96] Rheinfrank, John & Evenson, Shellet Design Languages. In Bringing Design to Software, Terry Winograd, ACM Press, New York, 1996.
[Rogers 95] Rogers, Erika A Cognitive Theory of Visual Interaction. In Diagrammatic Reasoning - Cognitive and Computacional Perspectives, Chandrasekaran, B. & Glasgow, J. & Narayanan, N. H. editors, MIT Press, 1995.
[Saint-Martin 90] Saint-Martin, F. Semiotics of Visual Language. Indiana University Press, 1990.
[Shneiderman 82] Shneiderman, Ben The Future of Interactive Systems and the Emergence of Direct Manipulation. in Behavior and Information Technology. Vol. 1 (1982), 237-56.
[Shneiderman 83] Shneiderman, Ben Direct Manipulation: A Step Beyond Programming Languages. in IEEE, August/1983, 57-69.
[Searle 69] Searle, John R. Speech Acts - An Essay in the Philosophy of Language. Cambridge University Press, 1969.
[Searle 75a] Searle, John R. Indirect Speech Acts. In: Sintax and Semantics: Speech Acts. Vol. 3, Cole, P. & Morgan, J.L. (eds), Academic Press, New York, 1975.
[Searle 75b] Searle, John R. a Taxonomy of Illocutionary Acts. In: Expression and Meaning. Cambridge University Press, 1975.
[Sebeok 95] Sebeok, T. Signs: An Introduction to Semiotics. Toronto University Press, Toronto, 1995.
[Souza 92] Souza, Clarisse S. de A Semiotic Approach to User Interface Language Design. CSLI - Stanford University. Report No. CSLI-92-166, PCD-6, 1992.
[Souza 93a] Souza, Clarisse S. de Lingüística Computacional Interativa para a Interface entre Usuários e Sistemas. In Proceedings do Encontro de Processamento de Língua Portuguesa, INESC - APL, Lisboa 25-26 de fevereiro de 1993, pp 155-122.
[Souza 93b] Souza, Clarisse S. de The Semiotic Engineering of User Interface Languages. In International Journal of Man Machine Studies (1993) 39, 753-773.
[Souza 94] Souza, Clarisse S. de Structure and Expressiveness of Visual Computer Languages. Monografias en Ciência da Computação no. 06/94, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, 1994.
[Souza et alii 96] Souza, Clarisse S. de & Prates, Raquel O. & Varejão, Flávio M. On Classes and Cost of Multimodal Interfaces, Monografias em Ciência da Computação, 12/96, Departamento de Informática, Puc-Rio, 1996.
[Stefik 95] Stefik, Mark Introduction to Knowledge Systems. Morgan Kaufmann Publishers, Inc, 1995.
[Stenning & Inder 95] Stenning, Keith & Inder, Robert Applying Semantic Concepts to Analyzing Media and Modalities. In Diagrammatic Reasoning - Cognitive and Computacional Perspectives, Chandrasekaran, B. & Glasgow, J. & Narayanan, N. H. editors, MIT Press, 1995.
[Stenning & Oberlander 95] Stenning, Keith & Oberlander, Jon A Cognitive Theory of Graphical and Linguistic Reasoning: Logic and Implementation. in Cognitive Science 19, 97-140, 1995.
[Sutcliffe 97] Sutcliffe, Alistair Task-Related Information Analysis, in International Journal of Human-Computer Studies, (1997) 47(2) 223-257.
[Svanaes 97] Svanaes, Dag Kinaesthetic Thinking: The Tacit Dimension of Interaction Design. In Computers in Human Behavior, Vol 13, No. 4, pp 443-463, 1997.
[Ware 93] Ware, Colin The Foundations of Experimental Semiotics: a Theory of Sensory and Conventional Representation. In Journal of Visual Languages and Computing (1993) 4, 91-100.
[Winograd & Flores 86] Winograd, T. & Flores, F. Understanding Computer and Cognition: a New Foundation for Design. Ablex Publishing Corporation, Norwood, New Jersey, 1986.
[Winograd 96] Winograd, T. Bringing Design to Software. ACM Press, New York, 1996.
[Word 95] Word - Microsoft Word for Window 95 - Versão 7.0, Microsoft Corporation, 1995.
[Ziegler & Fähnrich 88] Ziegler, J.E. & Fähnrich, K.P. Direct Manipulation, in Handbook of Human-Computer Interaction, M. Helander (eds.), North-Holland, 1988.

Anexo A

Relatório de Estudo de Caso

Este documento é um relatório de observação de um conjunto de operações em quatro aplicativos com interfaces gráficas por MD. Foram selecionadas as seguinte operações: criar, apagar, mover e alterar forma. Os aplicativos avaliados foram: Microsoft Draw [Word 95], AutoCAD [AutoCAD 96], MGE [MGE 96] e Mtool - Bidimensional Mesh Tool [Mtool 97].
O relatório está organizado por operação, cada uma delas relatada nos quatro aplicativos. Em cada operação foram observados: os pré-requisitos, os estados do sistema e a especificação do objeto da operação. Para os dados das operações foi observado seu comportamento, através da expressão de seus atributos e relacionamentos. Em toda a análise buscou-se a distinção entre o conteúdo do sistema e o recurso expressivo utilizado. Ao final de cada operação foi apresentada uma tabulação dos dados. A tabulação de dados foi apresentada de modo a registrar a alternância de expressão entre os agente da comunicação, na unidade de discurso referente à tarefa.

A.1. Operações de Criação

A seguir está relacionado o comportamento de cada um dos aplicativos no que se refere à criação de primitivas gráficas.

Criação de Objeto no MsDraw

- A Figura A.1 ilustra as primitivas de interação do MsDraw, no menu na parte baixa da tela. Nas versões anteriores do MsDraw a entrada de dados para a criação de primitivas era implementada como um estado do sistema (à exceção da polilinha e desenho livre), uma vez ativado o estado de criação várias instâncias daquela primitivas poderiam ser desenhadas, até que o estado fosse explicitamente desativado. Na versão 7.0 a indicação da primitiva a ser traçada deve ser feita uma a uma, através do menu. O botão em reverso no menu indica que o sistema está aguardando a entrada de dados para aquela primitiva. O desenho das primitiva é executado por movimentos de pressão-e-arrasto do mouse, exceto para a polilinha. O desenho livre e a polilinha são traçados a partir da mesma primitiva de interação. Se for usado pressão-e-arrasto o sistema traçará em desenho livre, se forem usados cliques simples sucessivos o sistema traçará uma polilinha. O usuário pode misturar desenho livre e segmentos de retas, apenas alternando os sinais de ativação. Durante a entrada de dados o cursor assume uma forma de sinal de soma. O término da entrada de dados é sinalizado por um clique duplo.
  A última primitiva criada fica automaticamente selecionada até que o usuário altere essa situação. A seleção é sinalizada por pontos de manipulação. Através dos pontos de manipulação podem ser feitas alteração de forma sobre a primitiva. Fora dos pontos de manipulação, apontando o cursor ao longo do traçado, podem ser feitas movimentações da mesma. Os pontos de manipulação para alteração de forma e movimentação se diferenciam pelo cursor. Ao se aproximar dos pontos de alteração de forma o cursor assume uma forma de seta dupla, para os pontos de movimentação o cursor assume uma forma de seta dobrada associada a um sinal de soma com setas duplas.
  Se, após ter ativado o botão referente à primitiva de interação, o usuário desistir de executar o traçado, ele poderá desativá-lo com um novo clique (modelo de interação para botões), ativar outro botão ou clicar na área de canvas. No caso de ativação da primitiva de desenho livre ele deverá clicar duas vezes na área de canvas para desativá-la. O sistema trabalha com estado de criação (cursor sinal de soma), estado de seleção (cursor em seta) e estado neutro (cursor I).

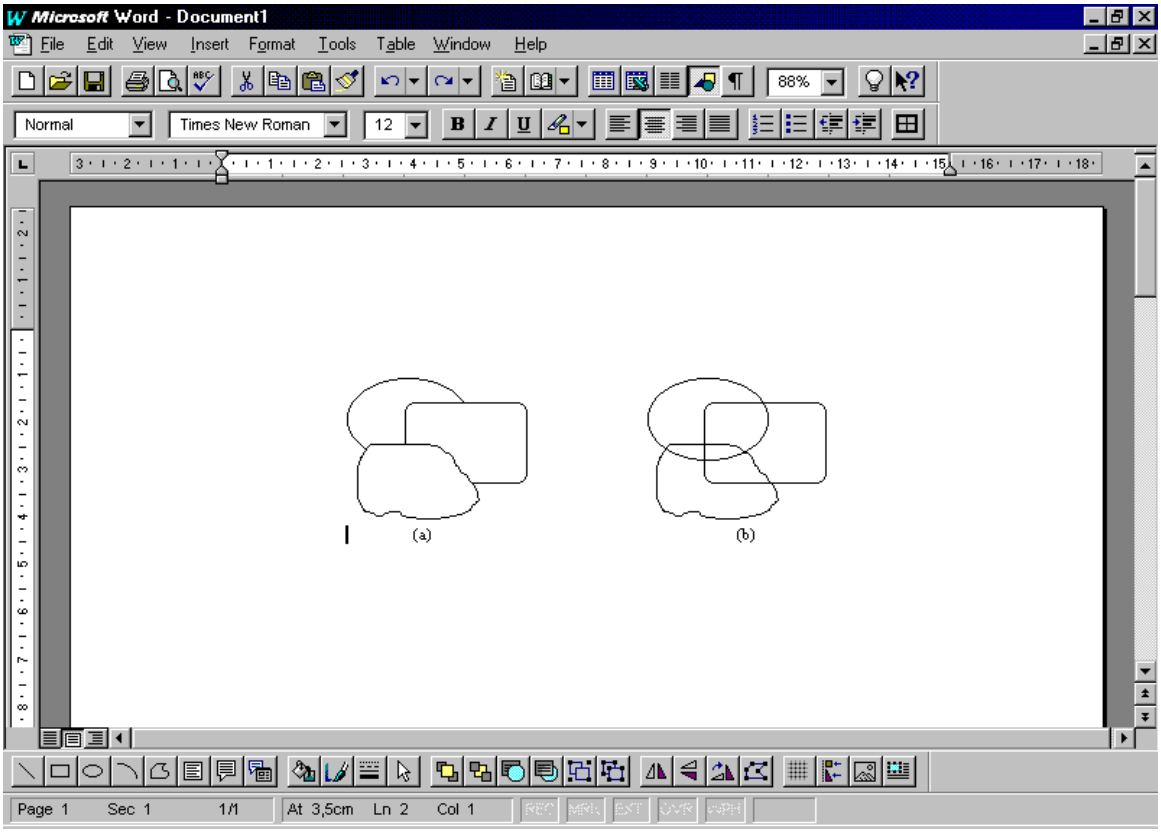


Figura A.1. Primitiva de interação no MsDraw e sobreposição de primitivas traçadas

- As primitivas traçadas obedecem a uma hierarquia de desenho, sobrepondo-se visualmente na ordem em que são desenhadas. No caso de figuras fechadas, se não forem vazadas (interior não cheio), as mais recentes escondem as anteriores. Por exemplo, a Figura A.1 também ilustra a sobreposição de três primitivas, uma forma oval sobreposta por uma forma retangular, sobreposta por uma forma livre. Essa hierarquia é determinada pela ordem de traçado e pode ser alterada por comando do usuário. Na Figura A.1 (a) as primitivas "cheias" permitem a identificação visual da hierarquia, na Figura A.1 (b) as primitivas "vazadas" não permitem a identificação, embora a situação seja a mesma.
  Se o estado "atrair para a grade" (Snap to Grid) estiver ligado e o usuário sobrepuser dois pontos ao traçar uma polilinha, o sistema fecha uma área encerrando o traçado automaticamente. Este procedimento não vale para desenho livre. No conjunto de aplicativos analisado, somente no MsDraw foi verificada essa característica.

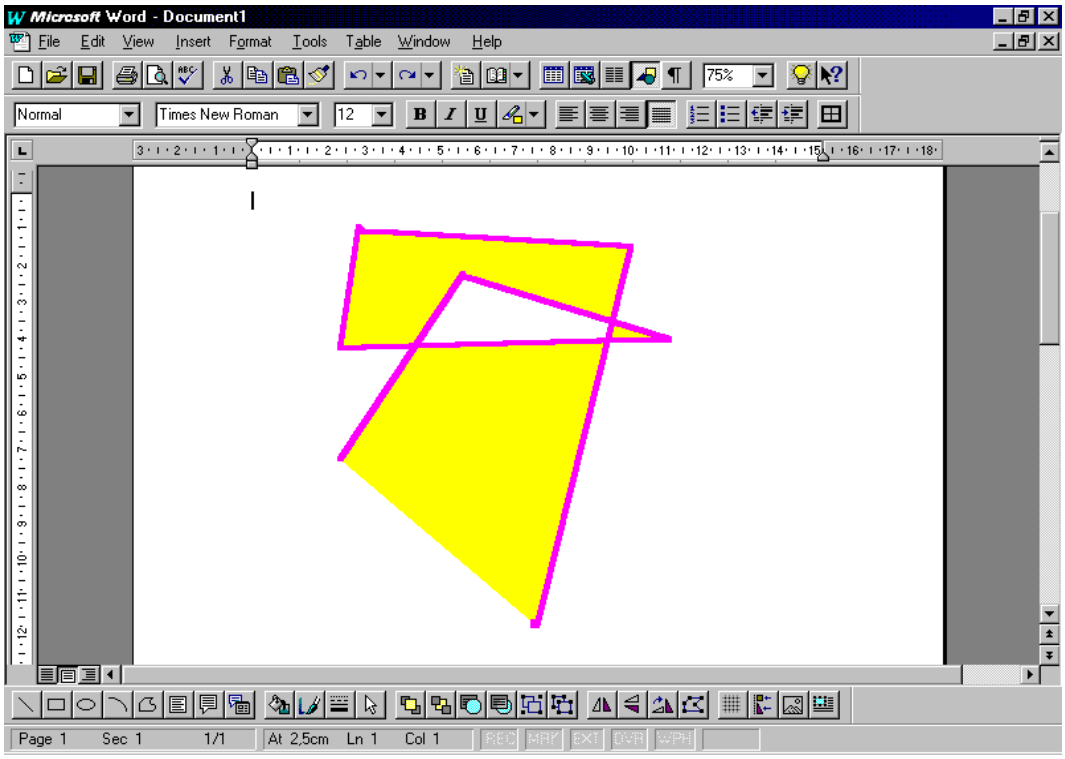
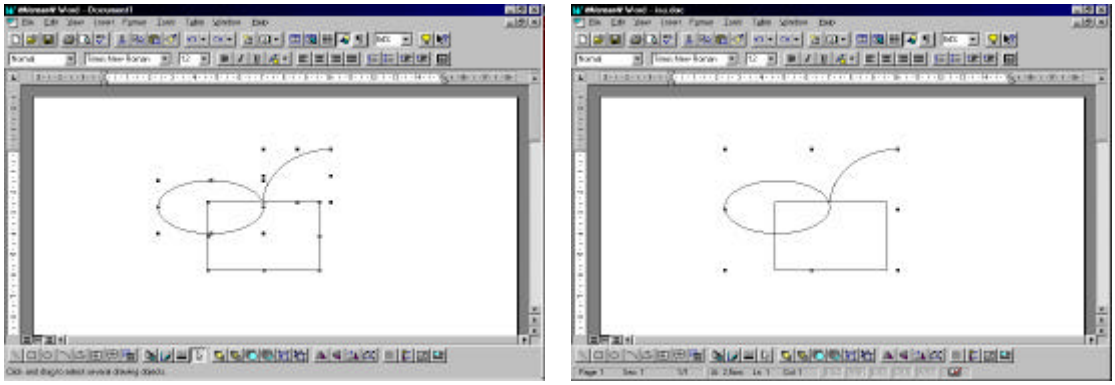


Figura A.2. Poligonal aberta traçada com opção de preenchimento

Se for traçada uma poligonal aberta e for solicitado ao sistema a opção de Fill, o sistema traça uma linha imaginária fechando a poligonal e faz o preenchimento. Caso a poligonal se auto-intercepte ele determina algumas áreas como sendo internas e outras como externas, conforme ilustrado na Figura A.2.

- As primitivas traçadas não perdem sua identidade, no que se refere à sua natureza e à sua individualidade. Por exemplo, um polígono oferecido nas primitivas de interação contínua sendo tratado como tal no domínio do sistema. O usuário tem opção de manipular as primitivas em conjunto, de forma temporária através de seleção, ou de forma definitiva através de agrupamento. O agrupamento temporário está ilustrado na (Figura A.3 (a)), o definitivo na (Figura A.3 (b)). Nesta figura é apresentado o agrupamento de três primitivas: uma elipse, um arco e um retângulo. A condição de agrupamento definitivo é expressa pelos sinalizadores de manipulação que são colocados apenas em torno do grupo, como este fosse um único objeto. Tanto no agrupamento temporário como no definitivo as primitivas são tratadas como um objeto único. Em ambos os casos as primitivas a serem tratadas em conjunto não precisam estar uma ao lado da outra, podem estar misturadas com outras que não farão parte do conjunto. O agrupamento definitivo é reversível, podendo ser desagrupado. A manipulação das primitivas em grupo é novamente tratada na seção sobre operação de alteração de forma.



(a) **(b)**

Figura A.3. Agrupamento de primitivas no MsDraw

- Os atributos associados às primitivas têm características estéticas, como, por exemplo, cor e espessura de linha. No que se refere a associação de atributos, as primitivas traçadas são tratadas em dois componentes: contorno e interior, cada um deles tendo seu próprio conjunto de atributos. A Tabela A.1 mostra um resumo do conteúdo manipulado nas operações de criação de primitivas e na associação de atributos. Essa diferença é importante para a linguagem de interação, porque embora o conteúdo manipulado seja diferente em cada uma das operações, essa diferença não é visualmente expressa. Visualmente o objeto tratado é o mesmo, em ambos os casos.

Operações	Conteúdo Manipulado
Operação de criação	- primitivas de interação
Associação de atributos	- contorno das primitivas - interior das primitivas

Tabela A.1. Conteúdo manipulado nas operações dos desenhadores

Os atributos associados às primitivas atuam como estados do sistema. Os atributos correntes são automaticamente associados a toda primitiva criada. Por exemplo, se a cor verde estiver ativa para o componente contorno, toda primitiva traçada terá seu contorno em verde. Se o usuário não ativar atributos específicos, o sistema utiliza os atributos padrão.

Tabulação dos Dados da Operação de Criação no MsDraw

Na tabulação dos dados o conteúdo está distinto do recurso expressivo. Em cada coluna estão registrados o tipo de conteúdo e a instância de conteúdo, o tipo expressivo e a instância expressiva. Por exemplo, tipo de expressão do sistema - sinalizador, instância de expressão do sistema - cursor em seta. A instância será sempre colocada entre parênteses, na forma: sinalizador (cursor em cruz). Cada linha da tabela corresponde à expressão de um dos agentes da comunicação. Por exemplo, uma tabela com sete linhas significa que ocorreram sete passos na interação entre usuário e sistema, sendo que em cada passo da interação alternam-se os agentes. A tabela se propõe a registrar a realidade dos sistemas.

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (neutro) primitivas de interação (linha)	sinalizador (cursor I) menu (botão)		
2			indicação	clique

3	estado sistema (criação) primitiva de interação (linha)	sinalizador (cursor sinal de soma) menu (botão reverso)		
4			indicação	pressão-e-arrasto
5	dado (ponto inicial)	sinalizador (rubber-band)		
6			indicação	soltar pressão
7	estado sistema (neutro) primitivas de interação (linha) dado (linha) estado dado (selecionado)	sinalizador (cursor I) menu (botão) forma (segmento de reta) sinalizador (pontos manipulação)		

Tabela A.2. Dados referentes à operação de criação no MsDraw

Criação de Objeto no AutoCAD

- A Figura A.4 ilustra as primitivas de interação no AutoCAD, no menu lateral. As primitivas de traçado são separadas em categorias, as de traçado de contornos em 2D (lateral da tela na Figura A.4) e as de traçados de sólidos (parte de baixo da tela na Figura A.4). Na de traçado de sólidos são traçadas as indicações para que posteriormente a figura seja transformada em um sólido de fato, através do modelador 3D. No traçado, os objetos se sobrepõem sem interferirem uns com os outros. As primitivas traçadas não perdem sua identidade, no que se refere à sua natureza e à sua individualidade.

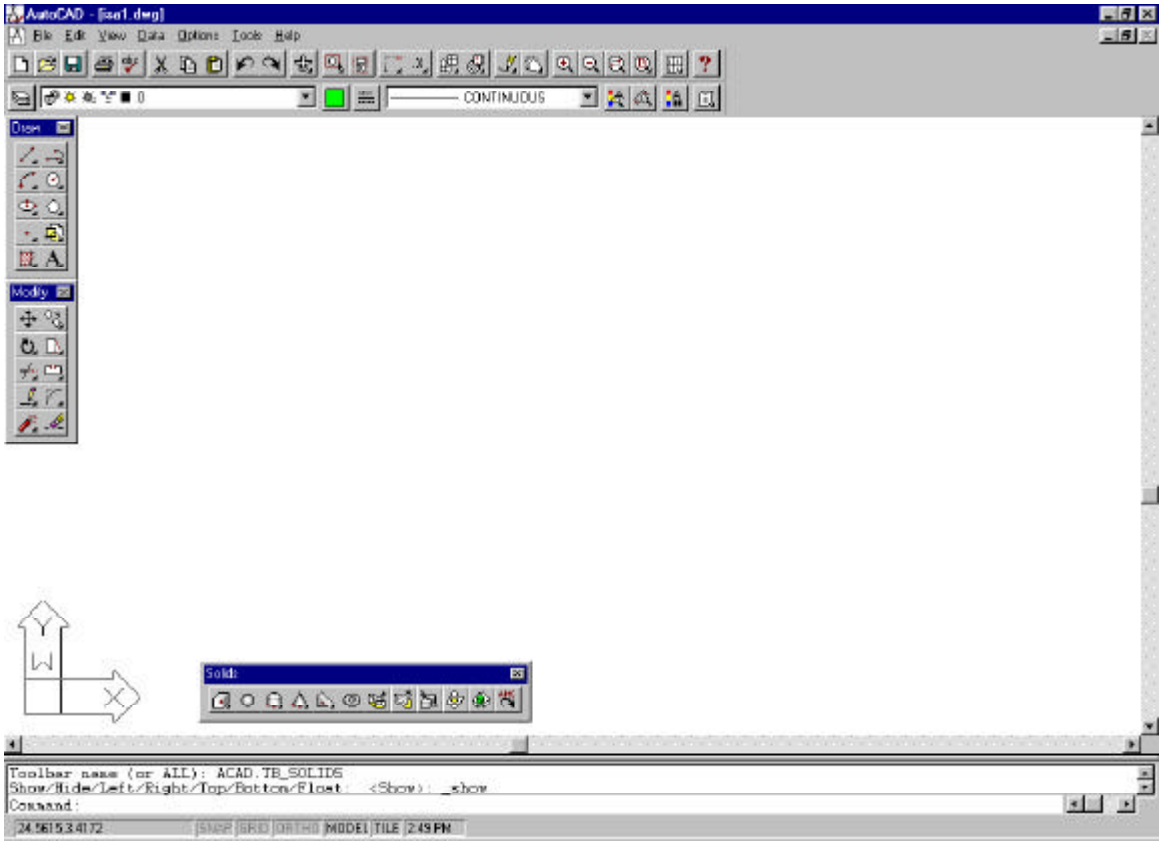
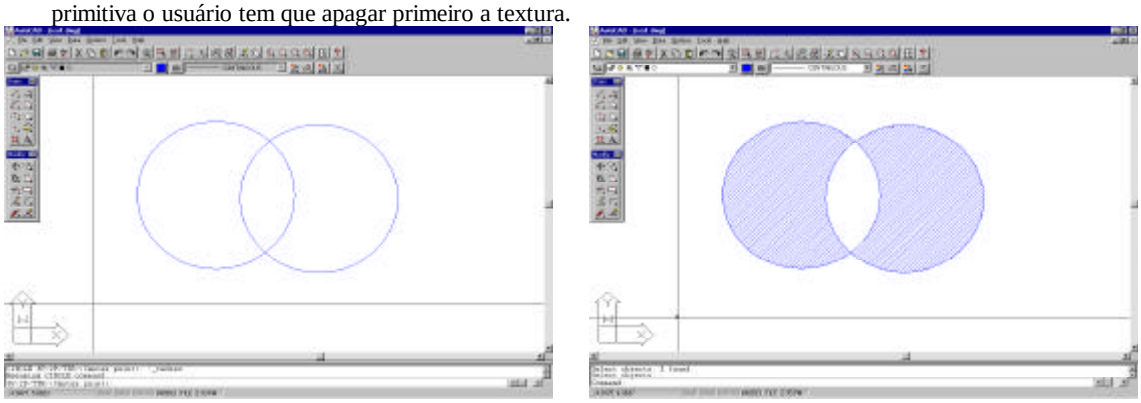


Figura A.4. Primitiva de interação no AutoCad

- A ativação da criação da primitiva é feita uma a uma, através do menu. O botão do menu não oferece nenhuma indicação da primitiva que foi ativada (não oferece retorno). As primitivas são traçadas por cliques simples. Aquelas que precisam interrupção linha e polilinha) a interrupção é feita por clique no botão da direita. O círculo é traçado por dois cliques, que delimitam o raio. O arco é traçado por três cliques, os dois primeiros delimitam o diâmetro, o terceiro é dado após o posicionamento do arco conforme o usuário deseja. Da mesma forma a elipse, os dois primeiros cliques delimitam o raio de um círculo inicial e o terceiro clique é dado após o ajuste da elipse. Durante o traçado de todas as primitivas oferece rubber-band, exceto no arco. Entre o primeiro clique e o segundo clique do arco só aparecem cruzinhas no ponto clicado.
  Em estado de espera o cursor em quadradinho é preso a duas linhas de orientação alinhadas peos eixos x e y (Figura A.5), na linha de mensagens são assinaladas as coordenadas por onde passa o cursor. Quando o sistema aguarda a entrada de dados o cursor perde a forma de quadradinho ficando apenas as linhas de orientação.
  Existem duas primitivas de interação para traçado de linhas, a polilinha e a linha simples. Através da linha simples o aplicativo permite traçar uma sequência de linhas com a aparência visual de uma polilinha. Através de cliques sucessivos o usuário traça segmentos de retas interligados. A diferença de comportamento pode ser notada no processo de seleção, a polilinha é tratada como um objeto único, na linha cada segmento de reta é tratado como um objeto independente.
  Diferente da maioria dos aplicativos gráficos a última primitiva criada não fica automaticamente selecionada.
- Em cada ponto do canvas que o usuário clicar aparece uma pequena marca, por qualquer motivo, mesmo quando o usuário traçe uma janela. Sendo solicitado um redraw as marcas desaparecem.
- Os atributos associados às primitivas têm características estéticas, como, por exemplo, cor. A cor é um atributo estado do sistema. O preenchimento com hachurado não é um atributo associado automaticamente, conforme Figura A.5 (a). No hachurado o sistema faz algum tipo de interpretação sobre o traçado. No preenchimento das primitivas ele não preenche as áreas de interseção, conforme Figura A.5 (b). Para apagar a primitiva o usuário tem que apagar primeiro a textura.



(a) **(b)**

Figura A.5. Preenchimento de área no AutoCAD

Tabulação dos Dados da Operação de Criação no AutoCAD

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (neutro) primitivas de interação (linha)	sinalizador (cursor quadradinho com linhas de orientação) menu (botão)		
2			indicação	clique
3	estado sistema (criação)	sinalizador (cursor linhas de orientação)		
4			indicação	clique

5	dado (ponto inicial)	sinalizador (rubber-band)		
6			indicação	clique
Iteração 4 a 6				
7			interrupção	botão direita
8	estado sistema (neutro) dado (linha)	sinalizador (cursor quadradinho com linhas de orientação) forma (segmento de reta)		

Tabela A.3. Dados referentes à operação de criação no AutoCAD

Criação de Objeto no MGE

- A Figura A.6 ilustra as primitivas de interação do MGE, no menu lateral. O menu oferece grupo de primitivas: elementos lineares (linha, polilinha, multi-linha, ponto, pontos sobre linhas já existentes), elementos circulares (círculo, elipse, arco), poligonais (retângulo, poligonal fechada, polígono regular). Em cada grupo uma primitiva fica ativa no menu, as outras aparecem clicando duplo sobre o menu.
  A entrada de dados é um estado do sistema. O botão em reverso no menu indica que o sistema está aguardando a entrada de dados para aquela primitiva. O traçado das primitiva é executado por cliques simples. O estado de criação de uma primitiva fica ativado até que seja substituído por outro estado de criação ou pelo estado de seleção. O estado de seleção de primitivas concorre com a entrada de dados. Isto significa que traçar novas primitivas e manipular as já traçadas são ações mutuamente exclusivas. As primitivas criadas não são automaticamente selecionadas.
  O cursor é diferenciado para cada situação: estado de seleção - cursor seta com bolinha, estado de entrada de dados - cursor sinal de soma grande, estado de alteração de primitiva já traçada - cursor X grande.
  A polilinha e a linha são traçadas de modo semelhante, com cliques sucessivos. Ambas exigem interrupção do traçado. A interrupção é feita no botão da direita. Através da linha simples o aplicativo permite traçar uma sequência de linhas com a aparência visual de uma polilinha. Através de cliques sucessivos o usuário traça segmentos de retas interligados. A diferença de comportamento pode ser notada no processo de seleção, a polilinha é tratada como um objeto único, na linha cada segmento de reta é tratado como um objeto independente. As primitivas traçadas se sobrepõem umas às outras visualmente. As primitivas traçadas não perdem sua identidade, no que se refere à sua natureza e à sua individualidade.
  Os atributos atuam como estados do sistema. Os atributos têm características estéticas, como por exemplo cor. Para cada primitiva de interação assinalada no menu o sistema oferece uma pequena janela com os atributos correntes, que serão associados àquela primitiva. Entre esse atributos estão a cor de preenchimento, a espessura de linha e a forma de traçado. Por exemplo, para um polígono regular pode ser especificado o número de arestas, no traçado de um segmento de reta ele permite fixar o ângulo de traçado.

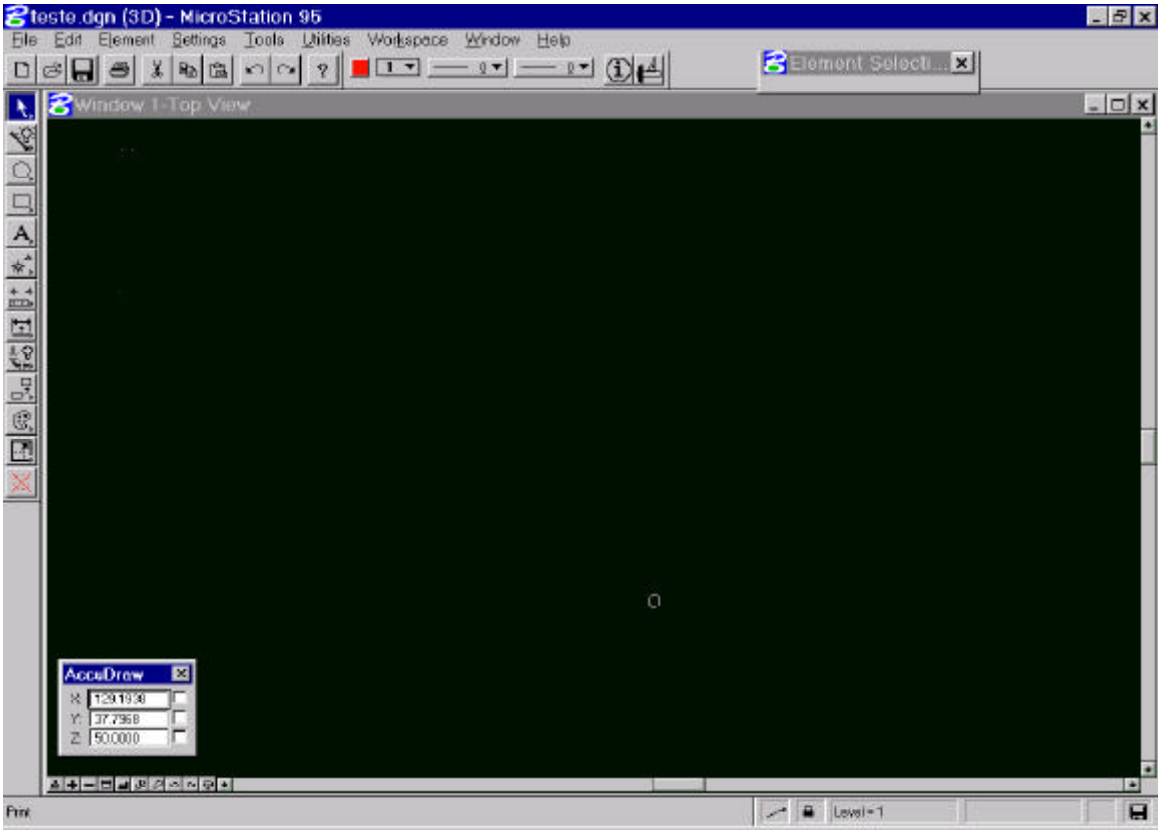


Figura A.6. Primitiva de interação no MGE

Tabulação dos Dados da Operação de Criação no MGE

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (seleção) primitivas de interação (linha)	sinalizador (cursor seta com bolinha) menu (botão)		
2			indicação	clique
3	estado sistema (criação) primitiva (linha)	sinalizador (cursor sinal de soma grande) menu (botão reverso)		
4			indicação	clique
5	dado (ponto inicial)	sinalizador (cursor X + rubber-band)		
6			indicação	clique
Iteração 4 a 6				
7			interrupção	botão direita
8	estado sistema (criação) primitivas de interação (linha) dado (linha)	sinalizador (cursor sinal de soma grande) menu (botão reverso) forma (segmento de reta)		
Iteração 4 a 8				
9			indicação	clique
10	estado sistema (seleção) primitivas de interação (linha)	sinalizador (cursor seta com bolinha) menu (botão)		

Tabela A.4. Dados referentes à operação de criação no MGE

As características do MTool foram registradas comparativamente ao desenhador MsDraw.

Criação de Objeto no Mtool

- O conjunto de primitivas de interação é uma antecipação dos elementos geométricos de estruturas submetidas aos sistemas de cálculo e simulação. A Figura A.7 ilustra as primitivas de interação do MTool, no menu à esquerda.
- Diferente dos desenhadores, no MTool as primitivas de interação perdem sua característica inicial. Depois de traçadas, elas se transformam em primitivas geométricas de acordo com o modelo semântico. Por exemplo, um círculo traçado é tratado pelo sistema como um conjunto de arestas. Se o usuário quiser apagar a entidade círculo não vai conseguir, porque, após traçada, ela não existe mais. Outro exemplo, quando duas arestas se interceptam elas são subdivididas em quatro.
  É interessante observar que após o traçado o círculo, como todas as primitivas, fica automaticamente selecionado. Nessa situação o círculo é selecionado por inteiro. Durante a manipulação, se o traçado do círculo for novamente selecionado, ele será selecionado em partes, pelo menos dois semi-círculos. Internamente ele é tratado como dois conjuntos de arestas separados.
  A hierarquia entre as primitivas do sistema obedece ao modelo matemático. Por exemplo, matematicamente uma aresta tem hierarquia superior ao vértice, uma vez que a aresta é definida por dois vértices. Da mesma forma, a face tem hierarquia superior à aresta, uma vez que ela é definida por um conjunto de arestas. Nos PPEF, quando o usuário traça uma linha, o sistema cria automaticamente dois vértices, que a suportam. Nos desenhadores isto não acontece, a linha é uma primitiva independente - é apenas um traço. Nos PPEF a hierarquia das primitivas não é visual, é semântica, seus efeitos serão percebidos nas restrições impostas à manipulação.
- Nos PPEF as primitivas geométricas são tratadas como partes de um conjunto. Cada primitiva traçada é articulada com outras adjacentes, para compor um objeto gráfico segundo um modelo de engenharia. O sistema captura a geometria e a topologia do objeto gráfico. No caso do MTool o modelo semântico é permanentemente ativo, sendo a consistência do objeto verificada durante a criação. Embora as primitivas geométricas sejam tratadas como partes de um conjunto, elas podem ser editadas individuamente, alterando as características do objeto global.

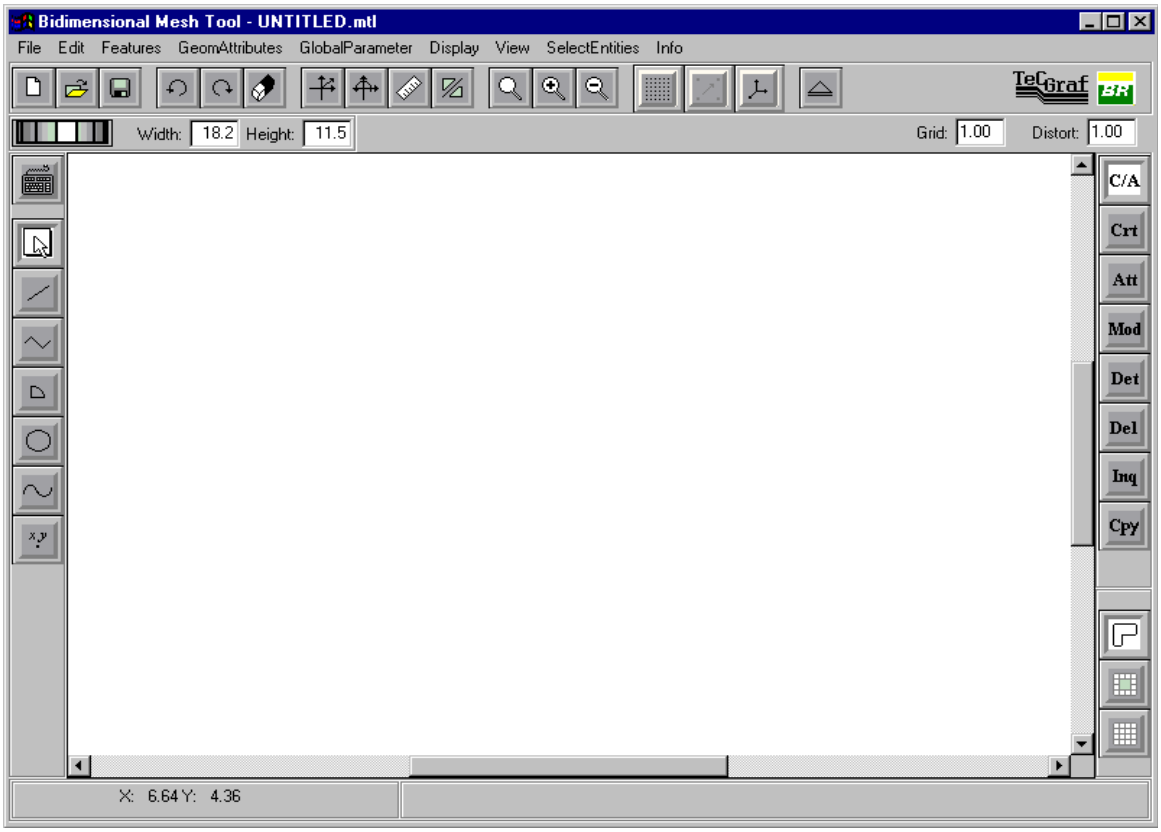


Figura A.7. Primitiva de interação no MTool

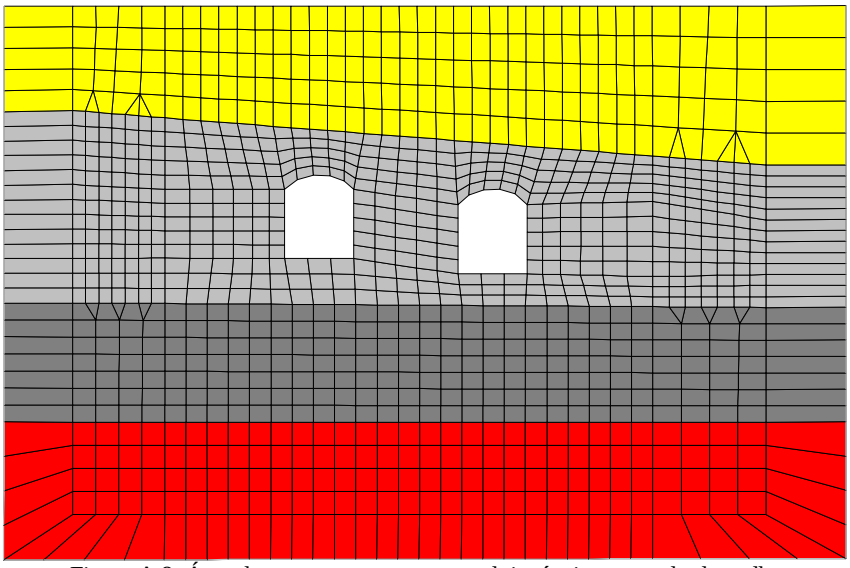


Figura A.8. Área de terreno em corte, com dois túneis, e traçado de malha de elementos finitos

Através da geometria do objeto, o sistema não tem com identificar a ocorrência de um buraco. O buraco é uma qualificação da geometria traçada, devendo ser explicitado pelo usuário. No MTool ele é associado à face como um atributo semântico, para que posteriormente o sistema de cálculo possa atuar corretamente. Por exemplo, a Figura A.8 contém o traçado correspondente a dois túneis em corte. Na área correspondente ao túnel a informação de buraco é associada como um atributo semântico. Para um melhor esclarecimento do problema, vamos exemplificar com a Figura A.9 em 3D. Essa figura contém uma representação aramada. Na forma em que está apresentada, ela tanto pode ser sólida no canal interno, vazada na área de contorno e recoberta por uma casca, como pode ser vazada no canal interno. Em ambos os casos a representação visual será a mesma, assim como a representação interna de dados. O usuário deverá especificar a ocorrência do buraco.

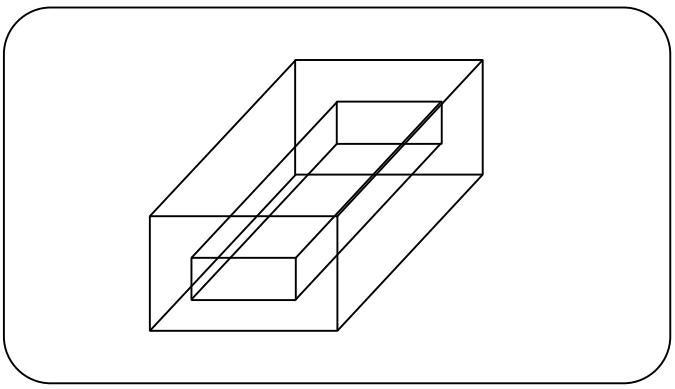


Figura A.9. Figura aramada em 3D

- De forma semelhante aos desenhadores, a entrada de dados é implementada como um estado do sistema. Uma vez ativada a criação de uma primitiva, toda ativação subsequente é interpretada como uma entrada de dados, até que este estado seja desativado. O estado de criação de uma primitiva pode ser desativado pela ativação de outra primitiva, ou pela tecla Escape (botão da direita).

  A entrada de dados para todas as primitivas é feita por cliques simples sucessivos. Nas primitivas que exigem encerramento explícito da entrada de dados (polilinha e Bezier), ele é executado por um clique no botão da direita. Nas duas últimas o número de pontos é fixo, mas o traçado original fica disponível para manipulação (Figura A.10), até que o usuário defina a forma desejada e ative o encerramento.

- A última primitiva traçada fica selecionada até que o usuário altere essa situação.

- Os atributos são específicos de cada primitiva e têm características técnicas ligadas ao domínio da aplicação. Por exemplo, materiais e forças são atributos de faces. Os atributos são associados às primitivas de acordo com a solicitação do usuário.

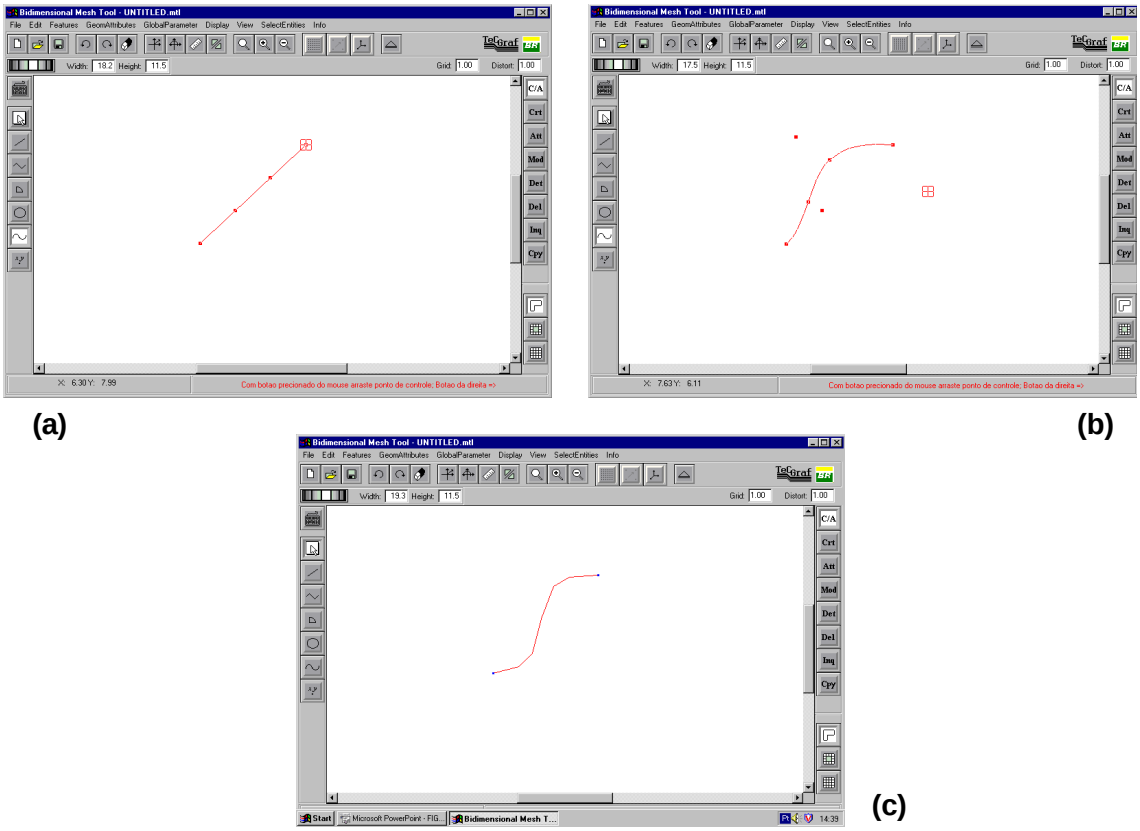


Figura A.10. Criação da curva de Bezier

Tabulação dos Dados da Operação de Criação no MTool

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (seleção) primitivas de interação (linha)	sinalizador (cursor seta) menu (botão)		
2			indicação	clique
3	estado sistema (criação) primitiva de interação (linha)	sinalizador (cursor quadradinho em quatro) menu (botão reverso)		
4			indicação	clique
5	dado (ponto inicial)	sinalizador (rubber-band)		
6			indicação	clique
7	estado sistema (criação) primitivas de interação (linha) dado (linha) estado dado (selecionado)	sinalizador (cursor quadradinho em quatro) menu (botão reverso) forma/cor (pontinhos azuis + segmento de reta vermelho) cor (vermelha)		
Iteração 4 a 7				
8			indicação	clique
9	estado sistema (seleção) primitivas de interação (linha) dado (linha) estado dado (selecionado)	sinalizador (cursor seta) menu (botão) forma/cor (pontinhos azuis + segmento de reta vermelho) cor (vermelha)		

Tabela A.5. Dados referentes à operação de criação no MTool

A.2. Operações de Apagamento

Observamos os seguintes aspectos nas operações de apagamento, nos aplicativos de referência.

Apagamento de Objeto no MsDraw

- A operação de apagamento tem como pré-requisito a seleção dos objetos. Conceitualmente essa operação atua sobre um conjunto de objetos selecionados, o qual pode ser unitário. A especificação desse conjunto obedece a diferentes critérios, podendo ser extensional ou intensional. Cada critério tem uma forma de expressão específica.
  Para um conjunto unitário, a forma mais comum de seleção por MD é um clique sobre a primitiva traçada. Para a seleção de conjuntos não unitários, existem duas formas bastante comuns de ativação. Uma delas é intensional e a relação de pertinência é definida pela localização da primitiva no plano de visualização. A seleção desse conjunto é expressa por circunscrição de um espaço físico na tela, através de uma janela. Esta forma de seleção é adequada quando as primitivas de interesse estão fisicamente agrupadas. Outra forma é extensional onde o conjunto é definido um a um, clicando sobre as primitivas simultaneamente à tecla <shift>. Esta forma de seleção é adequada quando as primitivas estão fisicamente dispersas, ou próximas mas misturadas a outras que não devem ser selecionadas.
  O estado de seleção é visualmente representado por pontos de manipulação sobre as primitivas. Exemplos foram apresentados na Figura A.3.
- Embora as operação de corte e apagamento não tenham o mesmo comportamento, em alguns casos são utilizadas com o mesmo objetivo, por produzirem o mesmo efeito visual. A primeira é comandada por menu e guarda o conteúdo cortado em uma área de trabalho para recuperação posterior, enquanto outro elemento não for cortado. A segunda é comandada por tecla (Del) e apenas apaga o conteúdo solicitado, que só pode ser recuperado por comando de Undo, imediatamente após o apagamento.

Tabulação dos Dados da Operação de Apagamento no MsDraw

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	estado sistema (seleção) operação (apagamento) dado (linha)	sinalizador (cursor seta) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação)		
4			ativação	clique

Tabela A.6. Dados referentes à operação de apagamento no MsDraw

---

Apagamento de Objeto no AutoCAD

- A operação de apagamento tem como pré-requisito a seleção dos objetos. A seleção individual dos objetos é feita clicando-se sobre eles. A seleção de um objeto não interfere com outra seleção previamente especificada (a anterior não é desfeita), ele vai acumulando objetos selecionados. Portanto, o uso da tecla shift para selecionar uma a uma mais de uma primitiva, muitos comum na maioria dos aplicativos, neste caso não é necessária. A seleção de objetos pode ser feita por janela. A definição da janela é feita por dois cliques (e não por pressão e arrasto como na maioria dos aplicativos).

O apagamento dos objetos é feito pelo Cut, não aceita tecla Del. Portanto, toda primitiva apagada vai para a área de clipping, podendo ser recuperada.

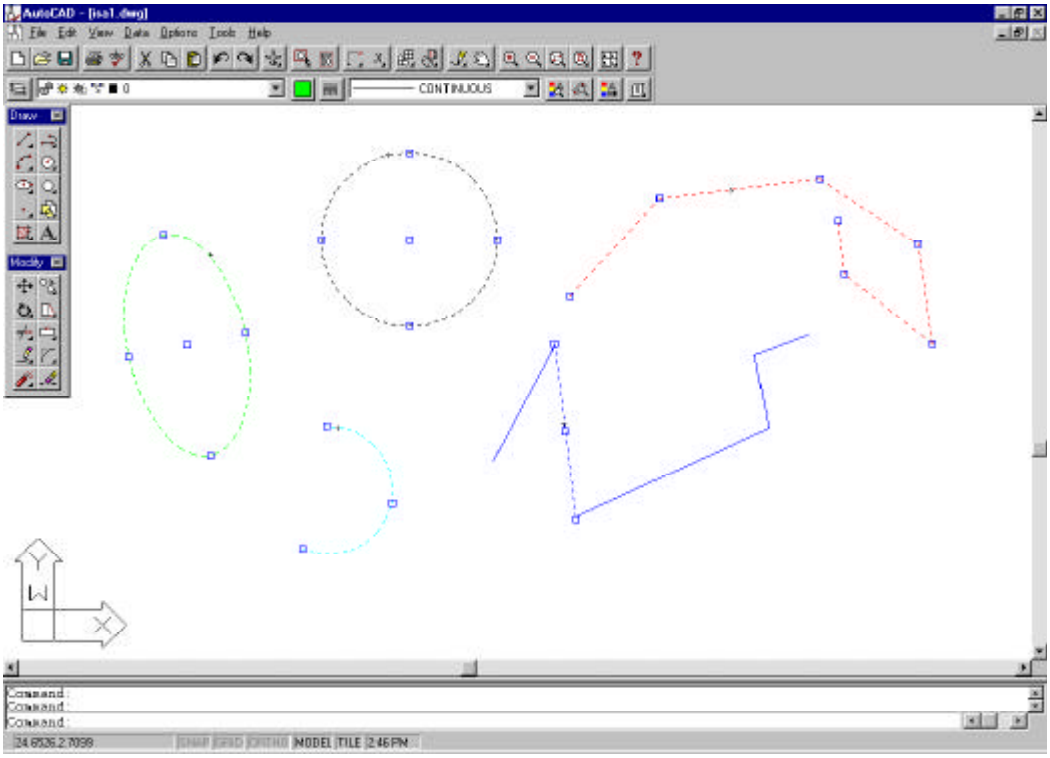


Figura A.11. Primitivas selecionadas no AutoCAD

Quando uma primitiva é selecionada ela tem seu traçado alterado de sólido para tracejado e são apresentados pontos de manipulação. Alguns pontos são assinalado com quadradinhos e outros com quadradinhos com sinal de soma no meio, conforme Figura A.11. Na polilinha todas as pontas dos segmentos ficam com quadradinho com sinal de soma. Na linha as pontas ficam com quadradinho com sinal de soma e o meio fica com quadradinho simples. No meio carrega (simples), nas pontas deforma (com mais). Na elipse e no círculo o centro fica com quadradinho com mais e no contorno tem 4 quadradinhos simples. No centro carrega (com mais), no contorno deforma (simples). É o caso inverso das linhas. No arco é semelhante às linhas, quadradinho com mais nas pontas e simples no centro, só que os três deformam. São três situações diferentes.

Tabulação dos Dados da Operação de Apagamento no AutoCAD

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (neutro) operação (apagamento) dado (linha)	sinalizador (cursor quadradinho com linhas de orientação) menu (ícone) forma (segmento de reta)		
2			indicação	clique

3	estado do dado (selecionado)	sinalizador (pontos de manipulação em azul + traçado tracejado)		
4			ativação	clique

Tabela A.7. Dados referentes à operação de apagamento no AutoCAD

Apagamento de Objeto no MGE

- O aplicativo oferece duas formas de apagamento. Uma delas, o sistema estando em estado de seleção, selecionar as primitivas e apagá-las por Del ou Cut. As primitivas podem ser selecionadas uma a uma clicando em cima. Neste caso a seleção é mutuamente exclusiva, uma primitiva selecionada implica na desseleção de outra. Ainda com o sistema em estado de seleção, pode ser traçada uma janela através de pressão-e-arrasto, selecionando as primitivas circunscritas a esse espaço. Um clique na área de canvas desseleciona todas as primitivas.
  Outra forma de apagar é entrar em estado de apagamento através do menu. Neste caso não há necessidade de seleção prévia das primitivas e ativação do apagamento por menu. O usuário clica sobre a primitiva, ela tem sua cor alterada para branco, clicando novamente ela se apaga. No estado de apagamento o cursor é um sinal de mais com uma bolinha.
- Diferente dos outros aplicativos, o MGE oferece apagamentos parciais. Ele oferece o apagamento parcial no menu de alterações. Neste caso o cursor funciona como uma borracha que vai sendo passada sobre o traçado, apagando. Esta opção não foi tabulada.
- O MGE trabalha com o conceito de cerca (fence). A cerca é um delimitador semelhante à janela (bounding-box), mas ela tem associadas a si operações de seleção diferentes da comum associada a janela. Nesta, normalmente são selecionados os objetos totalmente inseridos no espaço circunscrito pela janela, aqueles que atravessam a fronteira da janel normalmente não são selecionados. No MGE, em função de um modelo conceitual voltado para modelagem geográfica, a cerca permite selecionar, por exemplo, o que está dentro, o que está fora, o que está dentro e o que está passando, e outros.

Tabulação dos Dados da Operação de Apagamento no MGE

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	estado sistema (seleção) operação (apagamento) dado (linha)	sinalizador (cursor seta com bolinha) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação)		
4			ativação	clique

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (seleção) operação (estado de apagamento) dado (linha)	sinalizador (cursor seta com bolinha) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado sistema (apagamento) operação (estado de apagamento)	sinalizador (cursor soma grande com bolinha) menu (ícone reverso)		
4			indicação	clique
5	estado do dado (indicado para apagamento)	cor (branco)		
6			indicação	clique

Tabela A.8. Dados referentes a duas formas de apagamento no MGE

Apagamento de Objeto no MTool

- O estado de seleção concorre com o estado de criação de primitivas. A seleção só pode ser executada se o aplicativo estiver com o estado de seleção ativado. O MTool tem uma particularidade com relação à seleção, não é permitida a seleção simultânea de diferentes primitivas semânticas. Por exemplo, não é permitido selecionar arestas e faces ao mesmo tempo. A seleção intensional de conjuntos não unitários é feita através da manipulação de uma janela, porém o sistema questiona o usuário sobre o tipo de primitiva que ele deseja selecionar na área circunscrita (vértices, arestas ou faces). De forma semelhante, a seleção extensional de conjuntos é feita clicando-se sobre a primitiva simultaneamente à tecla <shift>. Neste caso, o sistema ignora a seleção de primitivas semânticas diferentes.
  O estado de seleção é expresso através de cor. Os vértices, que são azuis, quando selecionados ficam vermelhos. As arestas, que são pretas, quando selecionadas ficam vermelhas. As faces, que podem ser de diferentes cores, dependendo do atributo que têm associado a si, quando selecionadas mantém a cor corrente, hachurada de vermelho.
- Nos PPEF a operação de apagamento sofre restrições do modelo semântico. Por exemplo, o apagamento de um vértice ligado a uma aresta é ilegal porque destruiria a aresta. O apagamento de uma face que tenha faces adjacentes é ilegal, porque as faces adjacentes seriam destruidas. No sistema MTool são verificadas algumas incoerências de comportamento, por exemplo, o sistema não permite apagar um vértice ligado a uma aresta, mas permite apagar uma aresta que compõe uma face.

Tabulação dos Dados da Operação de Apagamento no MTool

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (seleção) operação (apagamento) dado (linha)	sinalizador (cursor seta) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	cor (vermelha)		
4			ativação	clique

Tabela A.9. Dados referentes à operação de apagamento no MTool

A.3. Operações de Movimentação

Observamos os seguintes aspectos nas operações de movimentação dos objetos gráficos, nos aplicativos de referência.

Movimentação de Objeto no MsDraw

- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção já foram descritos. Ao aproximar o cursor do objeto selecionado, nos pontos de alteração de forma o cursor se transforma em uma seta dupla, nos pontos de movimentação ele ganha um sinal de soma com setas duplas, associado à seta. A operação movimenta simultaneamente todo o conjunto de primitivas selecionadas.

Tabulação dos Dados da Operação de Movimentação no MsDraw

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	estado sistema (seleção) dado (linha)	sinalizador (cursor seta) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação)		
4			varredura	posicionamento
5	estado do dado (apontado)	sinalizador (cursor seta e sinal soma)		
6			indicação	pressão-e-arrasto
7	estado do dado (em movimento)	forma (tracejada)		
8			indicação	soltar botão
9	estado do dado (selecionado)	forma (cheia) + sinalizador (pontos de manipulação)		

Tabela A.10. Dados referentes à operação de movimentação no MsDraw

- A movimentação por MD é feita através de um movimento de pressão-e-arrasto sobre o(s) objeto(s) selecionado(s). Após a movimentação os objetos continuam selecionados.

Movimentação de Objeto no AutoCAD

- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção já foram descritos. Neste aplicativo, mesmo havendo mais de uma primitiva selecionada apenas aquela cujo ponto de manipulação está sendo apontado pelo cursor é movimentada.
- A movimentação por MD é feita através de cliques simples sobre os pontos de manipulação. Quando ocorre um clique sobre um dos pontos de deformação ou de carregamento, o quadradinho do cursor fica vermelho, quando o mouse é arrastado uma rubber-band indica a movimentação que está sendo feita, e uma nova figura vai sendo traçada em linha cheia simultaneamente ao movimento. Quando é dado um segundo clique a nova

forma se estabelece, ficando selecionada como a original (tracejada e com quadradinhos de manipulação). A forma antiga é apagada.

Tabulação dos Dados da Operação de Movimentação no AutoCAD

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (neutro) dado (linha)	sinalizador (cursor quadradinho com linhas de orientação) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação em azul + traçado tracejado)		
4			varredura	posicionamento
5	estado do dado (apontado)	sinalizador (quadradinho do cursor em amarelo)		
6			indicação	clique
7	estado do dado (em movimento)	sinalizador (rubber-band + traçado em cheio + quadradinho do cursor em vermelho)		
8			indicação	clique
9	estado do dado (selecionado)	sinalizador (pontos de manipulação em azul + traçado tracejado)		

Tabela A.11. Dados referentes à operação de movimentação no AutoCAD

Movimentação de Objeto no MGE

- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção já foram descritos.
- A movimentação por MD é feita através de um movimento de pressão-e-arrasto sobre o(s) objeto(s) selecionado(s). Sobre os pontos de manipulação executa alteração de forma, sobre um ponto qualquer do objeto altera o lugar (semelhante ao MsDraw).

Tabulação dos Dados da Operação de Movimentação no MGE

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário

1	estado sistema (seleção) dado (linha)	sinalizador (cursor seta com bolinha) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação)		
4			indicação	pressão-e-arrasto
5	estado do sistema (em movimento)	sinalizador (cursor X)		
6			indicação	soltar botão
7	estado do dado (selecionado)	sinalizador (pontos de manipulação		

Tabela A.12. Dados referentes à operação de movimentação no MGE

Movimentação de Objeto no MTool

- Nos PPEF a movimentação está sujeita às restrições do modelo semântico. No Mtool a operação de movimentação é solicitada por menu, mas a indicação do movimento é feita por MD, através de um segmento de reta traçado na área de canvas. O movimento é feito na mesma direção, no mesmo sentido e de acordo com o tamanho do segmento de reta.
- A movimentação, da mesma forma que a rotação e o espelhamento, pode ser executada com opção de "deixar o original" (Leave Original). Isto significa que uma cópia do objeto é movimentada, deixando o objeto original no lugar, o que significa uma operação casada de cópia e movimentação.
- No MTool a movimentação de vértice pode causar alteração de forma nas arestas e faces. Ver tópico Operações de Alteração de Forma no MTool.

Tabulação dos Dados da Operação de Movimentação no MTool

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	estado sistema (seleção) operação (movimentação) dado (linha)	sinalizador (cursor seta) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	cor (vermelha)		
4			ativação	clique

5	estado sistema (entrada de dados)	sinalizador (cursor sinal soma grande vermelho)		
6			indicação	clique
7	estado sistema (entrada de dados)	sinalizador (cursor soma) + sinalizador (rubber-band)		
8			indicação	clique
9	estado sistema (seleção) estado do dado (selecionado)	sinalizador (cursor seta) sinalizador (cor vermelha)		

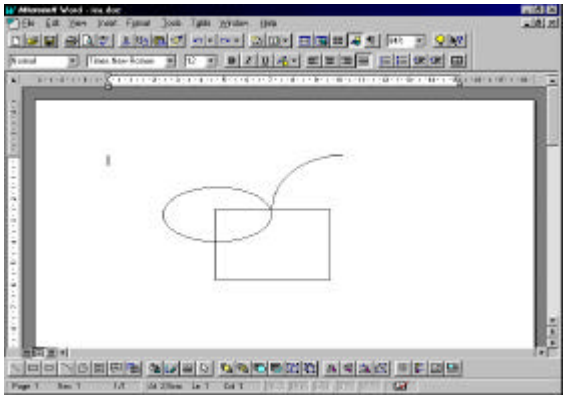
Tabela A.13. Dados referentes à operação de movimentação no MTool

A.4. Operações de Alteração de Forma

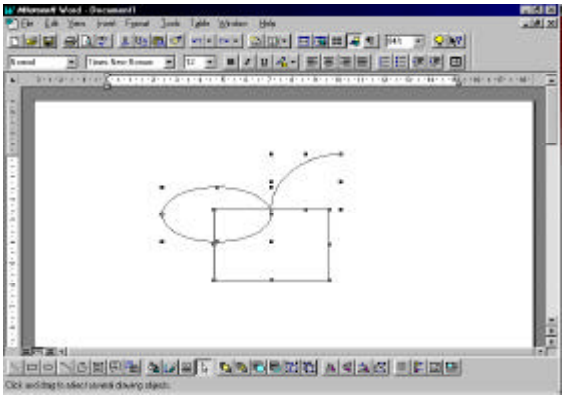
Observamos os seguintes aspectos nas operações de alteração de forma, nos aplicativos de referência.

Alteração de Forma de Objeto no MsDraw

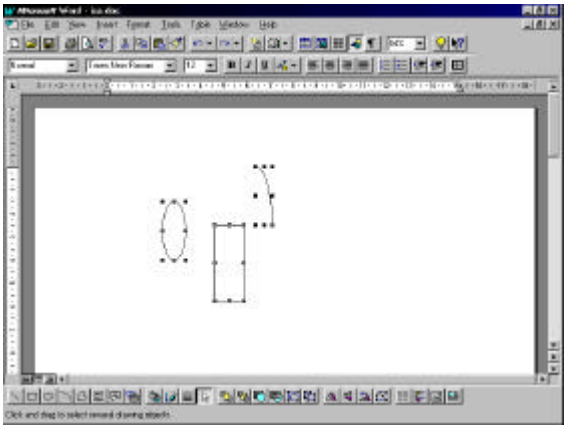
- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção são os mesmos já descritos. Quando um objeto é selecionado o sistema oferece pontos de manipulação para alteração de forma. Se um conjunto de primitivas (Figura A.12 (a)) estiver selecionado (Figura A.12 (b)), ao ser feita a manipulação de um ponto de uma delas todas terão sua forma alterada simultaneamente (Figura A.12 (c)). Se as primitivas forem agrupadas (Figura A.12 (d)), na alteração de forma serão tratadas como um objeto único (Figura A.12 (e)).
- No MsDraw só é permitida alteração de escala. Quando o usuário posiciona o cursor sobre um ponto de manipulação de uma primitiva e pressiona-e-arrasta o cursor na diagonal, a figura é escalonada simultaneamente nos eixos vertical e horizontal do plano cartesiano. Não é possível alterar a forma geral de uma primitiva. Por exemplo, os vértices de um polígono não podem ser carregados no sentido de alterar a sua forma.



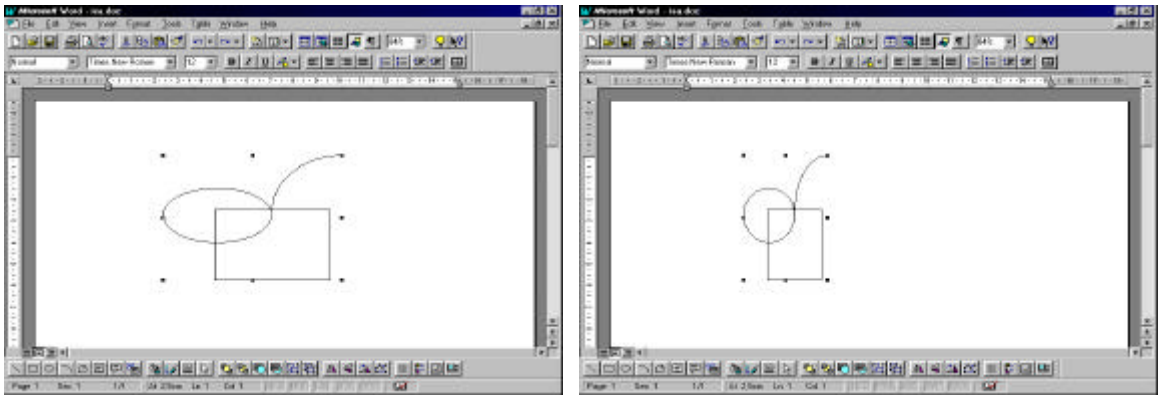
(a)



(b)



(c)



(d) **(e)**

Figura A.12. Alteração de forma no MsDraw

Tabulação dos Dados da Operação de Alteração de Forma no MsDraw

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (seleção) dado (linha)	sinalizador (cursor seta) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação)		
4			varredura	posicionamento
5	estado do dado (apontado)	sinalizador (cursor seta dupla)		
6			indicação	pressão-e-arrasto
7	estado do dado (em alteração)	forma (tracejada)		
8			indicação	soltar botão
9	estado do dado (selecionado)	forma (cheia) + sinalizador (pontos de manipulação)		

Tabela A.14. Dados referentes à operação de alteração de forma no MsDraw

Alteração de Forma de Objeto no AutoCAD

- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção são os mesmos já descritos. Neste aplicativo, mesmo havendo um grupo de primitivas selecionado, apenas aquela cujo ponto de manipulação tenha sido apontado terá sua forma alterada.

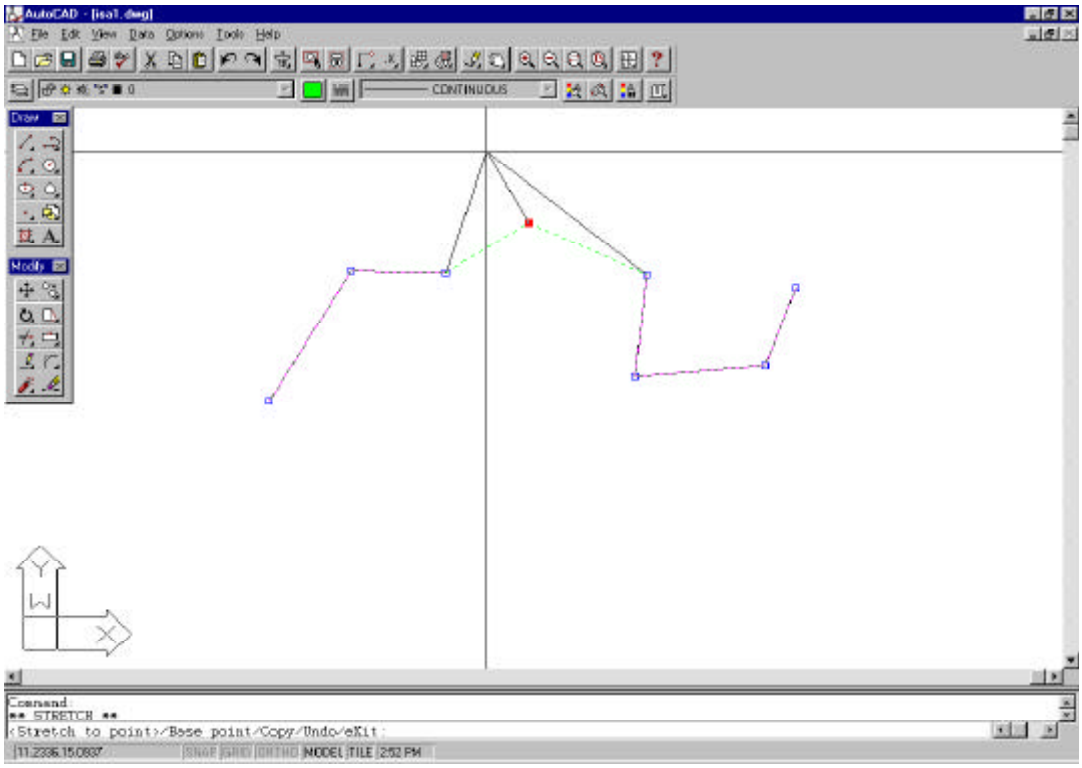


Figura A.13. Alteração de forma no AutoCAD

Por exemplo durante a movimentação do vértice de uma polilinha, Figura A.13, o sistema oferece os novos segmentos da polilinha em tracejado e uma rubber-band ligando o vértice em movimento à posição original.

Tabulação dos Dados da Operação de Alteração de Forma no AutoCAD

	Conteúdo do Sistema	Expressão do Sistema	Conteúdo do Usuário	Expressão do Usuário
1	estado sistema (neutro) dado (linha)	sinalizador (cursor quadradinho com linhas de orientação) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador (pontos de manipulação em azul + traçado tracejado)		
4			varredura	apontamento

5	estado do dado (apontado)	sinalizador (quadradinho do cursor em amarelo)		
6			indicação	clique
7	estado do dado (em alteração)	sinalizador (rubber-band + traçado em cheio + quadradinho do cursor vermelho)		
8			indicação	clique
9	estado do dado (selecionado)	sinalizador (pontos de manipulação + traçado tracejado)		

Tabela A.15. Dados referentes à operação de alteração de forma no AutoCAD

Alteração de Forma de Objeto no MGE

- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção são os mesmos já descritos.
  A alteração pode ser feita através dos pontos de manipulação ou em estado de alteração, por pressão-e-arrasto sobre qualquer ponto da primitiva.

Tabulação dos Dados da Operação de Alteração de Forma no MGE

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	estado sistema (seleção) operação (alteração) dado (linha)	sinalizador (cursor seta com bolinha) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	sinalizador		
4			varredura	apontamento
5	estado do dado (apontado)	sinalizador (/)		
6			indicação	pressão-e-arrasto
7	estado do dado (em alteração)	forma		

Tabela A.16. Dados referentes à operação de alteração de forma no MGE

Alteração de Forma de Objeto no MTool

- Essa operação tem como pré-requisito a seleção dos objetos. Os procedimentos de seleção são os mesmos já descritos.
- De forma semelhante ao desenhador, não é permitido fazer alteração de forma nas primitivas, apenas de escala. Após a seleção da(s) primitivas(s) a operação de escalonamento é solicitada por menu. A dimensão da escala é dada por MD através de três pontos que o sistema obriga sejam sinalizados sobre uma reta. A partir da indicação do segundo ponto (definição da reta) o sistema obriga o caminhamento do cursor de acordo com a reta, sobre a qual ele espera que o usuário assinale o terceiro ponto, que vai definir a escala. Simultaneamente o fator de escala é sinalizado na área de canvas em valores numéricos.
- Neste aplicativo a alteração de forma de uma aresta ou de uma face pode ser simulada através da movimentação de vértices. Porque a representação gráfica baseia-se em um modelo matemático, onde uma aresta é suportada por dois vértices, a alteração na localização de um único vértice vai causar uma alteração de forma na aresta, diferente do escalonamento. Da mesma forma, a alteração na localização de um único vértice, pertencente a uma face, vai gerar uma alteração de forma na face. Este tipo de comportamento é comum nas versões mais recentes de desenhadores, porém tem uma interpretação diferente dos mesmos.

Tabulação dos Dados da Operação de Alteração de Forma no MTool

	Conteúdo do Sistema	**Expressão do Sistema**	**Conteúdo do Usuário**	**Expressão do Usuário**
1	estado sistema (seleção) operação (escala) dado (linha)	sinalizador (cursor seta) menu (ícone) forma (segmento de reta)		
2			indicação	clique
3	estado do dado (selecionado)	cor (vermelha)		
4			ativação	clique
5	estado sistema (entrada de dados)	sinalizador (cursor sinal soma / vermelho)		
6			indicação	clique
7	estado sistema (entrada de dados)	sinalizador (cursor soma) + sinalizador (rubber-band)		
8			indicação	clique
9	estado sistema (complemento entrada de dados escala)	sinalizador (cursor soma) + sinalizador (rubber-band) + sinalizador (pontos escala)		
10			indicação	clique
11	estado sistema (seleção) estado do dado (selecionado)	sinalizador (cursor seta) sinalizador (cor vermelha)		

Tabela A.17. Dados referentes à operação de alteração de forma no MTool

Anexo B

Exemplo de Aplicação da STAG para Projeto de Interfaces

Neste anexo está sendo apresentado um exemplo de aplicação da STAG para projeto de uma interface gráfica. A especificação foi construída usando como referência um sistema Pré-Processador de Elementos Finitos. Neste sistemas o usuário pode interagir, através da representação visual, com base em dois modelos semânticos diferentes, o que nos permite criar um exemplo mais detalhado. O exemplo inclui a especificação de dois contextos paralelos de discurso (CPD): o CPD 1 para edição da geometria do objeto, com base no modelo de engenharia, e o CPD 2 para edição da malha, com base no modelo de elementos finitos. O exemplo considera também diferentes unidades tópicas de discurso (UTD) para o CPD 1.

O exemplo apresentado não é real, não foi construído com base em um sistema específico. Ele foi idealizado de forma a conter alguns elementos de interesse específico. Por exemplo, a expressão da relação de hierarquia entre as primitivas do modelo, através do brilho, não é comum. Foi incorporada a esta especificação como um exemplo.

O exemplo considerou as mesmas operações de edição observadas nos aplicativos comerciais (Anexo A) e analisadas no Capítulo 5. São elas: criação, apagamento, movimentação e alteração de forma.

B.1. Especificação da STAG

CPD 1

Declaração de Tipos e Instâncias

.....

1. Tipos de Dados do Sistema (entidades, atributos e relacionamentos)
    primitiva
    primitiva-interação
    plano-interação
    força
    material
    hierarquia { }
1. Instâncias de Dados do Sistema (entidades, atributos e relacionamentos)
    primitiva: vértice, aresta, face
    primitiva-interação: marca, linha, polilinha, arco, círculo, área
    plano-interação: plano
    força: peso, vento
    material: concreto, ferro
    hierarquia: 1{vértice:0, aresta:1, face:1}

.....

1. Tipos de Operações
    operações-edição
1. Instâncias de Operações
    operações-edição: criar, apagar, mover, deformar

.....

1. Tipos de Variáveis de Estado das Entidades
    estado-seleção
1. Instâncias de Variáveis de Estado das Entidades
    estado-seleção: ativo, inativo

.....

1. Tipos de Variáveis de Estados de Execução do Sistema
    estado-criação
    estado-traçado

1. Instâncias de Variáveis de Estados de Execução do Sistema
    estado-criação: ativo, inativo
    estado-traçado: ativo, inativo

.....

1. Tipos de Variáveis de Estados Correntes do Sistema
    estado-aresta
    estado-deixar-original
1. Instâncias de Variáveis de Estados Correntes do Sistema
    estado-aresta: linha geométrica, fronteira
    estado-deixar-original: ativo, inativo

.....

1. Tipos de Sinalizadores de Apoio
    sinalização-retorno
1. Instâncias de Sinalizadores de Apoio
    sinalização-retorno: indicação-grupo

.....

1. Relações de Atribuição
    primitiva <-- força
    primitiva (face) <-- material
    primitiva (aresta) <-- material

.....

1. Relações de Sobreposição
    primitiva <-- estado-seleção
    primitiva <-- hierarquia

.....

1. Tipos de Expressão do Sistema
    forma
    forma-composta
    cor
    textura
    tom
    brilho
    texto

1. Instâncias de Expressão do Sistema
forma: asterisco, traço
forma-composta: botão # botão-sombreado, quadradinho, bounding-box, cursor-cruz, rubber-band, janela-rolagem
cor: amarelo, azul, vermelho, verde, branco, rosa, cinza, marrom
textura: hachurada, pontilhada
tom: forte, fraco
brilho: alto, normal, baixo
texto: menu-hierárquico, rótulo

.....

1. Tipos do Conteúdo do Usuário
intenção
1. Instâncias do Conteúdo do Usuário
intenção: indicar, indicar-grupo # sinalização-retorno (indicação-grupo), movimentar, ativar, interromper

.....

1. Tipos da Expressão do Usuário
sinal
1. Instâncias da Expressão do Usuário
sinal: clique, clique-duplo, pressão-e-arrasto, clique-botão-direita

.....

Mapeamento

.....

1. Mapeamento de Tipos do Sistema
primitiva <=> forma
primitiva-interação <=> forma-composta <> texto
plano-interação <=> forma-composta <> cor
força <=> cor
material <=> textura
hierarquia <=> brilho
estado-seleção <=> cor <> forma-composta
estado-criação <=> forma-composta
estado-traçado <=> forma-composta
sinalização-retorno <=> forma-composta

.....

.....

1. Mapeamento de Instâncias do Sistema

primitiva (vértice) <=> forma (asterisco)
primitiva (aresta) <=> forma (traço)
primitiva (face) <=> forma (n traços)
primitiva-interação (marca) <=> forma-composta (botão) <> texto (rótulo)
primitiva-interação (linha) <=> forma-composta (botão) <> texto (rótulo)
primitiva-interação (polilinha) <=> forma-composta (botão) <> texto (rótulo)
primitiva-interação (círculo) <=> forma-composta (botão) <> texto (rótulo)
primitiva-interação (arco) <=> forma-composta (botão) <> texto (rótulo)
primitiva-interação (área) <=> forma-composta (botão) <> texto (rótulo)
plano-interação (plano) <=> forma-composta (janela-rolagem) <> cor (branca)
força (peso) <=> cor (verde)
força (vento) <=> cor (azul)
material (concreto) <=> textura (hachurada)
material (ferro) <=> textura (pontilhada)
hierarquia (1{vértice:0, aresta:1, face:1} <=> brilho (1{alto:0, normal:1})
estado-seleção (ativo) <=> cor (amarela) <> forma-composta (quadradinho)
estado-seleção (inativo) <=> nulo
estado-criação (ativo) <==> forma (cursor-cruz)
estado-criação (inativo) <==> nulo
estado-traçado (ativo) <=> forma-composta (rubber-band)
estado-traçado (inativo) <=> nulo
sinalização-retorno (indicação-grupo) <=>
forma-composta (bounding-box)
operação-edição (apagar) <=> texto (menu-hierárquico)

.....

.....
1. Mapeamento de Tipos do Usuário
intenção <=> sinal
.....
1. Mapeamento de Instâncias do Usuário
intenção (indicar) <=> sinal (clique)
intenção (indicar-grupo) <=> sinal (pressão-e-arrasto)
intenção (movimentar) <=> sinal (pressão-e-arrasto)
intenção (ativar) <=> sinal (clique-duplo)
intenção (interromper) <=> sinal (clique-botão-direita)
.....

Dicionário

.....
1. Dicionário de Tarefas Básicas
Criar [Operação = criar, Objeto = aresta]
Mover [Operação = mover, Critério = único]
Mover grupo [Operação = mover, Critério = grupo]
Apagar [Operação = apagar, Critério = único]
.....

Gramática

.....
1. Regras Gramaticais

UTD 1.0
1.0.R1. Criar [Operação = criar, Objeto] -->
selecionar-ferramenta [Objeto] +
gerar-primitiva [Objeto] +
Estado-criação (inativo)

1.0.R2. selecionar-ferramenta [Objeto = aresta] -->
Primitiva-interação (linha)
Intenção (indicar) +
Estado-criação (ativo)
1.0.R3. gerar-primitiva [Objeto = aresta] -->
Plano-traçado (plano) +
Intenção (indicar) +
Estado-traçado (ativo) +
Plano-traçado (plano) +
Intenção (indicar) +
Estado-traçado (inativo) +
Primitiva (aresta) +
Estado-seleção (ligado)

UTD 1.1.
1.1.R.1. Mover [Operação = mover, Critério] -->
selecionar-objeto [Critério] +
Plano-traçado (plano) +
Intenção (movimentar) +
Estado-seleção (desligado)
1.1.R.2. Apagar [Operação = apagar, Critério] -->
selecionar-objeto [Critério] +
Operação-edição (apagar) +
Intenção (indicar)
1.1.R.4. selecionar-objetos [Critério = único] -->
Primitiva ( )
Intenção (indicar) +
Estado-seleção (ligado)

1.1.R.5. selecionar-objetos [Critério = grupo] -->
Primitiva ( )
Intenção (indicar-grupo) +
Estado-seleção (ligado)
.....

CPD 2

Declaração de Tipos e Instâncias

.....

2. Tipos de Dados do Sistema (entidades, atributos e relacionamentos)
    primitiva
    plano-interação
2. Instâncias de Dados do Sistema (entidades, atributos e relacionamentos)
    primitiva: nó, elemento
    plano-interação: plano

.....

2. Tipos de Operações
    operações-edição
2. Instâncias de Operações
    operações-edição: apagar, mover

.....

2. Tipos de Variáveis de Estado das Entidades
    estado-seleção
2. Instâncias de Variáveis de Estado das Entidades
    estado-seleção: ativo, inativo

.....

2. Relações de Sobreposição
    primitiva (nó) <-- estado-seleção

.....

2. Tipos de Expressão do Sistema
    forma
    forma-composta
    cor
    texto
2. Instâncias de Expressão do Sistema
    forma: ponto, traço
    forma-composta: botão # botão-sombreado, janela-rolagem
    cor: amarelo, azul, vermelho, verde, branco, rosa, cinza, marrom
    texto: menu-hierárquico

.....

.....
2. Tipos do Conteúdo do Usuário
intenção
2. Instâncias do Conteúdo do Usuário
intenção: indicar, movimentar, ativar, interromper
.....
2. Tipos da Expressão do Usuário
sinal
2. Instâncias da Expressão do Usuário
sinal: clique, clique-duplo, pressão-e-arrasto, clique-botão-direita
.....

Mapeamento

.....
2. Mapeamento de Tipos do Sistema
primitiva <=> forma
plano-interação <=> forma-composta <> cor
estado-seleção <=> cor
.....
2. Mapeamento de Instâncias do Sistema
primitiva (nó) <=> forma (ponto)
primitiva (elemento) <=> forma (traço)
plano-interação (plano) <=> forma-composta (janela-rolagem) <> cor (branca)
estado-seleção (ativo) <=> cor (amarela)
estado-seleção (inativo) <=> nulo
operação-edição (apagar) <=> texto (menu-hierárquico)
.....
2. Mapeamento de Tipos do Usuário
intenção <=> sinal
.....

.....

2. Mapeamento de Instâncias do Usuário

intenção (indicar) <=> sinal (clique)
intenção (movimentar) <=> sinal (pressão-e-arrasto)
intenção (ativar) <=> sinal (clique-duplo)
intenção (interromper) <=> sinal (clique-botão-direita)

.....

Dicionário

.....

2. Dicionário de Tarefas Básicas

Mover [Operação = mover]
Apagar [Operação = apagar]

.....

Gramática

.....

2. Regras Gramaticais

UTD 2.0

2.0.R.1. Mover [Operação = mover] -->
primitiva (nó) +
Intenção (indicar) +
Estado-seleção (ligado) +
Plano-traçado (plano) +
Intenção (movimentar) +
Estado-seleção (desligado)

2.0.R.2. Apagar [Operação = apagar] -->
primitiva (nó) +
Intenção (indicar) +
Estado-seleção (ligado) +
Operação-edição (apagar) +
Intenção (indicar)

.....

Anexo C

Exemplo de Aplicação da STAG para Análise de Interfaces

Neste anexo está sendo apresentado um exemplo de aplicação da STAG para análise da interface de um sistema já implementado. A operação de cópia será analisada comparativamente nos aplicativos MTool Versão 1.0 [MTool 92] e MTool Versão 1.3 [MTool 97]. Dois aspectos de interesse serão observados. Um deles refere-se ao uso da opção de "leave original" em algumas operações, como por exemplo movimentação, rotação e espelhamento. Através dessa opção, a operação de movimentação é executada na forma de uma cópia seguida de movimentação. O traçado original permanece, enquanto a cópia é movimentada. Outro aspecto de interesse refere-se à alteração de forma. Essa operação não é oferecida pelo sistema, mas pode ser simulada pelo usuário, através de uma operação de movimentação de vértice. Vamos observar de que forma a STAG pode atuar na análise destes casos. Ambos os casos foram citados na Seção 6.4.6 Uso da Notação para Análise da LV.
Antes da operação de movimentação propriamente dita, vamos observar uma questão referente à seleção de primitivas. As primitivas a serem movimentadas devem ser previamente selecionadas. A versão do MTool (92) implementa o estado de seleção em paralelo com o estado de criação de primitivas. Isto significa que, se o sistema está disponível para criação de primitivas, ele não permite selecionar primitivas já traçadas. Os estados de criação e seleção são mutuamente exclusivos. A Figura C.1 esclarece essa situação. No menu à direita da tela, se o usuário assinalar, por exemplo, a opção de criação de linha (Line), o sistema ativará um estado de criação de linha, permitindo que linhas consecutivas sejam criadas, até que esse estado seja interrompido. Esse estado poderá ser interrompido pelo assinalamento do estado de criação de outra primitiva, por exemplo a polilinha (Poly), ou pelo estado de seleção.

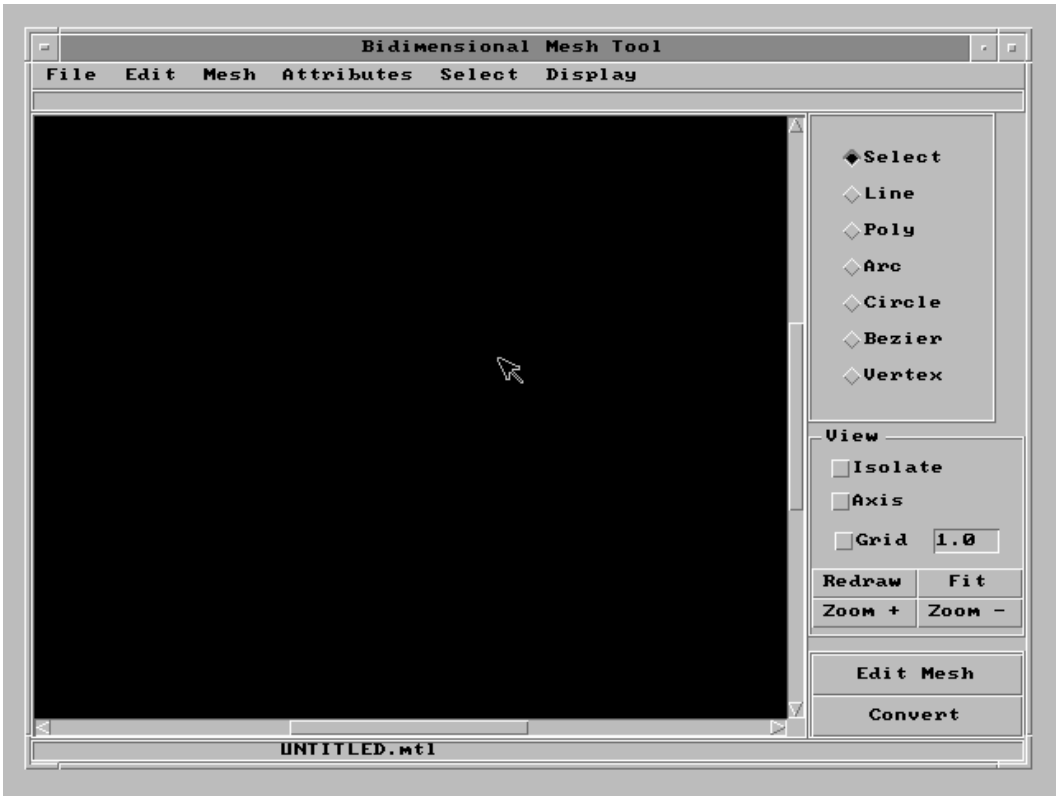


Figura C.1. Tela de abertura do MTool Versão 1.0 [MTool 92]

Nesta forma de implementação existem dois estados de seleção: o estado de seleção do sistema, quando ele permite que a seleção de primitivas seja executada, e o estado de seleção das primitivas, quando estas estão selecionadas. O estado de seleção do sistema é ativado pelo usuário através do menu, sendo sinalizado pelo pequeno botão em reverso. O estado de seleção da primitiva também é ativado pelo usuário, sendo sinalizado por

uma alteração de cor da primitiva (da cor original para vermelha). Para distinguir os dois estados, chamaremos o estado de seleção do sistema de estado-seleção e o estado de seleção da primitiva de estado-primitiva-selecionada. Ambos podem estar ativados ou desativados.
No exemplo foi omitida a numeração de contexto, considerando que a análise é parcial e que este aspecto não é relevante.

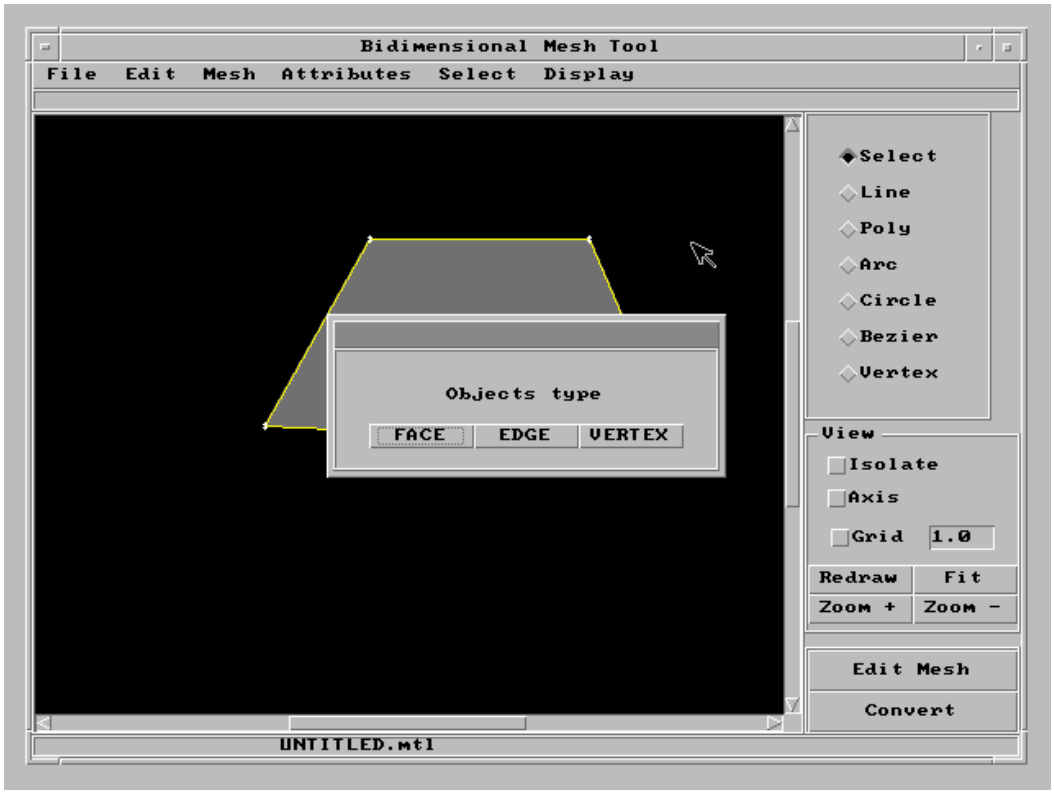


Figura C.2. Opções de primitivas para seleção no MTool Versão 1.0 [MTool 92]

C.1. Especificação da STAG

Declaração de Tipos e Instâncias

.....

Tipos de Dados do Sistema
    primitiva
Instâncias de Dados do Sistema
    primitiva: vértice, aresta, face

.....

Tipos de Operações
    operações-edição
Instâncias de Operações
    operações-edição: mover

.....

Tipos de Variáveis de Estado das Entidades
    estado-primitiva-selecionada
Instâncias de Variáveis de Estado das Entidades
    estado-primitiva-selecionada: ativo, inativo

.....

Tipos de Variáveis de Estados de Execução do Sistema
    estado-seleção
    estado-cópia
Instâncias de Variáveis de Estados de Execução do Sistema
    estado-seleção: ativo, inativo
    estado-cópia: ativo, inativo

.....

Relações de Sobreposição
    primitiva <-- estado-primitiva-selecionada

.....

Tipos de Expressão do Sistema
    forma
    forma-composta
    cor
    texto
Instâncias de Expressão do Sistema
    forma: asterisco, traço
    forma-composta: botão # botão-sombreado
    cor: amarelo, azul, vermelho, verde, branco
    texto: menu-hierárquico, rótulo

.....

Tipos do Conteúdo do Usuário
    intenção
Instâncias do Conteúdo do Usuário
    intenção: indicar, movimentar, ativar, interromper

.....

Tipos de Expressão
    sinal
Instâncias de Expressão Manual
    sinal: clique, clique-duplo, pressão-e-arrasto, clique-botão-direita

.....

Mapeamento

.....

Mapeamento de Tipos do Sistema
    primitiva <=> forma
    estado-primitiva-selecionada <=> cor
    estado-seleção <=> forma-composta <> texto
    estado-cópia <==> forma-composta <> texto

.....

Mapeamento de Instâncias do Sistema
    primitiva (vértice) <=> forma (asterisco)

primitiva (aresta) <=> forma (traço)
primitiva (face) <=> forma (n traços)
estado-primitiva-selecionada (ativo) <=> cor (vermelha)
estado-primitiva-selecionada (inativo) <=> nulo
estado-seleção (ativo) <=> forma-composta (botão#)
estado-seleção (inativo) <=> forma-composta (botão)
estado-cópia (ativo) <==> forma-composta (botão#)
estado-cópia (inativo) <==> forma-composta (botão)
operação-edição (mover) <=> texto (menu-hierárquico "Transformação")

.....

Mapeamento de Tipos do Usuário
intenção <=> sinal

.....

Mapeamento de Instâncias do Usuário
intenção (indicar) <=> sinal (clique)
intenção (movimentar) <=> sinal (pressão-e-arrasto)
intenção (ativar) <=> sinal (clique-duplo)
intenção (interromper) <=> sinal (clique-botão-direita)

.....

Dicionário

.....

Dicionário de Tarefas Básicas
Copiar e Mover [Operação = mover, Critério]]
Mover [Operação = mover, Critério]
Alterar forma [Operação = mover, Critério]

.....

Gramática

.....

Regras Gramaticais
R1. Copiar e Mover [Operação = mover, Critério] -->
selecionar-objeto [Critério] +
Operação-edição (mover) +
Intenção (indicar) +
Estado-cópia (inativo) +
Intenção (indicar) +
Estado-cópia (ativo) +
informação distâncias de movimento +
Intenção (ativar) +
Estado-cópia (inativo)
R2. Mover [Operação = mover, Critério] -->
selecionar-objeto [Critério] +
Operação-edição (mover) +
Intenção (indicar) +
informação distâncias de movimento +
Intenção (ativar)
R3. Alterar forma [Operação = mover, Critério] -->
selecionar-objeto [Critério] +
Operação-edição (mover) +
Intenção (indicar) +
informação distâncias de movimento +
Intenção (ativar)
R4. selecionar-objetos [Critério = único] -->
ativar-estado-seleção +
Primitivas ( ) +
Intenção (indicar) +
Estado-primitiva-selecionada (ativo)
R5. ativar-estado-seleção -->
Estado-seleção (inativo) +

Intenção (indicar) +
Estado-seleção (ativo)

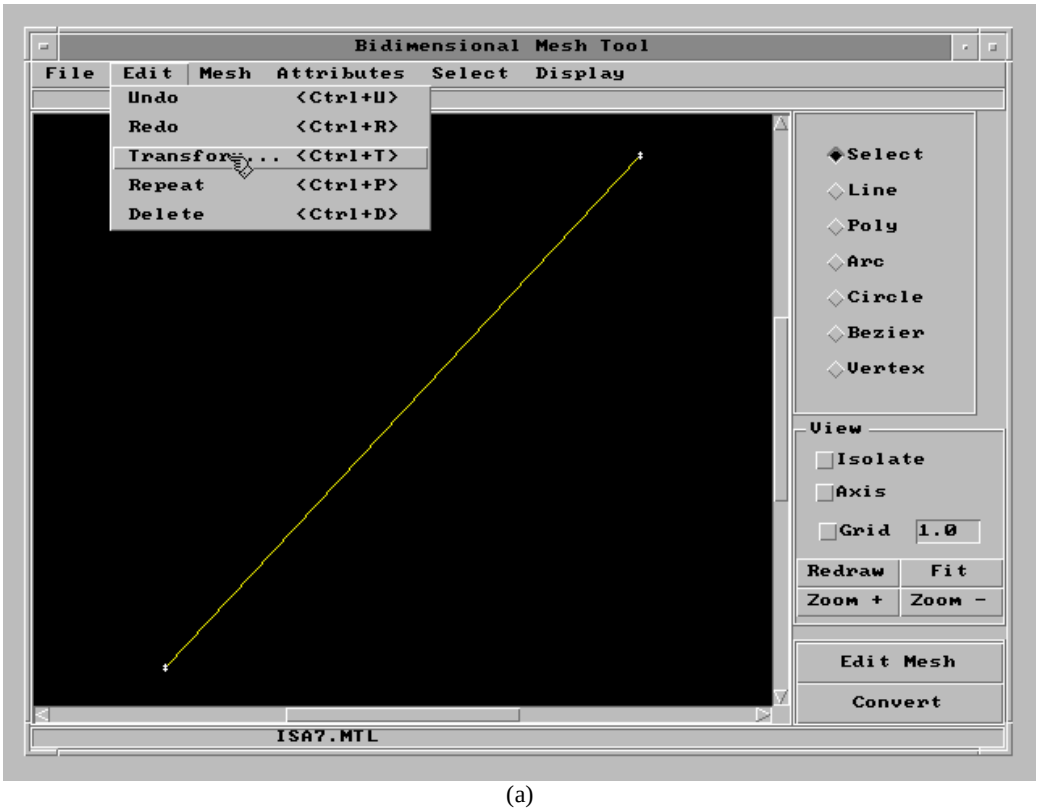
....

C.2. Observações sobre a Especificação da STAG

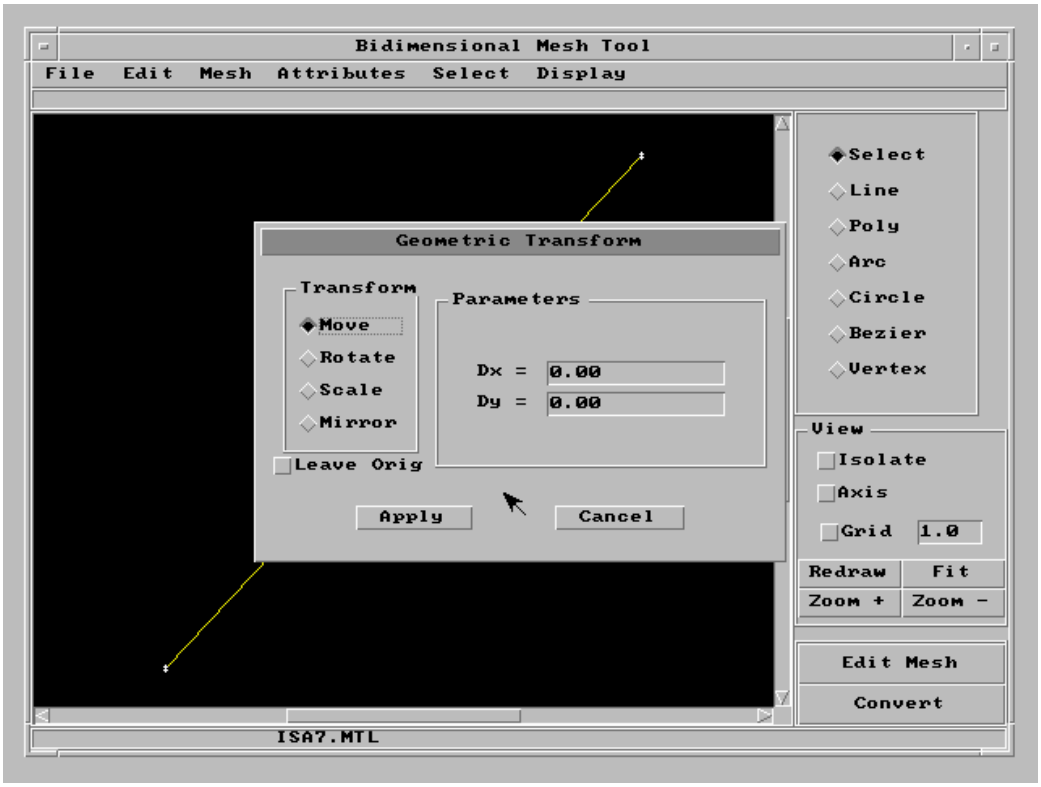
Inicialmente observamos que três intenções diferentes do usuário (copiar e mover, mover, alterar forma) estão sendo mapeadas em uma mesma operação. De acordo com a regra R1, para a tarefa de copiar e mover, primeiro é selecionado o objeto da operação (dado), depois é selecionada a operação a ser executada, depois é alterado o estado do sistema que pode alterar o resultado da operação, depois são fornecidos os dados para movimentação, por fim a operação é ativada. A operação de movimentação não é executada por MD, sendo ativada através do menu. A alteração no estado do sistema, que vai causar a diferença no produto, está sendo tratada após a ativação da operação. Após a execução da operação o estado-cópia do sistema volta automaticamente à situação anterior.

A regra R1 não reflete a estrutura da interface, apresentada nas Figuras C.3 (a) e (b). Quando o usuário escolhe a operação, na verdade ele está clicando em um item de menu Transformações. Este item abrirá uma janela através da qual o usuário irá optar pela operação que deseja, pelo estado do sistema com relação a cópia e pelas coordenadas referentes à alteração a ser executada. Através da janela central da Figura C.4 observamos que não é incompreensível para o usuário o que deve ser feito. Porém, do ponto de vista de tradução semântica, da sua intenção para uma ativação do sistema, a tradução da intenção Copiar e Mover para uma operação de transformação, dentro da qual ele deverá ajustar um estado do sistema, existe um hiato. Além disso, a observação da regra R1 sugere que o estado-cópia permaneceu ativado após a operação, uma vez que o sistema não informa a desativação. Mas isto não é verdadeiro. Após a operação, o estado-cópia é automaticamente desativado. Essa informação não é passada ao usuário porque quando o usuário ativa a operação a janela se fecha, e ele não recebe a informação de retorno do estado-cópia à situação anterior.

Uma forma mais coerente de organizar a linguagem, em termos cognitivos, seria conforme apresentada abaixo, para a regra R1. Nesta nova forma, a ativação do estado-cópia precede a ativação da operação, e o retorno automático do estado-cópia é informado ao usuário. Porém, para que essa estrutura gramatical fosse implementada seriam necessárias alterações na estrutura da interface.



(a)



(b)

Figura C.3. Operação de movimentação de objeto no MTool (92)

R1. Copiar e Mover [Operação = mover, Critério] -->
Estado-cópia (inativo) +
Intenção (indicar) +
Estado-cópia (ativo) +
selecionar-objeto [Critério] +
Operação-edição (mover) +
Intenção (indicar) +
informação distâncias de movimento +
Estado-cópia (inativo)

Outro aspecto que pode ser observado neste exemplo refere-se à alteração de forma. A alteração de forma é executada através da movimentação de um vértice. A operação de movimentação pode ser feita sobre qualquer primitiva. No entanto, a alteração de forma é feita através do movimento sobre os vértices. A notação terá dificuldade em acomodar essa operação.

Por tratar apenas com conteúdo a notação não reflete a composição visual da interface, Figura C.3. Embora em uma situação de análise esta forma possa parecer problemática, ela é bastante positiva, porque o projetista pode se concentrar no conteúdo da comunicação. Ele pode até recorrer à forma expressiva da interface para compreender certos aspectos da LV implementada, mas ele deve analisar a coerência do conteúdo sem preocupação com a forma. Uma vez encontrada a coerência do conteúdo ele pode buscar a melhor forma, tanto em termos de recursos expressivos utilizados, quanto em termos de estruturação das estruturas de menus e até mesmo de composição visual da interface.

Na versão mais atualizada do aplicativo MTool (97) a movimentação é executada por MD. Nela, por força das circunstâncias, o projetista foi forçado a outra solução, no que se refere à opção de leave-original. Neste caso, como a movimentação é feita por MD, o usuário solicita a operação de movimentação através do menu e, em seguida, assinala dois pontos na área de canvas, traçando um segmento de reta, cuja dimensão e direção, darão os parâmetros necessários à movimentação. Nesta versão a opção de leave-original foi implementada como um estado do sistema, que pode ser ativado a qualquer momento e que permanece ativado, independente de quais operações sejam executadas. Para esta versão a regra R1 teria o formato conforme apresentado abaixo.

R1. Copiar e Mover [Operação = mover, Critério] -->
Estado-cópia (inativo) +
Intenção (indicar) +
Estado-cópia (ativo) +
selecionar-objeto [Critério] +
Operação-edição (mover) +
Intenção (indicar) +
Plano-interação (canvas) +
Intenção (indicar) +
Plano-interação (canvas) +
Intenção (indicar)

Sob o aspecto cognitivo, nesta versão a tradução entre a intenção do usuário e as ações ativadas está mais coerente. Porém observando a Figura C.4 veremos que as operações de edição e a operação de mudança de estado do sistema estão agrupadas em um mesmo nível de menu, quando são ações de natureza diferente. Conforme a declaração de tipos e instâncias da notação, os primeiros itens do menu referem-se a operações de edição, enquanto o item "leave original" refere-se a uma variável de estado do sistema. Embora a instância dessa variável afete o produto das operações de edição, seria adequado um tratamento diferenciado que esclarecesse o modelo funcional ao usuário. Em ambos os casos podemos entender que ocorreram soluções ocasionais, que poderiam ter sido melhor ajustadas com base no uso da notação.

Sob o aspecto cognitivo, nesta versão a tradução entre a intenção do usuário e as ações ativadas está mais coerente. Porém observando a Figura C.4 veremos que as operações de edição e a operação de mudança de estado do sistema estão agrupadas em um mesmo nível de menu, quando são ações de natureza diferente. Conforme a declaração de tipos e instâncias da notação, os primeiros itens do menu referem-se a operações de edição, enquanto o item leave original refere-se a uma variável de estado do sistema. Embora a instância dessa variável afete o produto das operações de edição, seria adequado um tratamento diferenciado que esclarecesse o modelo funcional ao usuário. Em ambos os casos podemos entender que ocorreram soluções ocasionais, que poderiam ter sido melhor ajustadas com base no uso da notação.

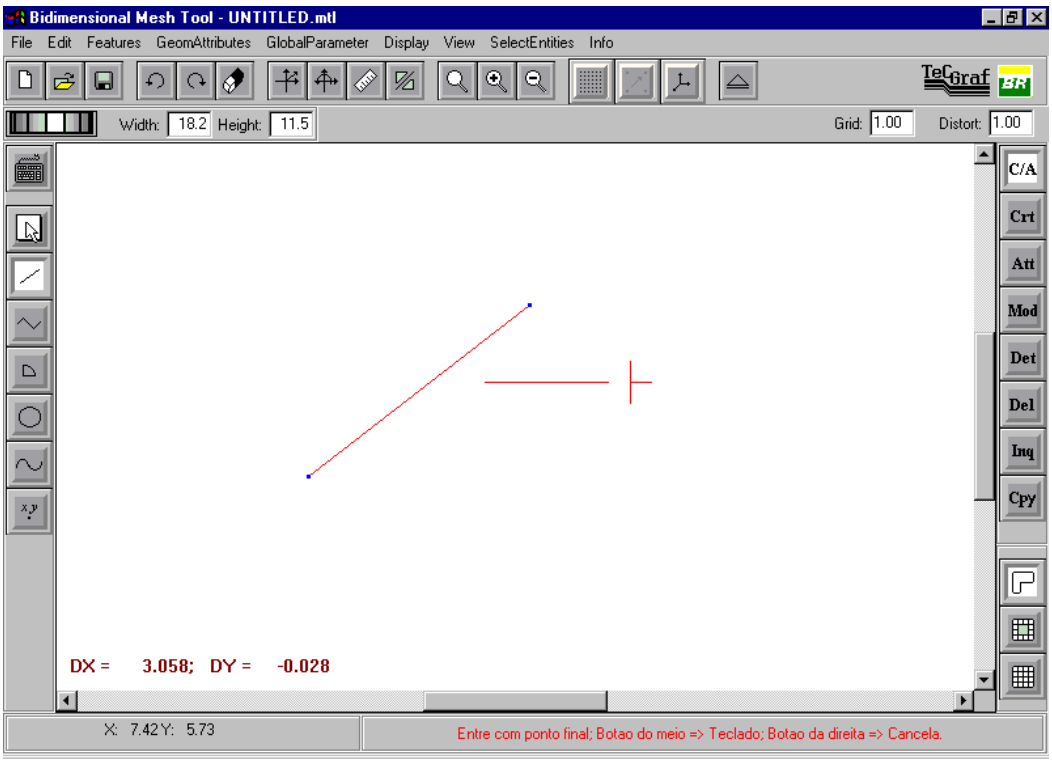


Figura C.4. Operação de movimentação de objeto no MTool (97)