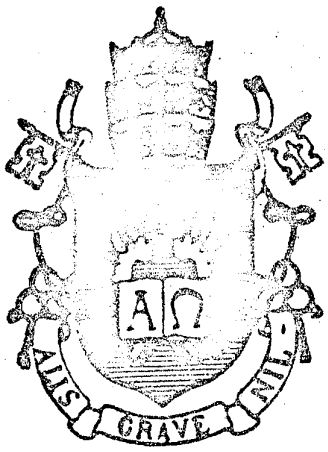

PUC

Série: Monografias em Ciência da Computação,
No. 15/91

INTERAÇÃO COM SISTEMAS DE INFORMAÇÃO BASEADOS EM CONHECIMENTO

Maria Tereza M. P. Silva

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP-22453
RIO DE JANEIRO - BRASIL

PUC RIO - DEPARTAMENTO DE INFORMÁTICA
Série: Monografias em Ciência da Computação, No. 15/91
Editor: Carlos J. P. Lucena Julho, 1991

INTERAÇÃO COM SISTEMAS DE INFORMAÇÃO BASEADOS EM CONHECIMENTO *

Maria Tereza M. P. Silva

* Trabalho patrocinado pela Secretaria de Ciência e Tecnologia da Presidência da República.

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC Rio - Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453 - Rio de Janeiro, RJ
Brasil

Tel.:(021)529-9386 Telex:31078 Fax:(021)511-5645
E-mail:rosane@inf.puc-rio.br

Resumo

Este artigo fornece uma análise crítica das tendências que vêm apresentando os ambientes de desenvolvimento de aplicações, inclusive no que se refere a Sistemas de Informação Baseados em Conhecimento (SIBC). São analisados ganhos e perdas em várias opções de projeto, à luz dos atuais conceitos de projeto centrado no usuário. É enfatizada a interdependência entre a estrutura interna do sistema e a interface com o usuário. A Linguagem Natural (LN) é advogada como meta-linguagem ideal para integração de sistemas em ambiente de trabalho.

Palavras-Chave

ambiente, desenvolvimento, usuário, sistema, interface, linguagem

Abstract

This paper gives a critic analysis of the ongoing tendencies on application development environments, including those of Knowledge Based Information Systems (KBIS). It analyses trade-offs among design options, in the light of the present concepts of user centered design. The dependencies between the internal structure and the user interface are emphasized. Natural Language is advocated to be the ideal meta-language for system integration in the working environment.

Keywords

environment, development, user, system, interface, language

Sumário

apontadas algumas dificuldades.

Finalmente, na seção 7 teçemos algumas considerações sobre o papel da interface como agente de integração do sistema no ambiente de trabalho.

2 - Sistemas Centrados no Usuário

Muito se tem analisado a questão de sistemas centrados no usuário ultimamente [Norman 86]. Para esclarecer bem o que seja centrar o projeto de um sistema no usuário, do ponto de vista que nos interessa neste trabalho, vamos fazer algumas considerações sobre a responsabilidade final de uma tarefa a ser executada através de um sistema de computação.

Em primeiro lugar, argumentamos que, uma vez que a parcela de trabalho do usuário não foi embutida no sistema, então pelo menos alguma responsabilidade é do usuário. O que acontece é que usuário e sistema participam da execução de uma tarefa. Vamos ver como poderíamos discernir que parte da tarefa é de responsabilidade e julgamento do usuário e qual a parte que compete ao sistema.

Existe sempre um nível de diagnóstico que o usuário deve fazer a respeito do sistema. No mínimo ele é quem decide se o sistema está disponível para ser usado ou não o está. No máximo

1 - Introdução

Este artigo é uma crítica às tendências divergentes dos projetos de ambientes de desenvolvimento de aplicações, indicando as dificuldades geradas para a idealização de sistemas centrados no usuário.

Na seção 2 são apresentados alguns conceitos atuais em que deveriam se basear os projetos de sistemas, enfatizando as características preconizadas para as interfaces dos sistemas com o usuário.

Na seção 3 são apreciados três tipos de sistemas: aplicativos, linguagens de programação e aplicações para o usuário final. É analisada a interação desses sistemas com vários tipos de usuário.

Na seção 4 são analisadas possíveis ligações entre a pobreza de estrutura de um sistema, a opacidade da interface e a vida útil do sistema.

Na seção 5 é abordado o problema da interface para um SIBC, mostrando as dificuldades devidas ao alto grau de complexidade deste tipo de sistema.

Na seção 6 a linguagem natural é indicada como instrumento geral de revelação de sistemas, sendo entretanto

ainda vai ser o usuário que vai verificar se um sistema está executando bem a parte que lhe cabe na tarefa.

Já o sistema não diagnostica o usuário. O que um sistema pode fazer é apenas classificar o usuário em algum estereótipo predefinido pelo idealizador do sistema. A finalidade desta classificação é a de garantir um mínimo de possibilidade de comunicação dos parâmetros necessários para a participação do sistema na execução da tarefa. No máximo seria a de permitir um acoplamento estrutural ótimo entre as duas partes, para que a tarefa seja executada da forma mais eficiente possível [Booth 90].

Uma interface para um sistema cuja idealização é centrada no usuário deve permitir ao usuário níveis de diagnóstico que satisfaçam as condições que avalizam a parte da tarefa que é de responsabilidade do usuário. Ou seja, no mínimo a interface deve revelar ao usuário as características do comportamento do sistema com respeito ao foco de interesse do usuário.

Entretanto, do ponto de vista da comunicação de parâmetros entre o usuário e o sistema, a revelação das características do comportamento do sistema tem uma finalidade mais ampla. O objetivo aqui é mais do que possibilitar a comunicação: é de tentar garantir uma comunicação eficaz e eficiente. Isto é, promover a integração das partes envolvidas, usuário e sistema, na execução da tarefa [Booth 90].

A razão desta preocupação de integrar o sistema e o usuário deriva de alguns fatos. Em primeiro lugar temos que afirmar que o sistema não é exatamente um andróide. O seu padrão de comunicação fica limitado pelas características físicas do computador hospedeiro e, também, pelas características da sua própria estrutura interna. Este fato acarreta certas dificuldades de integração de um sistema num ambiente de trabalho impregnado de interações entre seres humanos [Norman 86].

Podemos contornar estas limitações ou atuando sobre as características físicas do computador hospedeiro, ou sobre as características estruturais do próprio sistema. Podemos incluir, por exemplo, na idealização do sistema, componentes que tratem cada falha gerada pelas limitações existentes. Uma das maneiras de tratar estas falhas consiste justamente na exposição das limitações candidatas a determinantes da falha ocorrida. Isto é o que ocorre num diálogo normal, entre duas pessoas. Enquanto é o sistema que tem que se explicar, é preciso que ele saiba diagnosticar as causas destas falhas. Para isto ele deveria conter uma base de conhecimento sobre si próprio, a ser utilizada nesta revelação ao usuário.

Ainda analisando estas falhas na comunicação entre o usuário e o sistema, temos um outro lado da questão. A falha pode ter sido gerada por alguma limitação do usuário. O usuário teria então que se explicar para o sistema. Este pode ser um ponto mais

angustiante para o usuário. Este tipo de falha pode advir justamente do não enquadramento do usuário em nenhum dos modelos esperados pelo sistema, de acordo com a especificação do projetista. Neste caso seria interessante que o sistema percebesse este fato e levasse então o diálogo para um nível de meta-linguagem onde o usuário poderia encontrar meios de assumir um dos comportamentos esperados.

O sistema é um resultado de uma interpretação ontológica do mundo, particular do projetista do sistema, existente no momento da idealização do sistema. A visão do mundo por parte de um usuário qualquer pode ser inteiramente diferente daquela que foi utilizada pelo projetista idealizador do sistema. E, pior do que isto, o usuário pode não saber exatamente qual é esta visão do mundo que foi utilizada na concepção do sistema que ele pretende ou pretendia utilizar.

Vamos voltar para a analogia de um diálogo para tratar a interação [Booth 90]. Os diálogos entre ser humano e computador acontecem em contextos pre-determinados pelo projetista do sistema. A dinâmica do diálogo gira em torno da execução de alguma tarefa pre-concebida pelo projetista do sistema e a intenção é obter algum resultado que o projetista tem para si como possível e de interesse em determinados contextos. Dentro destas limitações, podemos imaginar que este diálogo surge de uma necessidade do usuário e tem um objetivo suficientemente claro dentro de um contexto claramente definido.

Vamos supor como modelo para este diálogo um interesse em estabelecer uma cooperação mútua para suprir uma necessidade diagnosticada pelo usuário. Este padrão de cooperação mútua implica em mútua confiança e objetividade no diálogo. Temos que um diálogo se desenvolve em torno de objetos cuja concepção é comum para as partes. Enquanto as referências se mantêm desconhecidas ou ambíguas para alguma parte, o diálogo não atinge a finalidade a que se propõe.

Então, no mínimo o sistema precisa fixar claramente as referências que ele próprio utiliza para os objetos do contexto da tarefa, revelando-as ao usuário sempre que o diagnóstico de uma falha na comunicação indicá-las como possível causa.

O ideal seria que o sistema tentasse também resolver as referências do usuário aos objetos do contexto da tarefa. Desta forma o sistema não obrigaria o usuário a usar apenas aquelas referências a objetos preconcebidas no projeto do sistema. Para isto ele deveria conter diversas interpretações do mundo conforme as características típicas dos usuários esperados.

Na próxima seção discutimos essa questão das diversas interpretações adotadas pelos projetistas para a idealização de sistemas dos tres tipos seguintes: aplicativos, linguagens de programação e aplicações para o usuário final.

3 - Conflitos Gerados por Interpretações

O desenvolvimento de aplicações para o usuário final vem evoluindo para um modelo cada vez mais flexível [Booth 90]. A prototipagem e o desenvolvimento parcelado, sujeito a modificações ao longo do processo de desenvolvimento, vem se tornando uma prática corrente.

Principalmente com o advento dos micro-computadopres, surgiu uma linha de produtos extremamente flexíveis, abrangendo características desejáveis em um grande número de contextos de aplicação. Um aplicativo, por exemplo, pode ser visto ou como uma fábrica de produtos, ou como um produto semi-acabado.

Um editor de textos, por exemplo, seria uma máquina de preparar textos formatados. Um pacote de criação e acesso a bases de dados poderia ser considerado da mesma forma, na medida em que pudesse ser usado diretamente para manipulação de dados e arquivos. Este mesmo aplicativo poderia ser considerado como semi-acabado, uma vez que pudesse ser utilizado como estrutura base para o desenvolvimento de aplicações para usuários finais. Uma planilha eletrônica também poderia ter esses dois tipos de utilização.

Um ambiente para sistemas especialistas, embora a primeira vista possa parecer da mesma natureza, é muito mais um semi-acabado. Isto acontece porque o nível de complexidade dos parâmetros a serem passados para o sistema costuma ser maior do

que o suportável por um especialista leigo em computação. A utilização deste tipo de aplicativo necessita, em geral, da participação de um engenheiro de conhecimento. Este, sim, está geralmente adaptado às limitações de um computador e acostumado a representar parâmetros complexos nestas máquinas ingênuas.

Temos a observar que ao se substituir um aplicativo na sua maneira direta de uso por uma aplicação de linha,via de regra se introduz uma barreira entre o usuário e o sistema. A revelação da estrutura do sistema, que vinha embutida na interface do aplicativo, fica escondida, muitas vezes completamente. Poderíamos argumentar que o usuário final não está interessado em uma revelação tão profunda. Mas um usuário mais esperto, descontente com alguma característica da aplicação a ele oferecida, pode se animar a construir uma nova aplicação a partir da mesma estrutura. Em alguns casos, a presença destes aplicativos nos ambientes de trabalho acaba formando programadores e analistas. Por outro lado, aquelas pessoas não muito entusiasmadas pelo uso de computadores não se sentem muito atraídas para investigar este tipo de engenho e preferem usar parcimoniosamente alguma aplicação cuja informação seja realmente valiosa.

Talvez não seja um exagero afirmar que as interfaces destes aplicativos foram imaginadas para um tipo de usuário que tem perfil de programador, embora não o seja. Não sendo o usuário deste tipo, mesmo que se familiarize e use certas funções do

aplicativo, não vai investigar a capacidade real do sistema e não vai obter dele toda a vantagem que poderia. Por outro lado, um programador propriamente dito tira vantagens incríveis de um aplicativo, utilizando todo o seu potencial, inclusive por meio do uso de linguagens de programação apropriadas para isto. Ele não só aproveita a revelação natural embutida no aplicativo, como investiga e estuda todas as suas características até a profundidade máxima: o computador e seus dispositivos.

Conforme observado, nos ambientes de trabalho onde são introduzidos sistemas de computação, aparecem e se desenvolvem programadores amadores. Explorando este fato, surgiram algumas liguagens de programação projetadas para serem usadas diretamente pelo usuário final. Se fizermos uma avaliação levando em conta todos os usuários, vamos concluir que a iniciativa não foi bem sucedida. Mas temos que lembrar duas condicionantes: uma é que não costuma ser do interesse do pessoal da área de computação a proliferação de "concorrentes" no usuário final, o que se manifesta através de um certo boicote a esta iniciativa; outra é de que o sucesso talvez devesse ser medido em função daquela minoria que tem mentalidade adequada.

Uma coisa que vem acontecendo muito é a utilização, no desenvolvimento de aplicações pelos programadores, de linguagens de programação sem revelação em níveis mais profundos, do tipo proposto para o usuário final. Neste caso temos uma distorsão do modelo do usuário e isto pode tornar as coisas difíceis. A linguagem de altíssimo nível em geral é muito opaca para o que

convém num ambiente de desenvolvimento de aplicações. Mais do que isto, ela costuma ter uma certa rigidez nas suas formas de interação com o usuário, o que acaba dificultando a tarefa do programador. Às vezes, a estrutura ou o padrão de processamento embutido na linguagem impede soluções mais flexíveis para a estrutura e a interface do sistema. Ou seja, não se trata de linguagens propriamente centradas em usuário programador.

Temos também, é claro, linguagens centradas no programador propriamente dito. Este tipo de linguagem costuma permitir ao usuário alterar a sua estrutura, ligando seus módulos a objetos completamente concebidos pelo programador.

Na próxima seção falaremos sobre as consequências destas estruturas rígidas largamente utilizadas e da opacidade dos sistemas em geral.

4 - Estruturas Rígidas e Obsolescência

As linguagens de nível muito alto costumam apresentar uma certa rigidez. Algumas soluções alternativas já estão prontas, mas muitas outras soluções ficam proibidas.

O projetista, ao idealizar o sistema conforme a sua interpretação do mundo, costuma definir as funções do sistema com

uma liberdade considerável. Entretanto, ao optar pela linguagem de altíssimo nível, ele se vê obrigado a contornar inúmeros obstáculos, devido justamente a ter que utilizar aquelas soluções já previamente definidas no ambiente de desenvolvimento. O que acaba acontecendo é um esvaziamento do projeto. Os padrões do ambiente acabam se transformando em camisas de força e o projetista acaba optando pela mediocridade. A estrutura do ambiente da linguagem que foi idealizada para o usuário normal acaba gerando projetistas medíocres que vão comprometer a qualidade das aplicações para o usuário final.

Surge a grande moda da automatização. As tarefas são simplesmente automatizadas: a maneira de executá-las é previamente definida e fica encrustada no sistema de um modo tal que o usuário fica realmente restrito e tolhido de qualquer iniciativa que lhe seria natural.

Ocorre então a coisa mais natural: o mundo muda. As tarefas se modificam para se adaptar a novas condicionantes. E o sistema? Que estrutura ele tem para sobreviver a essas mudanças? E o usuário? Que revelação ele tem do sistema para adaptar a sua utilização a essas novas situações? Aqui é onde o barato sai caro. Que fazer? Alterar o sistema? Desenvolver novas funções? Sai caro!

Um sistema que refletisse os objetos do contexto da tarefa do usuário na sua estrutura interna, e revelasse ao usuário estas suas características de tal forma que este pudesse

associar estes objetos com um certo grau de liberdade, propiciaria uma adaptabilidade maior à dinâmica do ambiente. Provavelmente vale a pena investir neste tipo de projeto mais flexível e numa interface que inclua uma meta-linguagem para revelar ao usuário estas características [Atkinson 89].

Na próxima seção tratamos do caso de estruturas mais complexas e seus problemas de revelação.

5 - Complexidade de Estrutura e Revelação

A necessidade de munir o sistema da capacidade de se explicar para o usuário fica cada vez mais imperativa na medida em que os sistemas englobam processamentos mais sofisticados. Os sistemas especialistas, por exemplo, que costumam tratar problemas segundo cálculos e heurísticas muito particulares, dificilmente ganhariam a confiança dos usuários se não revelassem as bases de raciocínio e de cálculo utilizadas para calcular as suas respostas. Aqui o problema é: que bases são estas que precisam ser reveladas? Seria apenas a revelação do caminho percorrido por entre as regras de produção? Isto é o que a maioria dos sistemas costuma fazer. Mas talvez o importante não seja dizer como foram utilizadas as regras, mas a que vêm estas regras. Ou seja, talvez fosse melhor revelar qual é o modelo de abordagem do problema utilizado e como se encaixa no modelo o caso específico analisado. Este seria um meta-conhecimento a ser revelado ao usuário.

Um tipo de aplicação de uso crescente na atualidade é o Sistema de Informação Baseado em Conhecimento [Parsaye 88]. Este conhecimento se refere exatamente ao contexto da tarefa do usuário. Por mais simples que possa parecer uma tarefa, pode existir um certo grau de especialidade passível de ser embutido no sistema. A concretização disto gera uma base de conhecimentos com características mais simples que aquelas típicas dos sistema especialistas. Assim, a dificuldade de explicar as bases de raciocínio pode ainda ser grande mas, certamente, a capacidade do usuário entender do que se trata aumenta muito. Por causa disto há uma tendência grande para formalizar a correspondência entre o corpo de regras e os elementos do domínio do contexto do usuário. Isto feito, a explicação do raciocínio através das regras pode se referenciar diretamente ao contexto do usuário.

Existem dois tipos de interação no caso de Bases de Conhecimento:

(1) com o especialista ou o engenheiro de conhecimento, quando da representação do conhecimento na base;

(2) com o usuário, não especialista, quando da necessidade de obter uma posição de um especialista para o caso específico que precisa ser resolvido.

O caso (2) se assemelha muito ao caso de consulta a uma

base de dados. A grande diferença é o nível de revelação das estruturas das regras que levam a indicar cada possibilidade de solução apresentada [Parsaye 88].

O caso (1) se refere a um diálogo de mais "baixo nível" _ mais próximo ao formalismo. A interação do usuário é mais ambiciosa _ representar um conhecimento. Na prática, o sistema não é autosuficiente para este diálogo e usa-se um interlocutor _ o engenheiro de conhecimento. Este interlocutor tem a habilidade de conduzir um diálogo bastante desnivelado com o sistema, sendo, em geral, um profundo conhecedor de computação.

De toda a maneira, tanto no caso (1) como no caso (2), o que se objetiva na idealização da interface é integrar o sistema no ambiente de trabalho.

Aliás, este é o objetivo perseguido no caso de Bases de Dados também [Freundlich 90]. Ao se tratar de Bases de Conhecimento, pretende-se ampliar o potencial do sistema de forma a que ele venha a ser mais útil no trabalho. Mesmo que não se incluam novas regras no SIBC, que se use apenas as antigas regras antes embutidas nos programas de aplicação, o fato de elas estarem agora explícitas e poderem ser mostradas ao usuário como justificativa de respostas já eleva o nível de utilidade do sistema, em tese. Por outro lado, a absorção desta intenção nova do usuário, de explorar estas novas potencialidades do sistema, enfatiza algumas características desejáveis numa interface para

usuários de SIBC.

A pressuposição na idealização de uma interface para um SIBC é a de que existe uma situação que precisa ser descrita para o sistema, e que esta situação é candidata a um tratamento heurístico, típico de um especialista no assunto de referência daquele ambiente de trabalho.

Na verdade, a introdução de um SIBC numa rotina de trabalho modifica as relações básicas no contexto do usuário [Brown 86] [Winograd 86]. E mais, a própria visão que o usuário tinha a respeito de sua tarefa específica.

Identificamos uma primeira fase na interação do usuário com o sistema: a verificação da pertinência da situação a algum caso previsto na construção do sistema. Para isto o sistema contém uma representação interna para os casos possíveis. Por exemplo: locais definidos dentro de quadros de referência ("frames" e "slots"), que devem ser preenchidos conforme as informações fornecidas pelo usuário.

Normalmente, o diálogo para esta fase da interação se resume a uma série de perguntas propostas pelo sistema, que devem ser respondidas pelo usuário, ou uma tela preformatada a ser preenchida pelo usuário. Para verificar e/ou complementar estas informações, o sistema pode ir à base de dados durante o diálogo.

Concomitantemente ao preenchimento, o sistema pode ir

tentando julgar a pertinência da questão. Feito este julgamento da aplicabilidade das heurísticas embutidas no SIBC ao caso apresentado, pode acontecer uma resposta vazia. Ou seja, a declaração de que o caso não se enquadra em nenhuma das espectativas do sistema. Podemos interpretar esta situação de duas maneiras: ou o sistema não tem a abrangência necessária, ou o usuário está supondo uma possibilidade que não existe realmente. Como, do ponto de vista do sistema, não é possível assegurar qual é o caso, a melhor atitude de resposta no diálogo da interface é a de revelação, ou seja, mostrar ao usuário as possibilidades e limitações do sistema. Este tipo de resposta auxilia na integração do sistema no ambiente de trabalho, na medida em que o usuário passaria a reconhecer melhor a capacidade do sistema.

A segunda fase seria a de aplicação de heurísticas. Aqui realmente reside um certo mistério. Tanto que, na construção de sistemas especialistas, sempre é considerada importante a capacidade do sistema responder o porque da indicação das soluções por ele apresentadas.

As razões disto são: em primeiro lugar porque pode ser que o sistema faculte ao usuário a escolha de uma entre várias soluções, induzindo-o a um processo decisório; em segundo lugar porque, focalizando mais uma vez a integração do SIBC no ambiente de trabalho, na medida em que o sistema consegue transmitir ao usuário o seu "raciocínio" para chegar à solução, ele vai ganhando maior credibilidade do usuário, o qual vai ficar

disposto a consultar o sistema com maior frequência.

Fica evidente a importância da estrutura do sistema, em particular do SIBC, para que se consiga uma revelação positiva na interação com o usuário. Uma estrutura ambígua, informal ou muito rígida, ao ser revelada para o usuário, acaba apontando defeitos no sistema e criando um clima negativo para um diálogo objetivo.

Na próxima seção contemplamos o uso da linguagem natural para revelação de sistemas, considerando os diversos tipos de sistemas e os casos mais críticos.

6 - Linguagem Natural e Revelação

Podemos discorrer sobre a linguagem natural contemplando-a sob dois aspectos: como tradicional forma de representação do conhecimento e como base comum de comunicação para a ação como atos de fala.

A primeira característica, consolidada desde a antiguidade nas inscrições em pedras, nos papiros e nos coromandéis, garante uma posição bem fundamentada para a LN como verdadeira forma de representação de todo e qualquer conhecimento. Entretanto, parece que a LN não é autosuficiente nesta área. É comum ser feita a afirmativa de que uma peça arqueológica pode transmitir muito mais informação do que uma

simples inscrição, ou que o que vale é a combinação entre as inscrições e os objetos.

Isto poderia ser explicado pelo fato de que a LN é uma representação de uma cultura e não a descrição completa de uma cultura. Embora de valor incontestável, a LN não é autosuficiente para esta finalidade.

Fazendo um paralelo, a LN, embora rica em expressividade, talvez não resolva completamente a questão da revelação do sistema para o usuário. Provavelmente o usuário é levado a criar um modelo abstrato do sistema, na medida em que lê a respeito dele. Mas fica faltando algum tipo de gancho ou aterramento para que seja estabelecida uma correlação entre as características do modelo abstrato e as funções reais do sistema. Na prática esta ligação é feita na medida em que o usuário se habitua a usar efetivamente o sistema.

Podemos afirmar que quanto mais clara e formalizada nos seus diversos níveis de abstração for a estrutura do sistema, mais nítida fica a correspondência entre o modelo mental que o usuário vai montando a respeito do sistema e o que o sistema faz realmente.

Falamos aqui em níveis de abstração, nos referindo justamente aos diferentes focos de interesse do usuário. Voltando aos três tipos de sistema analisados nas seções anteriores, vemos a LN ocupando a mesma posição de destaque em qualquer caso. Seja

o usuário um leigo em computação ou um programador de sistemas, ele está acostumado a interpretar LN no seu dia a dia. O que varia é o enfoque dado por cada tipo de usuário e a preconcepção do sistema. Um programador espera grandes limitações por parte de um sistema e consegue compreender rapidamente muitas delas. A sua linguagem de referência é consolidada nas publicações e no meio técnico de copmputação em geral. Já o usuário final é um forte candidato a experiências de alta perplexidade e não sabe, em geral, interpretar termos consagrados na área de computação.

Além destas diferenças de profundidade na área técnica, temos a influência de outros tipos de diferenças advindas de aspectos culturais. Em primeiro lugar não podemos deixar de lado o fato de que a LN existe em vários idiomas diferentes, que são sistemas de representação realmente distintos do ponto de vista léxico, sintático, semântico e cultural [Maximiliano c. 24]. As condicionantes dos ambientes de sobrevivência dos indivíduos imprimem características particulares às interpretações da LN em cada idioma, atribuindo às vezes referências diferentes à mesma palavra num mesmo idioma ou dissolvendo a correspondência entre idiomas diferentes. O comportamento social dos indivíduos também cria expressões que só fazem sentido para os componentes de um cérto segmento da sociedade e nem sempre tem representação em outros contextos.

Portanto, no caso de SIBC, considerando a primeira fase indicada, vamos advogar o uso de menu's e telas preformatadas.

Isto porque a finalidade é minimizar a possibilidade do usuário descrever situações que o sistema não abrange. Se tivéssemos aqui um diálogo em linguagem natural, por exemplo, o usuário poderia facilmente começar a formular questões não pertinentes. Isto obrigaria o sistema a saber conduzir o diálogo para o assunto de referência, revelando o seu despreparo numa série de áreas de conhecimento que podem parecer triviais para o usuário. Ou seja, o diálogo passaria para o usuário uma imagem negativa do sistema.

Considerando agora a segunda fase indicada, vamos advogar justamente o uso da linguagem natural para explicar o raciocínio empregado pelo SIBC. Em primeiro lugar porque a linguagem natural é o mais expressivo instrumento de comunicação. Em segundo lugar porque podemos pensar em ter um processo de geração de linguagem natural a partir do formalismo utilizado na construção da base de conhecimentos.

Um outro aspecto importante do emprego da LN é nos atos de fala. Aqui a ênfase é a estreita ligação entre discurso e ação. Vale aqui lembrar a analogia feita em [Laurel 86] a respeito da interação entre usuário e sistema. Existe um estado anterior à interação, existe uma verdadeira ação teatral na qual participam usuário e sistema que deve levar a uma catarse, e existe um estado final, após a experiência conjunta, o qual é o resultado de uma transformação obtida graças ao sucesso desta ação integrada.

Explorando ainda a LN como conjunto de atos de fala,

temos em [Winograd 86] a idéia de considerá-los estrutura básica para o funcionamento de uma organização.

Na próxima seção encerramos resumindo algumas idéias que nos parecem relevantes.

7 - Comentários Finais

Ao introduzir um sistema num ambiente de trabalho modifica-se o quadro geral de ação para a execução das tarefas [Brown 86]. Isto pode ser feito com o objetivo de melhorar a qualidade do produto, reduzir o seu custo e/ou melhorar a sua aceitação no mercado.

Neste contexto torna-se visível a importância fundamental da integração do sistema no ambiente de trabalho. Esta integração depende de um bom acoplamento entre usuário e sistema, onde a interface deve ficar, de preferência, imperceptível para o usuário. Como em geral são vários usuários diferentes usando o mesmo sistema, fica evidente que a interface deve ser sensível às características do usuário, adaptando o acesso às funções do sistema conforme se revele mais agradável a cada usuário.

Dada a dificuldade de representar no sistema todos os tipos possíveis de usuário, fica mais atraente o caminho de

explorar a capacidade mental do usuário como ser humano. A revelação do sistema ao usuário, principalmente através de explicações em linguagem natural, aparece como uma alternativa bastante interessante.

Na comunicação entre usuário e sistema a linguagem natural vai descrever o novo contexto de trabalho que nasce justamente da integração deste novo elemento: o sistema. Este novo mundo de referência apresenta novas alternativas para atingir os mesmos objetivos e até mostra novos objetivos que podem ser atingidos com o sistema. A idealização do sistema deve levar em conta todas estas mudanças que podem ocorrer em função de sua utilização [Brown 86].

Quanto mais claramente definida a estrutura do sistema, mais fácil de ser revelada ao usuário e mais adaptável à dinâmica do ambiente de trabalho. Em particular, na medida em que se consegue formalizar a estrutura do sistema, se torna possível também criar boas referências aos objetos do contexto do ambiente de trabalho, que é a área comum de referência no diálogo entre cada usuário e o sistema.

Entretanto, este acoplamento estrutural não prescinde de um modelo mental adequado que cada usuário deve construir e manter a respeito do sistema. Uma meta-linguagem explicativa, como a linguagem natural, pode ser considerada adequada para auxiliar ao usuário nesta construção mas depende de um ancoramento na experiência real de utilização do sistema.

Os exemplos de utilização do sistema constituem uma alternativa interessante para este ancoramento pretendido. Na apresentação dos exemplos valem as representações gráficas, as animações, os sinais luminosos e acústicos. Os exemplos, embora sejam um gancho no mundo real para os modelos abstratos construídos mentalmente, são apenas exemplos de comportamento e não revelam em profundidade as causas deste comportamento. Esta tarefa fica por conta das explicações em linguagem natural com apelos a analogias e/ou metáforas na medida em que pareçam adequadas.

Finalmente devemos observar que existe um alto grau de subjetividade tanto na idealização de sistemas como na definição das interfaces. Podemos dizer até que existe um conteúdo artístico neste trabalho de criação.

Referências Bibliográficas

Atkinson,M. et alli
"The Object-Oriented Database System Manifesto"
Rapport Technique Altair 1989

Booth, P.
"An Introduction to Human-Computer Interaction"
Hillsdale, Laurence Erlbaum Ass. 1990 2a. ed.

Brown,John Seely
"From Cognition to Social Ergonomics and Beyond"
in "User-Centered System Design"
eds. Norman,D.A. & Draper,S.W.
Hillsdale, Laurence Erlbaum Ass. 1986

Freundlich,Yehudah
"Knowledge Bases and Databases"
Computer nov. 1990

Laurel,B.K.
"Interface as Mimesis"
in "User-Centered System Design"
eds. Norman,D.A. & Draper,S.W.
Hillsdale, Laurence Erlbaum Ass. 1986

Lewis,C.;Norman,D.A.
"Designing for Error"
in "User-Centered System Design"
eds. Norman,D.A. & Draper,S.W.
Hillsdale, Laurence Erlbaum Ass. 1986

Maximiliano,Carlos
"Hermenêutica e Aplicação do Direito"
10a. ed. Forense 1988 (1a. ed. 1924)

Norman,D.A.
"Cognitive Engineering"
in "User-Centered System Design"
eds. Norman,D.A. & Draper,S.W.
Hillsdale, Laurence Erlbaum Ass. 1986

Parsaye,K.;Chignell,M.;Kosthatiam,S.;Wong,H.
"Intelligent Data Bases"
1988

Winograd,T. & Flores,F.
"Understanding Computers and Cognition:
A New Foundation for Design"
N.Y. Addison Wesley, 1986