Construção de Interfaces Homem-Computador: Uma Proposta Revisada de Disciplina de Graduação

Fábio Nogueira de Lucena
Hans K.E. Liesenberg

Relatório Técnico DCC–96-01

Janeiro de 1996

Construção de Interfaces Homem-Computador: Uma Proposta Revisada de Disciplina de Graduação*

Fábio Nogueira de Lucena Hans K.E. Liesenberg

Projeto Xchart†
DCC/IMECC/UNICAMP
Caixa Postal 6065
13081-970 Campinas/SP, Brazil
e-mail: xchart@dcc.unicamp.br
URL: http://www.dcc.unicamp.br/projects/Xchart

Sumário

A interface homem-computador é um dos componentes mais importantes de um sistema interativo. Grande parte das atuais propostas de grades curriculares de Cursos de Ciência da Computação não tratam adequadamente as várias questões pertinentes a este tópico. Este texto propõe a organização de uma disciplina de graduação desta natureza. A construção da interface de um sistema interativo é apresentada e exemplifica o tipo de trabalho prático que pode ser realizado pelos alunos.

1 Introdução

A interface homem-computador (interface, por simplicidade) é um elemento imprescindível para a aceitação de um sistema interativo pelo usuário. Estudos mostram que o usuário, muitas vezes, prefere um sistema com uma interface "agradável" àquele que oferece mais funcionalidade, desempenho e outros, mas com uma interface mais pobre. Após adquirido, um sistema interativo passa a ser utilizado freqüente ou esporadicamente. Em ambas as situações, a eficiência do usuário é influenciada pela interface. Os custos envolvidos são ainda mais significativos, se o uso for constante por vários usuários. A produção e impressão de manuais do usuário também contribuem com custos adicionais. Recursos funcionais, que jamais chegam a ser usufruídos devido a uma interface que os oculta ou inibe a sua utilização, significam trabalho sem proveito realizado por equipes de desenvolvimento. Há ainda os chamados sistemas críticos que, em alguns casos, são protagonistas de danos fatais

*Projeto financiado pela FAPESP, CAPES, CNPq/ProTem-CC/GEOTEC (Unicamp-UFG) e Microsoft. Este texto é uma revisão de trabalho apresentado no *Workshop de Educação em Informática*, Canela/RS, Julho/95. A revisão baseou-se nas experiências adquiridas com a realização da disciplina proposta no segundo semestre de 1995 no DCC/IMECC/UNICAMP.

†Projeto que se concentra no desenvolvimento (especificação e implementação) de interfaces homem-computador. Por curiosidade: a 22ª letra do alfabeto grego ($\mathbf{X}$ e χ) é identificada em Inglês pela palavra CHI, que coincide com o acrônimo de *Computer-Human Interface.*

atribuídos às suas interfaces mal projetadas. Dray [1] é uma elucidativa referência sobre a importância de interfaces.

Trabalhos nesta área visam melhorar a qualidade de interfaces produzidas através do emprego de métodos, técnicas, notações e ferramentas apropriados. O objetivo é diminuir os altos custos envolvidos na produção de boas interfaces e eliminar as dificuldades citadas acima.

Sociedades como a ACM e IEEE já recomendam a disciplina de interfaces nos cursos de graduação desde 1991. De fato, ela já vem sendo ministrada em várias instituições com seus próprios conteúdo [2, 7, 10]. A proposta aqui apresentada é uma simplificação das citadas. A profundidade e amplitude dos assuntos abordados são mais reduzidos, comparando com o tratamento dado pela propostas citadas. Esta redução em escopo se deve, em parte, à inexistência de uma infraestrutura didática adequada (principalmente *software*), e às dificuldades naturais decorrentes da implantação de uma disciplina relativamente recente. Não foi possível estabelecer comparações a nível nacional devido à carência de modelos semelhantes no país. Uma observação informal dos currículos de cursos de graduação publicados no Workshop em Educação em Informática, realizado em 1994, Caxambu/MG, revela a pouca atenção dada à área de interfaces.

Em conseqüência, este importante componente de um sistema interativo é tratado de forma inteiramente **ad hoc**, o que repercute negativamente na qualidade dos sistemas produzidos. A qualidade influi, entre outros, diretamente na satisfação do cliente, o que, por sua vez, se reflete nas vendas (impacto comercial).

Para muitos o emprego de ferramentas de última geração utilizadas em todo o mundo parece ser suficiente. Entretanto, boas ferramentas apenas facilitam o trabalho de bons profissionais. Elas não garantem necessariamente um bom produto, nem mesmo quando operadas por profissionais experientes. Além do mais, é preciso selecionar as ferramentas de uma forma criteriosa. Atualmente existem centenas delas, variando de dezenas até dezenas de milhares de dólares. Naturalmente possuem objetivos distintos, enfrentam de forma diferente os vários problemas de desenvolvimento e conduzem a produtos de qualidade e sofisticação variadas. O profissional pertinente deve estar habilitado a identificar a ferramenta certa não através de uma escolha superficial, mas pelo que ela fornece de forma subjacente.

Este trabalho apresenta uma proposta de disciplina voltada para o desenvolvimento de interfaces. Esse desenvolvimento é multidisciplinar e requer habilidades de várias áreas. A proposta enfatiza aspectos computacionais, sem descuidar do conhecimento aplicado e indispensável de outras áreas. Dessa forma, espera-se que a disciplina possa contribuir para uma *conscientização da realidade de interfaces (importância e problemas) e uma capacitação à tomada de decisões ao longo do processo de desenvolvimento de sistemas interativos (projeto e implementação)*. A proposta também é viável enquanto considera a configuração (hardware e software) típica ou comum dos atuais laboratórios de ensino de computação no país.

A disciplina, como aqui proposta, foi ministrada a partir do segundo semestre (1995) no DCC/UNICAMP, com o título `MC750 Construção de Interfaces Homem-Computador`. Apenas um requisito foi esperado dos alunos para se matricularem em `MC750: MC426 Engenharia de Software`, na qual o aluno adquire fluência em ciclos de vida, metodologias, notações e construção de protótipos. É interessante observar que esta última disciplina, em-

bora a "mais próxima" no âmbito de Ciências da Computação, não cobre apropriadamente o desenvolvimento de interfaces. Por exemplo, o tratamento dado a interfaces é superficial, quando existe, em livros de engenharia de software, como no texto, já clássico, de Pressman [8].

Houve uma procura inesperada pela disciplina por parte dos alunos — mais de 50 alunos inscreveram-se para cursá-la. Foram selecionados 23, que concluíram a disciplina.

ORGANIZAÇÃO DO TEXTO. A Seção 2 apresenta a disciplina proposta. As Seções 3,4,5 e 6 descrevem o consultor ortográfico e as várias etapas do seu desenvolvimento, que serviu de referência para a realização do trabalho prático pelos alunos.

2 Disciplina Proposta

A disciplina é descrita através de vários tópicos, a serem abordados nesta ordem, enumerados abaixo e comentados sucintamente. A proposta não é definitiva e alguns itens eventualmente serão refinados em função de experiências resultantes de futuras edições da disciplina.

Antes de descrever os tópicos e o conteúdo de cada um deles, convém discorrer sobre a bibliografia a ser utilizada pelos alunos. De acordo com a ênfase do curso, desenvolvimento de software de interfaces, a referência abaixo é recomendada:

> *Developing Software for the User Interface*
> Len Bass & Joëlle Coutaz
> Addison-Wesley, 1991

Ainda é recomendada a leitura da referência produzida pelos autores do presente trabalho, indicada abaixo, que fornece uma visão geral e atualizada da área de interfaces e dos seus processos de desenvolvimento específicos. Embora a ênfase dada também seja no desenvolvimento de software, esta referência é mais superficial, o que permite abordar mais assuntos e integrá-los em um todo coeso. Este é o propósito principal do texto:

> *Construção de Interfaces Homem-Computador*
> Fábio Nogueira de Lucena & Hans K.E. Liesenberg

Além dos textos acima também foi recomendada a leitura de:

> *Task-Centered User Interface Design — A Practical Introduction*
> Clayton Lewis & John Rieman (*shareware*), 1994
> FTP anônimo: `ftp.cs.colorado.edu`
> Diretório: `/pub/CS/distribs/clewis/HCI-Design-Book`

Estas referências ainda citam várias outras que podem ser empregadas por aqueles interessados em implantar um curso similar. Abaixo seguem os tópicos a serem abordados no curso:

1. Motivação. O impacto de uma interface boa/ruim e a importância da área devem ser bem esclarecidos aos alunos.

Também devem ser enfatizadas as conseqüências de um desenvolvimento inadequado de uma interface. Algumas perguntas deverão ter respostas bem definidas. Entre elas:

- Por que interfaces são importantes?
- Por que devemo-nos preocuparmo com interfaces?
- Quais os benefícios de uma boa interface?
- Quais os inconvenientes de uma interface mal projetada?
- Interfaces gráficas, *toolkits* e ferramentas modernas não é tudo que a equipe de programadores precisa?

2. Fundamentos básicos. Este tópico inclui a definição de vários conceitos: sistema interativo, interface homem-computador, diálogo, usabilidade, modelo mental/conceitual, níveis de conhecimento dos usuários finais, portabilidade no desenvolvimento de interfaces e consistência, para nomear alguns. Este tópico deverá fornecer ao aluno uma visão geral da área e suas subdivisões: projeto, implementação e avaliação.

Neste tópico devem ser esclarecidos conceitos e algumas dúvidas que em geral ocorrem. Por exemplo, a distinção entre projeto da interação e projeto do software da interação. O *projeto da interação* define a interface com a qual o usuário irá interagir e requer do projetista conhecimentos em fatores humanos e outras áreas. Esta tarefa, representada em uma notação apropriada, registra os aspectos de apresentação e comportamento de uma interface. Terminada esta atividade, é preciso confeccionar o software, que implementa o que foi definido. Um engenheiro de software é o especialista indicado para isto, desde que devidamente preparado para utilizar métodos e técnicas especificamente desenvolvidos para produzir componentes de software dessa particular categoria.

Neste tópico, uma visão horizontal e abrangente da área deverá ser fornecida, que deverá ser suficiente para o aluno se situar na vasta área de interfaces.

3. Problemas de desenvolvimento de interfaces. A cada dia é mais notória a importância de uma interface. Nem por isso as dificuldades de desenvolvimento deste componente são reduzidas. A maior dificuldade reside no projeto da interação. Ainda não há normas infalíveis para criar uma "boa" interface, o que torna a construção de protótipos um procedimento inevitável. Este procedimento, por outro lado, influencia o ciclo de vida do sistema e as abordagens a serem empregadas. "Boas intenções", "senso comum" e "decisões razoáveis" não são suficientes para produzir boas interfaces. A implementação é volumosa e complexa [6]. Myers [5] apresenta um excelente panorama dos desafios de desenvolvimento de interfaces.

Neste tópico serão apresentadas aos alunos as várias dificuldades enfrentadas para produzir uma "boa" interface. Nos tópicos precedentes foram apresentadas a motivação e uma visão geral da área. Neste ponto será identificado um dos procedimentos cuja execução requer atenção especial. O tópico corrente ressalta os desafios para produzir adequadamente uma interface. Nos tópicos seguintes deverá ser enfatizado o desenvolvimento do seu código.

4. Processo de desenvolvimento. O processo de desenvolvimento é um dos itens mais complexos e, em conseqüência, recebe a merecida atenção nesta disciplina. O modelo de ciclo de vida de desenvolvimento de sistemas denominado *Waterfall*, por exemplo, onde fases fluem seqüencialmente, uma após a outra, é impróprio para interfaces. Para garantir uma boa interface é necessário construir e avaliar protótipos. A palavra chave é iteração, conforme os modelos da Figura 1.

No ciclo da esquerda [7], modelos formais podem ser empregados para descrever o *projeto* e permitir uma avaliação sem a necessidade de gerar código. Por outro lado, outras notações específicas podem ser utilizadas por ferramentas para a produção automática de protótipos ou o produto final. Alternativamente, só a codificação manual será utilizada na construção de protótipos, o que inibe mudanças e experimentos com alternativas, devido às dificuldades de alterar código complexo e intrincado. Isto é ruim, pois mesmo o projetista experiente, munido de diretrizes e padrões precisa avaliar várias versões de um projeto. A *avaliação* permite identificar problemas na interface (desempenho do usuário e taxa de erros são alguns exemplos).

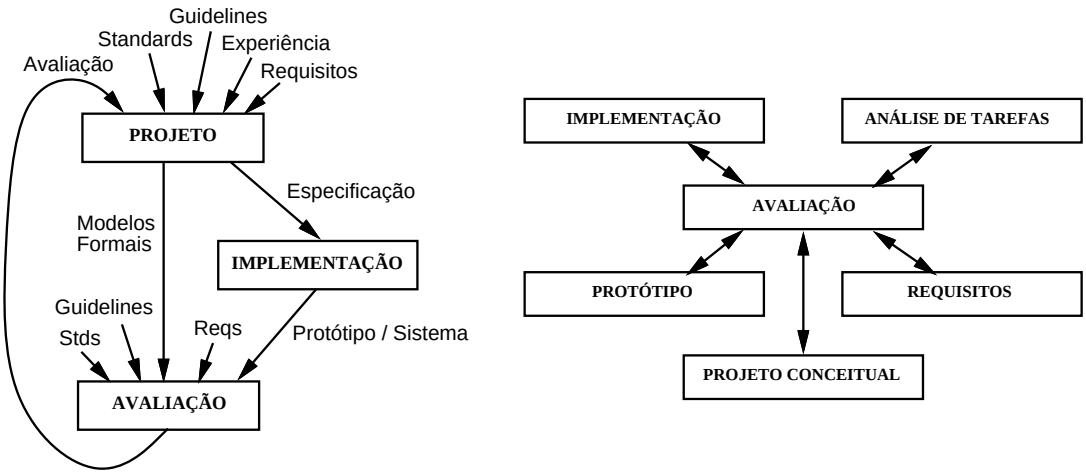

Figura 1: Ciclos de vida ([7] e [3])

O ciclo da direita [3] é fruto da observação de desenvolvedores na realização de suas atividades, onde atividades *bottom-up* (protótipos) são mescladas com outras *top-down* (análise de sistema). Neste modelo, o desenvolvimento da interface pode ser iniciado pela avaliação de um sistema já existente ou de um protótipo, por exemplo. Teoricamente qualquer atividade pode ser realizada após a avaliação de uma anterior.

O desenvolvimento de um sistema interativo não se restringe à sua interface, a funcionalidade do sistema também precisa ser desenvolvida. Os modelos vistos anteriormente são exclusivos para o desenvolvimento de interfaces. Felizmente, grande parte da bibliografia em engenharia de software está voltada para o componente funcional. A Figura 2 apresenta um diagrama de atividades necessárias e o fluxo de dados entre elas para o desenvolvimento

de todo um sistema interativo. Uma linha tracejada "separa" o desenvolvimento da interface (parte inferior) daquele da funcionalidade. Há atividades que devem ser realizadas em comum acordo entre os responsáveis pelo desenvolvimento dos dois componentes, desde a análise do sistema (onde requisitos são estabelecidos) até testes (onde erros funcionais podem ser identificados).

Neste tópico, estas e outras questões pertinentes ao processo de desenvolvimento de sistemas interativos (em especial, interfaces) devem ser abordados pelo professor. O objetivo é apresentar as propostas, mais comentadas na literatura, de ciclos de vida de sistemas interativos, além de identificar e definir suas várias etapas, produtos e técnicas utilizados em cada uma delas.

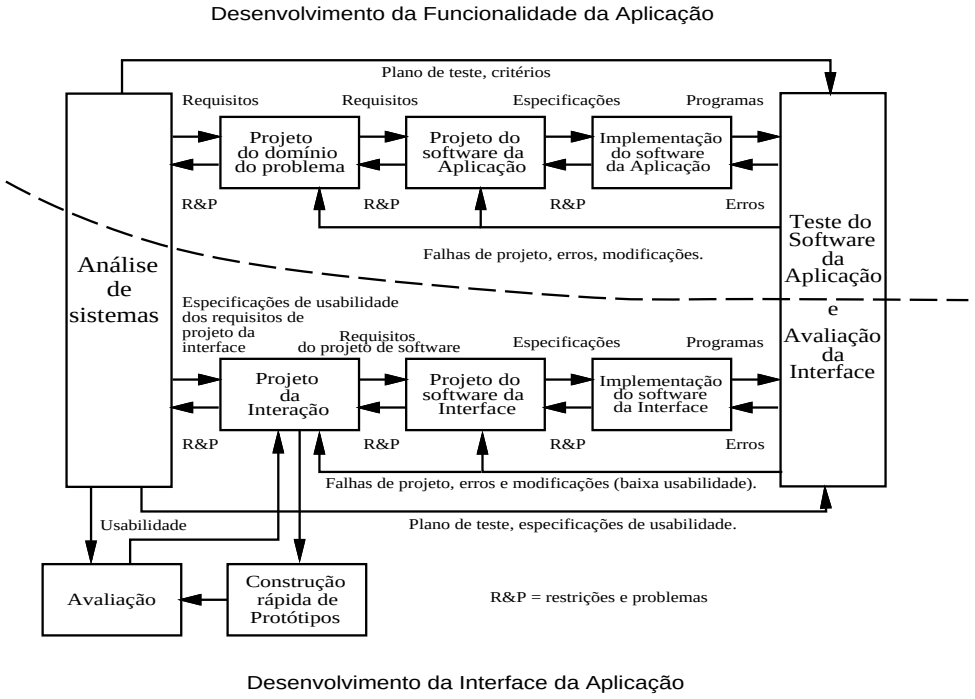

Figura 2: Modelo de desenvolvimento de sistemas interativos (adaptado de [3]).

5. Projeto. O projeto envolve o domínio de tipos de diálogo, dispositivos de entrada/saída, diretrizes, guias de estilo, padrões (como OpenLook, Motif), princípios e estilos de interação que orientam o projetista na tomada de decisões. O resultado deve ser registrado através de especificações formais, protótipos ou documentos informais, como esboços de telas.

Neste tópico será apresentado o material com o qual o projetista deve estar munido para realizar o projeto da interação de uma interface.

6. Implementação. Implementar adequadamente uma interface exige mais que simplesmente programar em MS-Windows ou Xlib. Conceitos como independência de diálogo, recursos disponíveis em *toolkits*, bibliotecas gráficas, sistemas de ajuda (*help*), modelos de estrutura de controle de diálogo, UIMS (*User Interface Management Systems*), sistemas

de janela e outros são necessários. Sem eles não há como tomar decisões seguras sobre alternativas de implementação.

Neste tópico serão apresentadas as abordagens atualmente empregadas para a organização e implementação do software de uma interface. Este tópico representa a ênfase principal de toda a disciplina.

7. Avaliação. A avaliação serve para orientar o ciclo iterativo de projeto. Sem a avaliação da interface, mesmo que empírica, conforme realizada no exemplo apresentado neste texto, não há como identificar problemas na interface em construção, não há como estabelecer quando parar este ciclo iterativo ou como saber que alterações estão conduzindo a um projeto melhor. A avaliação pode utilizar modelos formais e envolve uma coleta de dados seguida de sua análise.

Neste tópico os alunos receberão informações sobre algumas das técnicas mais comumente empregadas para avaliar o projeto da interação de interfaces.

Comentários sobre a Proposta

Cada um dos tópicos acima de 5 a 7 são suficientes para preencher todo um semestre (quatro horas semanais durante 15 semanas) de aulas. Optou-se por uma visão geral de cada um deles, conforme os conteúdos apresentados acima. Cada um deles aborda, contudo, uma particular "instância" que habilita o aluno a realizar suas próprias atividades de desenvolvimento e cria ao menos um "nível de consciência" dos problemas e soluções em cada um deles. Isto pode ser ampliado através de um trabalho prático, onde o aluno é estimulado a experimentar o conhecimento adquirido a partir de informações apresentadas em sala de aula. O exemplo discutido na seção seguinte mostra um possível resultado de um trabalho prático. O trabalho é possível de ser realizado por grupo de 2 ou 3 alunos, cuja proposta de desenvolvimento é realizada a medida que os fundamentos teóricos são fornecidos.

Algumas observações, fruto de experiência anterior com a disciplina, merecem ser expostas. Primeiro, o conceito de interface, a separação entre a interface e a aplicação, para citar alguns dos conceitos importantes, são difíceis de serem absorvidos pelos alunos. Existem dificuldades em observar um sistema interativo separado em interface e aplicação. Isto já era esperado pois, em geral, o desenvolvimento de um sistema interativo resulta em um código monolítico, conforme prática comum. Isto dificulta a compreensão de propostas para a organização estruturada do software de sistemas interativos. Para agravar ainda mais esta situação, as atuais propostas não estabelecem regras infalíveis. Muitas vezes são empregados meta-modelos para dar flexibilidade ao projetista na tomada de decisões para um sistema particular. Esta dificuldade pode ser amenizada através de exercícios nos quais o aluno é obrigado a estabelecer o que é de responsabilidade da interface e da aplicação.

Segundo, nem todos os alunos tinham fluência em orientação a objetos (OO) e C++, o que se tornou um empecilho para alguns, pois a grande maioria dos atuais *toolkits* disponíveis (comerciais ou de domínio público) para implementar a apresentação de interfaces faz uso de um modelo OO e de C++ para prover seus recursos. Esta falha deve ser sanada em disciplinas anteriores, pois é uma questão periférica para os propósitos da disciplina em questão.

Por último, no início do curso, os alunos foram estimulados a submeterem propostas

para o trabalho prático, que envolvia apenas o desenvolvimento da interface da proposta. Esta liberdade, contudo, não trouxe bons frutos. Muitas das interfaces desenvolvidas escondiam sistemas, onde a aplicação era o desafio. A interface era simples ou não contemplava exigências que obrigariam exercitar conceitos fornecidos em sala de aula. Como em uma disciplina de um semestre o tempo é um recurso escasso, não se pode esperar que os alunos façam uma escolha apropriada após o primeiro ou segundo mês de aula — eventualmente não haveria tempo suficiente para concluir o trabalho de propostas equivocadas. Para eliminar este problema, o professor pode apresentar um conjunto de propostas. Cada grupo seleciona aquele que o interessa. Ou talvez melhor ainda, uma única proposta deveria ser executada por todos os alunos, pois neste caso, discussões em sala de aula sobre algumas decisões específicas poderiam estimular o desenvolvimento e a competição salutar entre os grupos, já que todos têm um único sistema como referencial.

3 Um Sistema Interativo Ilustrativo: o Consultor Ortográfico

PROCESSO DE DESENVOLVIMENTO. Em vez de utilizar um sistema inexistente e desenvolvê-lo completamente, optou-se por um sistema já implementado e operacional. Para testar o processo, utilizou-se a interface do sistema existente que serviu como um protótipo inicial. Este protótipo foi avaliado dando origem a um segundo e assim por diante. Esta abordagem é particularmente interessante para a disciplina proposta, pois evita que os alunos utilizem uma quantidade de tempo considerável para o desenvolvimento da funcionalidade, que não é de interesse específico da disciplina. Vários protótipos foram sucessivamente propostos e avaliados. Aqui são apresentados apenas a avaliação do protótipo inicial, o protótipo resultante desta avaliação e o protótipo mais recente.

EXEMPLO. O consultor, em uso por mais de dois anos no DCC/IMECC/UNICAMP, tem por finalidade verificar a ortografia de palavras. Ele não realiza nenhuma espécie de análise sintática ou semântica. A entrada pode ser fornecida em um de quatro formatos possíveis. Uma palavra que não conste em um dicionário básico e em dicionários adicionais, eventualmente fornecidos pelo usuário, é tida como desconhecida. Para palavras desconhecidas o usuário pode requisitar a exibição de possíveis alternativas.

A tela inicial da versão original do consultor é mostrada na Figura 3. Nela observamos o botão de `Quit` no canto superior esquerdo. Alguns botões aparecem apenas sombreados (`Alternativas`, `Remova` e `Aprenda`). `Alternativas` fornece opções para uma palavra desconhecida. `Remova` apenas elimina palavras selecionadas de uma lista de palavras desconhecidas. `Aprenda` acrescenta palavras selecionadas ao dicionário previamente determinado. `Formato` e `Modo` fornecem opções para o formato dos dados de entrada e a forma com que será fornecida (de forma interativa ou via arquivo). No canto superior direito têm-se `Info` (informações gerais) e `Verifique` (inicia verificação).

4 Análise do Sistema

Este processo identifica o sistema e "o que" ele deverá fazer. Em outras palavras: (1) objetivo, (2) análise do usuário, (3) análise de tarefas e (4) análise funcional.

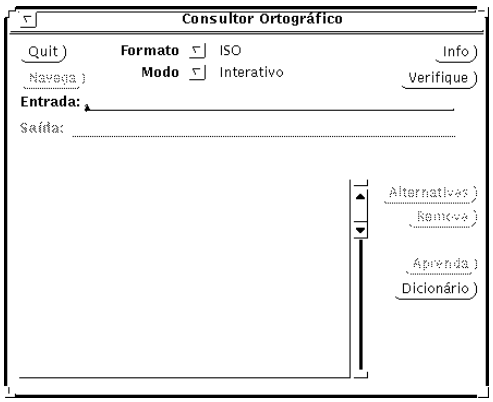

Figura 3: Tela inicial.

4.1 Objetivo: *verificação ortográfica*

4.2 Análise do usuário

- Características dos usuários (categorias):
  Especialistas em Ciência da Computação; universitários e funcionários públicos com, pelo menos, o segundo grau completo.

- Habilidades:
  Todos os usuários fazem uso de computador na realização de suas atividades diárias.

- Conclusões (requisitos a serem atendidos):

  - Simplicidade de uso.
  - Facilitar recordação do modo de funcionamento do sistema.
  - Estilos de interação não devem incluir uma linguagem de comandos.

4.3 Análise de Tarefas

1. Acrescentar dicionário local para ser consultado.
2. Remover dicionário local previamente selecionado para consulta.
3. Verificar arquivo: fornecê-lo e selecionar seu formato.
4. Selecionar formato de saída dos resultados.
5. Fornecer palavra para verificação.
6. Possibilitar o percurso de listas de palavras desconhecidas e sugeridas.

4.4 Análise Funcional

Nesta análise é suficiente observar que o consultor compreende um conjunto de ferramentas, *já implementadas*, que contribuem para a sua funcionalidade.

5 Projeto da Interação

O projeto envolveu vários ciclos de avaliação-projeto-protótipo, onde um protótipo gerado serve de entrada para novo ciclo. Nesses ciclos, atividades como a identificação de tarefas, análise do usuário e objetivos são paulatinamente refinadas. A avaliação foi feita com o auxílio de vários usuários representando as categorias identificadas. O projeto foi finalizado quando os usuários consideraram-se satisfeitos com o último protótipo, o que não significa necessariamente que seja o melhor! Tempo e recursos disponíveis limitam o número de vezes que o ciclo é repetido. Abaixo segue uma breve descrição de protótipos desenvolvidos com as respectivas avaliações que se basearam nas diretrizes contidas em [11] e em observações dos usuários. Limitamo-nos a apresentar o protótipo zero, o protótipo um e o protótipo N, que foi obtido após sucessivas avaliações dos precedentes. O protótipo zero corresponde à interface previamente existente do consultor (Figura 3) e não envolveu atividades de projeto.

5.1 Protótipo Zero

Interface do consultor brevemente comentada na Seção 3.

5.2 Avaliação do Protótipo Zero

A avaliação inicial revelou alterações necessárias agrupadas, em categorias, abaixo:

Consistência

1. É fácil constatar que o rótulo de vários botões, inclusive mensagens, estão escritas em Inglês ao contrário de outros que se encontram em Português. Os verbos (rótulos dos botões) alternam entre o modo imperativo na segunda pessoa do singular (`Navega`) e na terceira pessoa do singular (`Verifique`, `Remova` e `Aprenda`).
2. O botão `Alternativas` é executado apenas, se uma palavra estiver selecionada. O mesmo não ocorre com os botões (`Remova` e `Aprenda`).

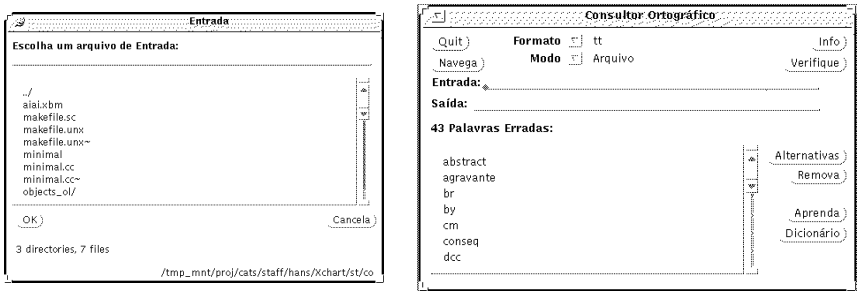

Figura 4: Outros instantes de interação.

Modelo Mental do Usuário.

Os botões `Remova` e `Dicionário` não são intuitivos. A maioria dos usuários consultados nunca tinha utilizado tais recursos e/ou não sabia de que se tratava.

Prevenção de Erros.

Na Figura 4 é possível observar uma lista de palavras desconhecidas, enquanto que o arquivo, cuja verificação deu origem a tais palavras, não é devidamente indicado ao usuário. Esta situação pode provocar consultas equivocadas, conforme apontado por alguns usuários.

Clareza

1. O rótulo `Info` não possui significado nítido, além de ser o único "abreviado."
2. Na Figura 4 (esquerda) não há uma definição visual clara, onde o usuário pode fornecer o nome do arquivo a ser considerado.
3. Na Figura 5 (esquerda) são fornecidas algumas informações sobre o sistema. O botão no canto superior direito conduz à janela apresentada no lado direito da figura. Este é um longo e escondido caminho para se ter acesso à facilidade de ajuda do sistema. Além disto, o texto de ajuda contém informações sobre a utilização de ferramentas responsáveis pela funcionalidade do consultor e não da interface gráfica.

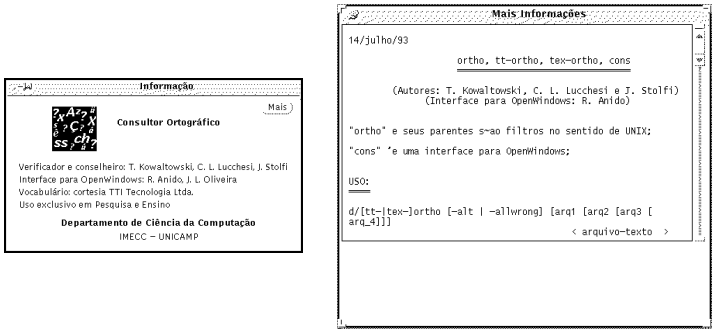

Figura 5: Instantes de interação.

Feedback

1. A Figura 3 mostra o sistema após ter sido selecionado um dicionário específico. Ou seja, não há *feedback* para o usuário após esta operação. Da mesma forma não se sabe qual ou quais são os dicionários que o consultor já está considerando. Embora funcionalmente implementado, a interface não permite utilizar este recurso! Além disso o botão `dicionário` seleciona um arquivo, onde toda palavra selecionada é armazenada, após ser pressionado o botão `Aprenda` – uma combinação complexa para a simplicidade da tarefa.
2. O conteúdo da janela de ajuda não é devidamente ajustado, para a surpresa do usuário, quando as dimensões da janela são alteradas. A Figura 5 (direita) exemplifica este fato.

5.3 Protótipo Um

A partir da avaliação feita do protótipo zero, vários esboços de tela, que tentavam eliminar as dificuldades identificadas, foram propostos e submetidos aos usuários. Tais esboços foram feitos em folhas de papel utilizando materiais comuns como régua e lápis. Cada consulta funcionou como uma "simulação" do sistema na qual o usuário interagia com a folha de papel, animada pelo projetista, à medida que o usuário descrevia suas "ações" para realizar determinada tarefa. Para cada tarefa era fornecido um cenário. Esta técnica tem suas vantagens e desvantagens [9]. O último esboço baseado nessa técnica é mostrado na Figura 6 (esquerda) e o protótipo operacional equivalente na Figura 6 (centro).

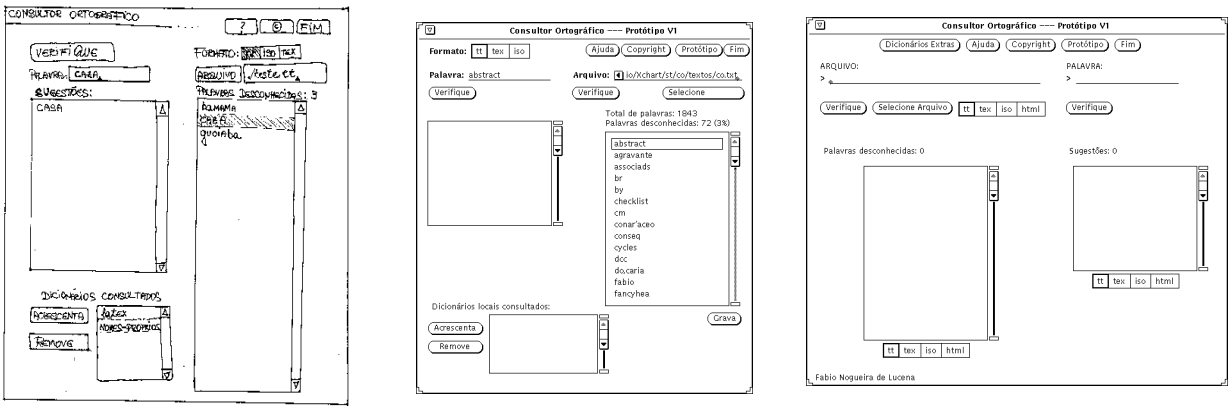

Figura 6: Protótipos *low-fi*, *hi-fi* e N.

5.4 Protótipo N

O protótipo N é apresentado na Figura 6 (direita).

6 Desenvolvimento do Software da Interface

Para a construção de protótipos operacionais e a implementação foi utilizado um *toolkit* de domínio público: *wxWindows* (FTP `skye.aiai.ed.ac.uk`, no diretório `pub/wxwin`). Um aluno, com conhecimentos básicos em C++ e orientação a objetos, pode, em pouco tempo, realizar pequenos projetos neste *toolkit*, para o qual existem implementações em *Ms-Windows* e *X-Windows*. Ele apresenta um bom compromisso entre a simplicidade conceitual das facilidades oferecidas e o poder de captura de características complexas de interfaces sofisticadas. Para facilitar o uso do *toolkit* foi elaborado um tutorial para iniciantes sobre o seu uso [4].

7 Comentários Finais

Em [2] é recomendada a disciplina de interfaces. Myers [5] mostra a relevância de interfaces no contexto de sistemas interativos e os desafios de projeto de interação e desenvolvimento do software de interfaces. Hix e Hartson [3] apresentam o processo de desenvolvimento de interfaces. O exemplo, descrito neste texto, exemplifica o emprego do processo nele descrito. Perlman [7] apresenta material pertinente à área e em [10] é apresentada a organização de cursos nesta área já ministrados em várias universidades de todo o mundo. Outras recomendações sobre o ensino do tema podem ser encontrados em *New Directions in HCI Education, Research and Practice* (`http://www.sei.cmu.edu/hci/directions/TitlePage.html`). Estes documentos são indispensáveis para docentes responsáveis pela implementação de disciplinas correlatas. Por último, este trabalho apresentou sucintamente várias questões pertinentes à área e uma proposta de disciplina, que esperamos ser lapidada paulatinamente à medida que novas experiências exigam melhorias.

Referências

[1] Susan Dray. The Importance of Designing Usable Systems. *Interactions*, 2(1):34–43, January 1995.

[2] ACM/IEEE-SC Joint Curriculum Task Force. Computing Curricula 1991, 1991. ISBN 0-8979-381-7.

[3] Deborah Hix and H. Rex Hartson. *Developing User Interfaces: Ensuring Usability Through Product & Process*. John Wiles & Sons, Inc., 1993. ISBN 0471-57813-4.

[4] Carlos Neves Júnior, Tallys H. Yunes, Fábio N. Lucena, and Hans K.E. Liesenberg. wxWindows: Uma Primeira Introdução. *Relatório Técnico DCC/IMECC/UNICAMP-95-19*, 1995.

[5] Brad A. Myers. Challenges of HCI Design and Implementation. *Interactions*, 1(1):73–83, 1994.

[6] Brad A. Myers and Mary Beth Rosson. Survey on User Interface Programming. In *CHI'92 Proceedings*, pages 195–202, Monterey, California, May 1992.

[7] Gary Perlman. User Interface Development. Technical Report SEI-CM-17-1.1, Software Engineering Institute, Carnegie Mellon University, 1989.

[8] Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. McGram-Hill, European edition, 1994.

[9] Marc Rettig. Prototyping for Tiny Fingers. *Communications of the ACM*, 37(4):21–27, April 1994.

[10] ACM SIGCHI. Curricula for Human-Computer Interaction, 1992. ISBN 0-89791-474-0.

[11] Sidney L. Smith and Jane N. Mosier. Guidelines for Designing User Interface Software. Technical Report ESD-TR-86-278, US Air Force, 1986. Anonymous ftp: `archive.cis.ohio-state.edu`, `pub/hci/Guidelines`.

Relatórios Técnicos – 1995

95-01 **Paradigmas de algoritmos na solução de problemas de busca multidimensional,** *Pedro J. de Rezende, Renato Fileto*

95-02 **Adaptive enumeration of implicit surfaces with affine arithmetic,** *Luiz Henrique de Figueiredo, Jorge Stolfi*

95-03 **W3 no Ensino de Graduação?,** *Hans Liesenberg*

95-04 **A greedy method for edge-colouring odd maximum degree doubly chordal graphs,** *Celina M. H. de Figueiredo, João Meidanis, Célia Picinin de Mello*

95-05 **Protocols for Maintaining Consistency of Replicated Data,** *Ricardo Anido, N. C. Mendonça*

95-06 **Guaranteeing Full Fault Coverage for UIO-Based Methods,** *Ricardo Anido and Ana Cavalli*

95-07 **Xchart-Based Complex Dialogue Development,** *Fábio Nogueira de Lucena, Hans K.E. Liesenberg*

95-08 **A Direct Manipulation User Interface for Querying Geographic Databases,** *Juliano Lopes de Oliveira, Claudia Bauzer Medeiros*

95-09 **Bases for the Matching Lattice of Matching Covered Graphs,** *Cláudio L. Lucchesi, Marcelo H. Carvalho*

95-10 **A Highly Reconfigurable Neighborhood Image Processor based on Functional Programming,** *Neucimar J. Leite, Marcelo A. de Barros*

95-11 **Processador de Vizinhança para Filtragem Morfológica,** *Ilka Marinho Barros, Roberto de Alencar Lotufo, Neucimar Jerônimo Leite*

95-12 **Modelos Computacionais para Processamento Digital de Imagens em Arquiteturas Paralelas,** *Neucimar Jerônimo Leite*

95-13 **Modelos de Computação Paralela e Projeto de Algoritmos,** *Ronaldo Parente de Menezes e João Carlos Setubal*

95-14 **Vertex Splitting and Tension-Free Layout,** *P. Eades, C. F. X. de Mendonça N.*

95-15 **NP-Hardness Results for Tension-Free Layout,** *C. F. X. de Mendonça N., P. Eades, C. L. Lucchesi, J. Meidanis*

95-16 **Agentes Replicantes e Algoritmos de Eco,** *Marcos J. C. Euzébio*

95-17 **Anais da II Oficina Nacional em Problemas Combinatórios: Teoria, Algoritmos e Aplicações,** *Editores: Marcus Vinicius S. Poggi de Aragão, Cid Carvalho de Souza*

95-18 **Asynchronous Teams: A Multi-Algorithm Approach for Solving Combinatorial Multiobjective Optimization Problems,** *Rosiane de Freitas Rodrigues, Pedro Sérgio de Souza*

95-19 **wxWindows: Uma Introdução,** *Carlos Neves Júnior, Tallys Hoover Yunes, Fábio Nogueira de Lucena, Hans Kurt E. Liesenberg*

95-20 **John von Neumann: Suas Contribuições à Computação,** *Tomasz Kowaltowski*

95-21 **A Linear Time Algorithm for Binary Phylogeny using PQ-Trees,** *J. Meidanis and E. G. Munuera*

95-22 **Text Structure Aiming at Machine Translation,** *Horacio Saggion and Ariadne Carvalho*

95-23 **Cálculo de la Estructura de un Texto en un Sistema de Procesamiento de Lenguaje Natural,** *Horacio Saggion and Ariadne Carvalho*

95-24 **ATIFS: Um Ambiente de Testes baseado em Inje,c ao de Falhas por Software,** *Eliane Martins*

95-25 **Multiware Plataform: Some Issues About the Middleware Layer,** *Edmundo Roberto Mauro Madeira*

95-26 **WorkFlow Systems: a few definitions and a few suggestions,** *Paulo Barthelmess and Jacques Wainer*

95-27 **Workflow Modeling,** *Paulo Barthelmess and Jacques Wainer*

Departamento de Ciência da Computação — IMECC
Caixa Postal 6065
Universidade Estadual de Campinas
13081-970 – Campinas – SP
BRASIL

`reltec@dcc.unicamp.br`