Avaliação Iterativa da Especificação de Interfaces com Ênfase na Navegação

Marckson Roberto Ferreira de Sousa

Tese de doutorado submetida à Coordenação dos Cursos de Pós-Graduação em Engenharia Elétrica da Universidade Federal da Paraíba em cumprimento às exigências para obtenção do grau de Doutor em Ciências no domínio da Engenharia Elétrica.

Área de Concentração
Processamento da Informação

Orientadora
Maria de Fátima Queiroz Vieira Turnell, PhD

Campina Grande, Paraíba
Dezembro de 1999

Avaliação Iterativa da Especificação de Interfaces com Ênfase na Navegação

Marckson Roberto Ferreira de Sousa

Tese de Doutorado

Orientadora

Maria de Fátima Queiroz Vieira Turnell

Campina Grande, Paraíba
Dezembro de 1999

S725s Sousa, Marckson Roberto Ferreira de
Avaliacao iterativa da especificacao de interfaces com enfase na navegacao / Marckson Roberto Ferreira de Sousa. - Campina Grande, 1999.
164 f. : il.

Tese (Doutorado em Engenharia Eletrica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Interfaces de Usuario (Sistema de Computador) 2. Avaliacao Iterativa da Especificacao de Interfaces 3. Avaliacao da Navegacao 4. Tese I. Turnell, Maria de Fatima Queiroz Vieira, Dra. II. Universidade Federal da Paraiba - Campina Grande (PB) III. Título

CDU 681.3(043)

AVALIAÇÃO ITERATIVA DA ESPECIFICAÇÃO DE INTERFACES COM ÊNFASE NA NAVEGAÇÃO

MARCKSON ROBERTO FERREIRA DE SOUSA

Tese Aprovada em 16.12.1999

Maria de Fátima Q. Vieira Turnell

MARIA DE FÁTIMA QUEIROZ VIEIRA TURNELL, Ph.D., UFPB
Orientadora

JOSÉ CARLOS MALDONADO, Dr., USP-São Carlos
Componente da Banca

Ana Carolina Salgado

ANA CAROLINA SALGADO, Dr., UFPE
Componente da Banca

ANGELO PERKUSICH, D.Sc., UFPB
Componente da Banca

BERNARDO LULA JÚNIOR, Dr., UFPB
Componente da Comissão

CAMPINA GRANDE - PB
Dezembro - 1999

Ainda que eu falasse as línguas dos homens e dos anjos, e não tivesse amor,
seria como o metal que soa ou como o sino que tine.

E ainda que tivesse o dom de profecia, e conhecesse todos os mistérios e toda a ciência,
e ainda que tivesse toda a fé, de maneira tal que transportasse os montes,
e não tivesse amor, nada seria.

(Coríntios 13: 1-2)

Resumo

Este trabalho apresenta um método e suas ferramentas de suporte utilizadas na especificação do componente interface com o usuário de sistemas interativos. O método se fundamenta na construção de modelos e associa procedimentos de avaliação a cada uma de suas fases. O propósito do método é assegurar a qualidade da interface no uso da interface de um protótipo derivado a partir de sua aplicação ao processo de concepção.

Com o objetivo de apoiar o método, foram desenvolvidas duas ferramentas. Uma ferramenta para coleta de dados durante uma avaliação empírica com o protótipo e uma ferramenta para suporte a análise das características estruturais da interface. Estas ferramentas se mostraram úteis no processo de localização de potenciais problemas de usabilidade e na investigação de soluções para estes problemas.

O método adota o formalismo MAD [SG 89] na análise e descrição de tarefas e apresenta um procedimento para conversão de uma descrição de tarefa em MAD para uma representação de modelo da interação. O método também apresenta um modelo de navegação no formalismo de redes de Petri construído com o propósito de analisar as características de navegação da interface com o usuário. Este documento ilustra a aplicação do método através de um estudo de caso - concepção da interface com o usuário de um Navegador (*Browser)* na Internet.

Abstract

This thesis presents a method and its supporting tools to be used in the specification phase of the user interface component of interactive systems. This method is based upon model building (formal and informal) and associates an evaluation procedure to each of its steps. The purpose of this method is to ensure quality in use for a user interface prototype derived from its application.

To support the method, two tools were developed. A data-logging tool to be used with the prototype during empirical evaluations, and another tool to support the structural analysis of the prototype features. Both tools have proven to help locate potential usability problems and represent a valuable help in the search for solutions for these problems.

The method adopts the formalism MAD [SG 89] for task analysis and description, and presents an algorithm for converting from the MAD task description into an Interaction Model Representation. The method presents a navigation model built in Petri Net to analyse user interface navigation characteristics. This model is built and analysed with the help of an existing tool – Design CPN [MSC 93]. This document illustrates the method's application by means of a case study – the conception of the user interface component for an Internet browser.

Sumário

Índice de Tabelas

Índice de Figuras

1 INTRODUÇÃO

O estudo da interação usuário-computador ainda é um campo novo, voltado para a pesquisa sobre a melhor forma como esta interação pode ocorrer. Este campo de estudo inclui tanto hardware quanto software, enfoca tanto o usuário quanto o sistema, e se fundamenta no estudo dos fatores humanos e da ergonomia da interação.

O processo de desenvolvimento deve ser guiado pela confiança no funcionamento (correção) que é importante para o software em geral, mas é particularmente importante para o software da interface, uma vez que é um aspecto diretamente percebido pelo usuário. Se a interface com o usuário apresenta problemas, o software poderá ser percebido como incorreto, independentemente da correção de sua funcionalidade [MC 99].

Os esforços iniciais de avaliação de sistemas interativos tinham em comum a ênfase no desempenho do computador. O registro da avaliação de interfaces, na década de 60, demonstra uma limitação de escopo, sendo esta voltada para verificar o sucesso ou insucesso na realização de uma tarefa específica, sem no entanto se aprofundar em aspectos relativos à qualidade da interação. Ao longo dos anos 70 e início dos anos 80, a avaliação ainda não consistia um foco de atenção dos projetistas, os quais se empenhavam em projetos mais elaborados, deixando a questão da avaliação para uma fase posterior.

Com a crescente complexidade dos métodos de interação, o foco da avaliação passou a enfatizar a relação usuário-computador e o desempenho do usuário. Na avaliação da relação usuário-computador, o objetivo maior tem sido relacionar as capacidades e limitações do usuário aos requisitos que estas capacidades e limitações impõem sobre a tecnologia dos sistemas interativos.

Um reflexo desta mudança de foco pode ser observado na evolução de padrões internacionais de qualidade, tais como os padrões da ISO (*International Standards Organization*[1]). Estes padrões evoluíram de conceitos de qualidade orientados ao produto, para conceitos de qualidade voltados para o atendimento às necessidades do usuário. Exemplos da primeira categoria são: a norma ISO 9000, que define qualidade como conformação com requisitos e, a norma ISO 8402 que define qualidade como a presença de aspectos especificados. Exemplos da segunda categoria são: a norma ISO 14598-1, que define qualidade como o atendimento às necessidades do usuário, a norma ISO 9241-11, que define métricas para avaliação de qualidade do ponto de vista de uso do produto e a norma ISO 9126 , que inclui usabilidade[2] entre outros atributos de qualidade de software.

A avaliação de qualidade é dependente da percepção do usuário e dos propósitos para os quais o produto foi desenvolvido. O propósito de um produto é apoiar os usuários na busca de um objetivo particular [Bev 97]. A norma ISO 14598-1 define 'qualidade externa' como a extensão na qual um produto satisfaz às necessidades declaradas e implícitas, quando utilizado sob condições específicas. Esta definição move o foco da qualidade do produto isoladamente, para a satisfação das necessidades dos usuários.

A norma ISO 14598-1, define qualidade no uso como a eficácia, eficiência e satisfação com a qual usuários específicos podem atingir objetivos específicos em ambientes específicos. A eficácia corresponde à precisão e completitude do produto. A eficiência diz respeito ao uso do tempo e de recursos disponíveis. É importante observar que a satisfação do usuário independe dos atributos específicos do produto.

Do ponto de vista de usabilidade, observou-se que a especificação de requisitos de qualidade em termos de desempenho é mais eficaz do que a especificação de atributos técnicos do produto necessários para atingir o desempenho desejado. Baseado neste princípio, a norma ISO 9241-3, para telas de vídeo, estabelece os atributos técnicos necessários para atingir a qualidade no uso.

---

[1] As normas ISO estão disponíveis para comercialização no 'ISO Catalogue' (http://www.iso.ch)

[2] Usabilidade pode ser traduzida na capacidade de aprendizado, compreensão e operação que o software oferece ao usuário.

Requisitos de qualidade devem ser expressos em termos de métricas que possam ser obtidas quando o sistema é utilizado, tais como a eficácia, eficiência e satisfação. Os valores pretendidos para as métricas levam à definição dos objetivos de projeto, os quais podem por sua vez levar à métricas para os atributos internos do software (por exemplo modularidade do software). Se as qualidades externas não são atingidas, os resultados da avaliação devem ser realimentados no processo de desenvolvimento do produto, modificando os atributos internos para que isto aconteça, levando assim a um processo contínuo de melhoras.

A norma ISO 13407 apresenta um arcabouço para alcançar a qualidade no uso por meio da adoção de atividades de projeto centrado no usuário, ao longo do ciclo de vida de sistemas interativos. No projeto centrado no usuário, a concepção de soluções passa pela utilização de simulações, modelos e protótipos, com a apresentação destas soluções para o usuário, de modo que possa realizar tarefas (ou simulá-las). Em seguida utiliza-se a realimentação do usuário para melhorar o projeto. Este ciclo é repetido até que os objetivos sejam alcançados.

1.1 MOTIVAÇÃO

Para que a qualidade no uso seja atingida, um passo importante é a avaliação da usabilidade. Os resultados da avaliação devem gerar informações que permitam a identificação e localização de problemas e sirvam de referência no desenvolvimento de soluções para estes problemas. A atividade de avaliação é essencial no projeto de sistemas interativos e deve acontecer ao longo de todo o ciclo de vida. As avaliações realizadas nas fases iniciais levam ao refinamento da atividade de projeto. Quanto mais cedo forem iniciadas, menos detalhadas são as especificações do produto e consequentemente mais barata e fácil a introdução de mudanças. Nos estágios iniciais do ciclo de projeto de software interativo, a construção e verificação de modelos tem se mostrado uma atividade valiosa.

1.2 OBJETIVOS DO TRABALHO

Indicadores de qualidade de uso podem ser obtidos a partir da análise da especificação da interface de um produto. Um dos indicadores da eficiência de realização de uma tarefa[3] (por exemplo o tempo de realização da tarefa), seria a extensão e complexidade do caminho projetado para o usuário realizá-la. A análise da especificação se fundamenta no uso de modelos.

No entanto, não se deve esquecer a importância da validação por um usuário. Um protótipo consiste em uma especificação mais facilmente validada por um usuário do que seria a abstração de um modelo. Assim este trabalho propõe como objetivo principal a investigação de um método que associe a avaliação baseada na verificação de modelos com a avaliação baseada na validação de protótipos, com o propósito de refinar o procedimento de análise da especificação da interface de produtos. O refinamento pretendido objetiva melhorar a capacidade de avaliar os mecanismos de interação projetados e aspectos do projeto visual, aos quais estão associados um número considerável de problemas de usabilidade.

O método deverá ser apoiado por um conjunto de ferramentas que facilitem sua aplicação, reduza os custos da avaliação e aumente a confiabilidade das informações coletadas, motivando a sua utilização.

1.3 ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado em oito capítulos e um apêndice, sendo este o primeiro capítulo.

O Capítulo 2 introduz os conceitos básicos necessários à compreensão do trabalho, e uma breve revisão dos métodos e ferramentas disponíveis para avaliação de interfaces.

O Capítulo 3 apresenta o método de concepção proposto neste trabalho.

[3] Neste trabalho, uma tarefa realizada por um usuário consiste de uma seqüência de ações realizadas para atingir um objetivo no seu contexto de trabalho.

O Capítulo 4 apresenta a etapa de análise da tarefa com a construção de uma representação em MAD (Método Analítico de Descrição). Para ilustrar esta etapa o Capítulo apresenta a descrição da tarefa do *Browser-B*, estudo de caso tratado ao longo deste trabalho. Ainda no Capítulo 4 é apresentado um esquema para mapeamento do modelo da tarefa para o modelo da interação que leva também em consideração o perfil do usuário. O capítulo encerra com a análise destes modelos para o estudo de caso.

No Capítulo 5 é apresentado um modelo de navegação construído em CPN (*Coloured Petri Nets*), com o propósito de avaliar os mecanismos de navegação tipicamente encontrados em interfaces cuja interação se baseia em janelas, menus e botões. Neste capítulo também é apresentado um mapeamento do modelo da interação para a representação CPN do modelo da navegação. Finalmente, é apresentada a construção do modelo do estudo de caso e sua análise a partir de um conjunto de propriedades de redes de Petri, as quais objetivam assegurar aspectos ergonômicos relativos a navegação.

O Capítulo 6 descreve a construção de um protótipo do *Browser-B*, para o ambiente *OpenLook*, com base no modelo da interação. Em seguida o protótipo é avaliado com base na coleta de dados da interação, apoiada pela ferramenta FAIUnix, desenvolvida no contexto deste trabalho. O capítulo conclui com a apresentação e análise dos dados coletados.

O Capítulo 7, ilustra o caráter iterativo do método, destacando a realimentação de dados entre as diferentes fases do método.

Finalmente, no Capítulo 8, são discutidos os resultados do trabalho, tecidas considerações sobre o alcance destes resultados e apresentadas propostas de continuidade.

No Apêndice A, é apresentada a árvore hierárquica da tarefa do *Browser-B*, de acordo com o método de descrição MAD.

2 DESENVOLVIMENTO DE SOFTWARE INTERATIVO

Diante do grande número de alternativas para concepção de interfaces é necessário averiguar hipóteses de especificação antes de iniciar a fase de implementação, até mesmo de protótipos [Tre 94a]. Este capítulo introduz os conceitos básicos envolvidos na concepção e avaliação de software interativo.

2.1 O CICLO DE VIDA DE SOFTWARE

De acordo com o paradigma da Engenharia de Software para desenvolvimento de software, o "ciclo de vida clássico", também conhecido por "modelo cascata", se inicia no nível de análise de requisitos, seguido do projeto, codificação, teste e manutenção.

Na fase de análise de requisitos, as necessidades devem ser documentadas e revistas com o cliente. Uma vez gerado o código, iniciam-se os testes do programa. Estes testes concentram-se nos aspectos lógicos, funcionais e de implementação, de modo a assegurar que as entradas definidas produzam resultados de acordo com as exigências.

Os métodos de teste devem ser econômica e efetivamente aplicáveis tanto a sistemas de grande porte, quanto a sistemas de pequeno porte. A fase de testes culmina com testes de validação. A validação é bem sucedida quando o software funciona da maneira esperada pelo cliente. Por sua vez, as expectativas do cliente são definidas na fase de especificação de requisitos, com base em uma documentação que descreve todos os atributos do software visíveis ao usuário em uma sessão denominada Critérios de Validação.

A atividade de testes é elemento de um tema mais amplo denominado Verificação e Validação, que abrange muitas das atividades referentes à garantia da qualidade de software. A verificação refere-se ao conjunto de atividades que garante que o software implementa corretamente uma função especifica. Segundo [Boe 88], a verificação assegura que o produto foi construído da forma correta. A validação, por sua vez, diz respeito às atividades que garantem que o software construído é o software especificado pelo cliente.

Verificação de Modelos

A verificação de modelos é um método que busca responder à pergunta "Qual a segurança no funcionamento de um sistema?". Esta tarefa é suscetível a erros, face ao grande número de transições entre os estados de uma interface, o que torna complexo o processo de análise [PBS 95]. O uso de ferramentas computacionais possibilita a verificação automática de modelos, viabilizando a análise a diferentes cenários da interação em um tempo mínimo, reduzindo a incidência de erros e promovendo reduções significativas na complexidade do procedimento de avaliação.

Validação de Protótipos

O processo de desenvolvimento iterativo é baseado na avaliação do projeto pelos usuários. No início do ciclo de vida esta avaliação pode se fundamentar na construção de protótipos-rápidos. A prototipação pode ser usada para validar e verificar se a interface atende aos requisitos de usabilidade.

2.1.1 Prototipação

A prototipação, é um processo baseado na criação de um modelo do software que será implementado. O modelo, pode assumir a forma de um protótipo em papel, um protótipo de trabalho, ou ainda um programa que executa parte da funcionalidade desejada. Os desvios ou erros descobertos na fase de validação de protótipos evitam atrasos no projeto. Dada a dificuldade de correção, algumas das deficiências encontradas, para serem superadas devem ser negociadas com o cliente. Assim, nas situações que envolvam incerteza, uma abordagem de prototipação pode ser mais adequada do que a abordagem clássica do ciclo de vida [Pre 95].

Embora o modelo clássico do ciclo de vida tenha sido amplamente utilizado, vem sendo criticado porque não acomoda a incerteza natural que envolve a fase de análise de requisitos. Muitas vezes é difícil para o cliente declarar explicitamente todas as exigências do projeto e deverá aguardar até que a versão final lhe seja entregue. Este é um modelo linear que não prevê a realimentação entre etapas. Um erro que só poderá ser detectado e revisto na fase de testes poderá ser desastroso. A necessidade de extensões na fase final do projeto pode resultar na necessidade de reprojeto.

Um modelo fundamentado na prototipação é apresentado em [Pre 95]. Este paradigma se baseia em um processo de iteração no qual protótipos são refinados à medida que o desenvolvedor compreende melhor as necessidades do cliente. Nesta abordagem o protótipo serve como mecanismo para identificar os requisitos do software. Quando são apoiados por ferramentas de desenvolvimento de interfaces os protótipos podem vir a se transformar no produto final, são os protótipos evolutivos.

2.1.2 Modelo Espiral

O modelo espiral do ciclo de vida incorpora o modelo clássico do ciclo de vida, e o conceito de prototipação, no qual o protótipo desenvolvido é usado como um núcleo do produto que vai sendo refinado a partir de iterações do ciclo de vida resultando em um protótipo evolutivo.

O Modelo Espiral para a Engenharia de Software envolve um tipo de ciclo que se repete várias vezes na seqüência do modelo do ciclo de vida clássico, a cada vez incluindo mais detalhes do sistema [Boe 88]. Após cada iteração seqüencial linear, um produto é entregue ao cliente para avaliação e realimentação. Devido ao grande número de vezes em que se retorna à fase de requisitos, o produto final é mais robusto do que aquele que deriva da aplicação do modelo clássico.

A iteração[4] é apropriada ao desenvolvimento de software e em particular no desenvolvimento da interface. No caso do projeto da interface, este método é particularmente interessante dada a imprevisibilidade resultante, principalmente da falta de compreensão, por parte do projetista, do contexto de uso do produto e das expectativas do cliente. Portanto, a atividade de desenvolvimento da interface, deve ser inerentemente iterativa.

2.2 O Ciclo de Vida do Software da Interface

No campo do desenvolvimento e avaliação de interfaces com o usuário, muitos têm sido os paradigmas propostos, a exemplo do modelo estrela descrito em [HH 93]. Embora as fases e passos possam variar entre eles, a essência do ciclo de vida clássico é preservada.

A Figura 2-1, ilustra a conexão entre as várias atividades do desenvolvimento da interface, segundo Hix e Hartson [HH 93]. Nesta figura estão representadas as atividades: projeto da interação e projeto do software.

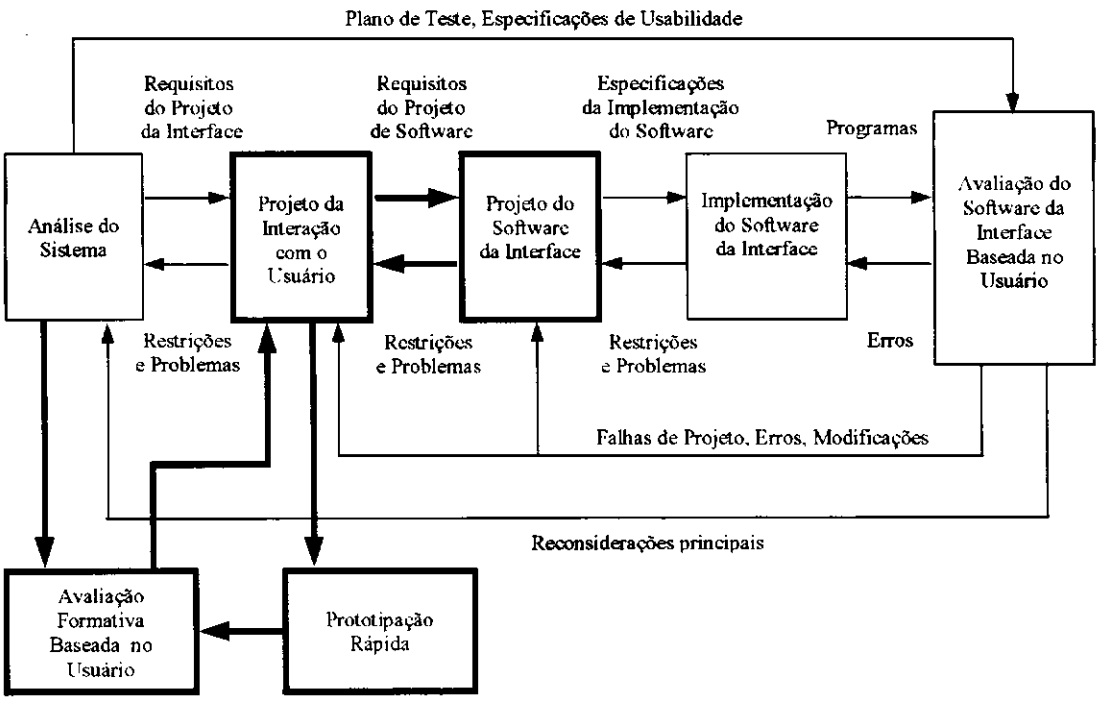

Figura 2-1: Prototipação rápida e Avaliação Formativa [HH 93]

[4] Neste contexto, iteratividade consiste em repetir as atividades precedentes até que especificações de usabilidade sejam atendidas.

O projeto do software da interface envolve os aspectos de projeto de software tratados na Engenharia de Software (algoritmos, estruturas de dados, bibliotecas, etc.). Na Figura 2-1, as conexões entre as várias atividades no processo de desenvolvimento não são temporais, isto é, não representam uma seqüência no tempo, são apenas canais de comunicação.

O projeto da interação envolve o projeto das ações do usuário, tarefas, realimentação do sistema, funcionalidade, sequenciamento de tarefas e ações, conteúdo, acesso à informação, objetos da interface, *layout* da tela e estilos de interação.

A análise de necessidades define para o sistema: os objetivos básicos, os propósitos e as características desejadas. O resultado é uma descrição do que o usuário será capaz de fazer com o sistema.

A análise do usuário define a classe de usuários em termos das tarefas que serão realizadas e o conhecimento necessário para realizá-las. O resultado é um conjunto de definições denominado "perfil do usuário".

Na Figura 2-1, a atividade de desenvolvimento foi estendida para representar a prototipação rápida e avaliação formativa[5] da usabilidade, gerando o ciclo: projeto – avaliação – reprojeto, ciclo este que será adotado neste trabalho.

2.2.1 Modelo Estrela

Para apoiar a avaliação contínua e a iteração durante o desenvolvimento de sistemas interativos, Hix e Hartson [HH 93] propuseram um modelo de ciclo de vida, denominado Modelo Estrela. Neste modelo, como é mostrado na Figura 2-2, os laços de iteração são menores do que aqueles resultantes da aplicação do Modelo Espiral, uma vez que este modelo, se caracteriza por um desenvolvimento que pode iniciar em qualquer ponto da estrela.

[5] A avaliação Formativa é realizada várias vezes ao longo do processo, seguida do reprojeto para cada versão significativa do projeto. A avaliação Somativa é realizada apenas uma vez, ao final do processo de desenvolvimento da interface.

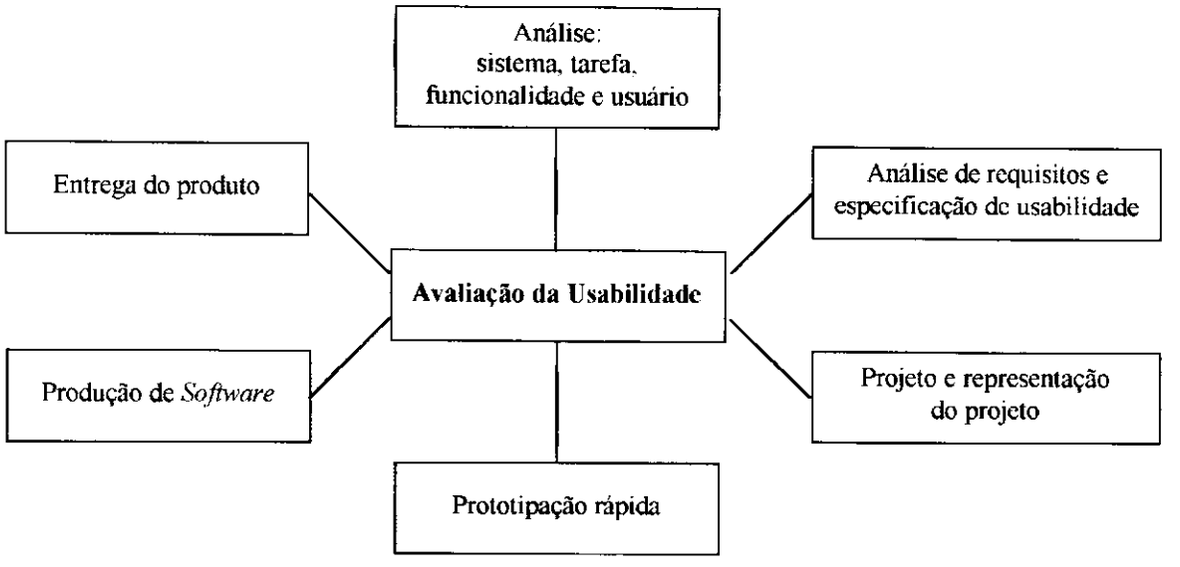

Figura 2-2: Modelo Estrela [HH 93]

Nele as atividades, não estão ordenadas ou conectadas em uma seqüência, significando que o desenvolvimento pode iniciar em qualquer das atividades e mover-se para qualquer outra delas. As atividades são fortemente interconectadas através do processo de avaliação da usabilidade (representado pelo centro da estrela). A usabilidade está relacionada à eficácia e eficiência da interface com o usuário e à reação do usuário face a interface [HH 93].

O processo iterativo termina quando objetivos de usabilidade são alcançados no projeto. Estes objetivos são expressos em termos quantitativos por ocasião da especificação e são testados ao longo do desenvolvimento. Os objetivos de usabilidade podem ser expressos em termos de atributos de usabilidade os quais podem ser medidos, conforme Tabela 2-1.

Tabela 2-1: Objetivos e Atributos da Usabilidade

Objetivos de Usabilidade	Atributos de usabilidade
Rapidez na realização de tarefas	Tempo de execução de tarefas: desempenho inicial, desempenho após o uso prolongado
Facilidade de Aprendizado	Quão rápido e fácil o usuário aprende a usar o sistema. Uso de recursos avançados
Elevada retenção de informações	Por quanto tempo o aprendizado é mantido
Satisfação subjetiva do usuário	Primeira impressão Opinião após o uso prolongado
Número reduzido de erros	Taxa de erros

Métricas

O tempo para realização de uma tarefa e o número de erros que um usuário comete ao realizá-la são exemplos de métricas utilizadas na avaliação da usabilidade [Nie 96]. Durante a avaliação, os valores medidos podem ser comparados a valores planejados, valores minimamente aceitáveis e valores desejáveis.

2.3 Modelando a Interface

Na área de projetos de interface a relação entre os usuários e a interface pode ser descrita e analisada através de modelos. A adoção desta abordagem, no entanto, se defronta com a dificuldade na escolha de um método adequado para modelar a funcionalidade das interfaces.

Embora muitas técnicas de modelagem existam, isoladamente elas não atendem completamente aos objetivos de um modelo da interação: analisar o conhecimento do usuário, predizer o desempenho do usuário e modelar a interface do usuário.

Segundo Schlungbaum [SE 96] os métodos para especificação formal podem ser utilizados pelos projetistas de interface para descrever o comportamento de um sistema sem que seja necessário realizar a sua implementação. A especificação possibilita a descrição formal dos requisitos e idéias de projeto em diferentes níveis de abstração.

2.4 Modelo do Usuário

As técnicas de representação de interfaces descrevem os objetos da interface e como eles funcionam, mas não descrevem adequadamente o que o usuário faz. Para tanto é necessária uma abordagem orientada-a-tarefa, uma vez que é através de tarefas que o usuário vê a interface. Nesta categoria estão as técnicas comportamentais, tais como GOMS (*Goals, Operators, Methods and Selection rules*) [CSM 83], TAG (*Task Action Grammars*) [Rei 81], e PUM (*Programable User Model*) [YGS 89].

O modelo GOMS (*Goals, Operators, Methods and Selection rules*)[CSM 83] caracteriza a interação do usuário como uma atividade orientada por objetivos, na qual o menor caminho para atingir um objetivo é considerado o melhor. Este modelo permite a predição de tempos de desempenho e taxas de erro a partir da análise da complexidade da hierarquia de objetivos do usuário. A idéia central deste modelo é que os estados correspondentes aos objetivos do usuário devem ser facilmente atingidos para que o sistema possa ser considerado um sistema utilizável. No entanto, este modelo não considera o planejamento e avaliação de estratégias alternativas de diferentes usuários.

Enquanto o desenvolvimento de GOMS foi motivado pelo interesse em métricas de desempenho, o desenvolvimento de TAG (*Task Action Grammar*) [Rei 81], objetivou utilizar uma gramática para reduzir a complexidade da semântica da execução de uma tarefa. Uma descrição em TAG permite avaliar o grau de especificidade do conhecimento que um usuário tem acerca da tarefa a ser realizada. No entanto, nem GOMS nem TAG representam o componente do sistema com o qual o usuário interage.

Contrastando com estes modelos, PUM (*Programable User Model*) [YGS 89] é um modelo executável que simula o comportamento humano durante a interação com uma máquina. Neste modelo é representado o efeito que as respostas do sistema provoca sobre o usuário e suas ações subseqüentes. Portanto, GOMS e TAG pertencem à categoria de modelos que possibilitam a predição do comportamento do usuário, enquanto que PUM pertence à categoria de modelos que permitem a simulação da interação.

Para os propósitos deste trabalho, nos interessa modelar apenas o conhecimento que o usuário deve ter acerca da tarefa que irá realizar com o apoio do sistema. Assim foi adotado o modelo Sintático-Semântico do conhecimento do usuário proposto por Shneiderman [Shn 98].

Segundo este modelo o conhecimento do usuário pode ser classificado em:

- Conhecimento sintático – variado, dependente de dispositivo, adquirido por memorização e facilmente esquecido. Por exemplo, a ação de apagar um caracter que pode ser realizada através da tecla *delete*, *backspace*, *CTRL-H*, *escape*, etc.
- Conhecimento semântico (tarefa e ações) – estruturado, independente de dispositivo, adquirido por aprendizado e estável na memória. Por exemplo, o usuário pode entender o conceito de salvar um arquivo como o armazenamento da informação, independente da

existência de uma entrada no diretório, que consiste em um nome, tamanho, data de criação, proprietário, controle de acesso, etc.

Deste modelo foram extraídos os aspectos mais relevantes para uma aplicação e compilados na forma de uma descrição denominada Perfil do Usuário.

2.5 MODELO DE TAREFAS

Do ponto de vista da tarefa, a modelagem de interfaces impõe a definição lógica da tarefa a ser executada, enfatizando-se entre outros aspectos: seqüências de execução, seqüências alternativas, bem como o tempo gasto pelo usuário para desempenhar cada tarefa.

O conceito de tarefa é usado para modelar uma atividade do usuário em um domínio específico de aplicação. Modelos de tarefa têm sido usados no desenvolvimento e projeto de sistemas interativos [SG 89]. A idéia é criar uma interface a partir de uma especificação declarativa da tarefa.

O modelo da tarefa é representado por uma hierarquia de tarefas, onde os níveis mais altos são mais abstratos e lógicos, enquanto que os níveis mais baixos são mais concretos e orientados para descrever as ações físicas necessárias para interagir com a interface. As tarefas neste nível são tarefas elementares, que não podem mais ser decompostas. O modelo é útil para entender as intenções do usuário.

A descrição de tarefas é a base para um bom projeto de interfaces. Antes de projetar as interfaces é necessário conhecer as informações a seguir [Seb 95]:

- As relações lógicas, funcionais e seqüenciais entre as subtarefas de uma tarefa;
- A lista de informações relacionada às operações de cada tarefa elementar;
- O agrupamento das informações;
- O tipo de atividade do usuário.

São também necessárias informações para tomar decisões mais abstratas tais como:

- A relação de objetivos e sub-objetivos entre subtarefas;
- As relações de sincronismo entre subtarefas;
- O estado-do-mundo para cada subtarefa;

A análise da tarefa consiste no estudo detalhado da natureza da tarefa, seu propósito, as partes que a compõem e a ordem em que devem ser realizadas. Envolve também a compreensão das seqüências de ações necessárias, por que são necessárias, qual o fluxo de informação envolvido, qual o papel do usuário e, qual a distribuição mais eficiente de tarefas entre o usuário e a máquina (o que pode ser automatizado).

A seqüência de passos (subtarefas) necessários para realizar uma tarefa ou solucionar um problema consiste no plano da tarefa ou cenário [Tre 94a]. É importante observar que o planejamento de uma hierarquia de tarefas não impõe o modo como os usuários realizarão a tarefa, mas indica o que está disponível aos usuários para realizar a tarefa [HR 98]. Uma representação de tarefas consiste em uma descrição dos objetos, atributos, ações, pré e pós condições das ações.

Entre as várias formas para representar tarefas encontradas na literatura, destacam-se: TKS [JJ 91], UAN [HH 93] e MAD [SG 89], descritas a seguir.

2.5.1 TKS (*Task Knowledge Structures*)

TKS (*Task Knowledge Structures*) é um método utilizado para identificar, analisar e modelar a estrutura de tarefas [JJ 91]. Permite a representação de diferentes tipos de conhecimentos necessários para o usuário realizar uma tarefa. Este conhecimento é armazenado na memória como estruturas conceituais. O método utiliza diferentes técnicas para coletar informações acerca do conhecimento do usuário com o propósito de construir um modelo de tarefa.

A estrutura da tarefa é descrita através de uma estrutura de quadros contendo: a estrutura hierárquica de objetivos e sub-objetivos, um plano de execução, uma estrutura de procedimentos, condições para execução de um objetivo e os objetos e ações envolvidos na interação.

2.5.2 UAN (*User Action Notation*)

UAN (*User Action Notation*) é uma notação orientada à tarefa e ao usuário, que descreve o comportamento do usuário e do sistema durante a realização de uma tarefa.

Consiste de uma representação de alto nível para a especificação do comportamento do sistema e das ações do usuário [HH 93].

Uma interface é representada como uma estrutura hierárquica de tarefas assíncronas que são seqüenciadas independentemente umas das outras. No nível mais baixo da abstração são representadas as ações do usuário, a realimentação correspondente do sistema e as informações sobre mudanças de estado. Em todos os níveis, as ações do usuário e tarefas são combinadas com relações temporais tais como sequenciamento, entrelaçamento e concorrência para descrever o comportamento do usuário com dependência temporal.

As relações temporais na UAN descrevem como as tarefas do usuário e suas ações relacionam-se no tempo As relações temporais representadas na UAN são: seqüência, iteração, opção, escolha repetida, independência de ordem, interrupção, entrelaçamento, concorrência e espera.

2.5.3 MAD (Método Analítico de Descrição)

O Método Analítico de Descrição - MAD [SG 89] é utilizado para a descrição de tarefas baseando-se em objetos aos quais estão associados estados. São utilizados elementos tais como estados, objetivos, ações, condições e uma estrutura baseada em construtores.

Este método utiliza o conceito de hierarquia de níveis de abstração. A estrutura baseada em objetos associa definições declarativas e procedurais. O conceito de tarefa é representado por um objeto genérico denominado objeto da tarefa - TO, o qual é caracterizado pelos seguintes elementos:

- Um estado inicial (I) – subconjunto do universo de estados, que consiste de uma lista de objetos e parâmetros de entrada para a tarefa.
- Um estado final (F) - subconjunto do universo de estados, que consiste de uma lista de objetos e parâmetros de saída da tarefa. Estes objetos são criados ou modificados durante a execução da tarefa e alguns podem fazer parte tanto da entrada quanto da saída.
- Um objetivo (G) – subconjunto do estado final (F) que estabelece o objetivo a ser alcançado a partir da execução da tarefa.
- Pré-condições (PrC) – conjunto de predicados que estabelece as restrições sobre o estado inicial da tarefa que deve ser satisfeito antes da tarefa ser executada.

- Pós-condições (PoC) - conjunto de predicados que estabelece as restrições sobre o estado final, que deve ser satisfeito após a execução da tarefa.

Os objetos da tarefa pertencem a duas classes: tarefas elementares e tarefas compostas.

- Tarefa elementar – tarefa caracterizada por uma relação simples entre métodos e objetos, isto é, uma ação. A tarefa elementar não pode ser decomposta no nível operacional. Contém um descrição da ação a ser executada.
- Tarefa composta – é uma tarefa cujo nível operacional pode ser caracterizado por uma estrutura que descreve o corpo da tarefa. A estrutura é um elemento da tarefa.

A estrutura é representada por um objeto genérico constituído por um construtor e por seus parâmetros. Os construtores descrevem a organização das tarefas envolvidas. Para MAD estão disponíveis os seguintes construtores:

- SEQ – tarefas seqüenciais, são realizadas uma após a outra, em uma seqüência.
- PAR – tarefas paralelas, a ordem das tarefas não é predefinida.
- ALT – tarefas alternativas (modos alternativos de realizar a mesma atividade)
- LOOP – tarefas iterativas (atividades repetidas)
- OP – tarefas opcionais (atividades que não são obrigatórias)
- SIM – tarefas simultâneas [Seb 95].

Aspectos de sincronismo de tarefa são importantes, uma vez que a interação se caracteriza por processamento de informações assíncronas, no entanto, os construtores acima não consideram casos mais complexos que demandem a representação de restrições de tempo, tais como interrupção e prazos (*deadlines*).

Descritores de Tarefas

No método MAD, existe a associação de descritores aos nós da árvore. No descritor estão representados a identificação da tarefa e os elementos que a compõem: I, F, G, PrC, PoC. A estrutura da tarefa é representada por um dos construtores do método (SEQ, PAR, ALT, LOOP, OP, SIM), seguido dos respectivos parâmetros (uma lista de objetos e tarefas). Um descritor de tarefa, como será utilizado neste trabalho, é representado na Tabela 2-2.

Tabela 2-2: Estrutura Geral dos Descritores

<table>
<tr><td colspan="2">TAREFA: T 1,2,n - Nome da Tarefa</td></tr>
<tr><td>ESTADO INICIAL (I): Relação dos objetos e parâmetros de entrada para a tarefa. (ENTRADA)</td><td>ESTADO FINAL (F): Relação dos objetos e parâmetros de saída da tarefa. (SAÍDA)</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td colspan="2">OBJETIVO (G): Objetivo a ser alcançado a partir da execução da tarefa</td></tr>
<tr><td>PRÉ-CONDIÇÕES (PrC): Conjunto de predicados que estabelece as restrições sobre o estado dos objetos que são necessários para a realização da tarefa.</td><td>PÓS-CONDIÇÕES (PoC): Conjunto de predicados que estabelece as restrições sobre o estado final, que deve ser satisfeito após a execução da tarefa.</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>NÍVEL SUPERIOR: Identificador da tarefa antecessora.</td><td rowspan="2">TAREFA COMPOSTA: Se a tarefa for composta é apresentada a estrutura da tarefa. (Estrutura da Tarefa: Construtor e seus parâmetros).</td></tr>
<tr><td>TAREFA ELEMENTAR: Se a tarefa for elementar contém uma descrição da ação a ser executada.</td></tr>
</table>

Utilização de Trilhas

Originalmente, Scapin propôs a identificação da tarefa na forma de um nome. Sebillotte em [Seb 95] propôs a utilização de um número de modo a identificar o nível de decomposição da tarefa. O número proposto identifica a tupla de geração, isto é, o caminho na árvore que conduz ao nó. O número junto com a tupla de geração define o caminho na estrutura arbórea de representação da tarefa. Com esta notação, os nós podem ser identificados de maneira não-ambígua, por uma trilha que permite a identificação das tarefas antecessoras e sucessoras.

Neste trabalho foi adotado o método MAD para a modelagem de tarefas dada sua simplicidade e adequação aos propósitos de modelagem.

2.6 Requisitos para a Modelagem da Interação

Atualmente, as interfaces incluem estilos de interação seqüencial e assíncrono. No estilo de interação seqüencial, o deslocamento do controle entre usuário e sistema é previsível entre as partes do diálogo. Cada ação leva a um sucessor predefinido: tela, janela, caixa de diálogo, etc. Nesta situação a representação do fluxo de controle tem sido feita através de diagramas de transição de estado.

No estilo de interação assíncrono, muitas tarefas estão disponíveis para o usuário simultaneamente e, o sequenciamento dentro de cada tarefa, independe do sequenciamento das demais. Neste caso, um tipo de técnica de representação do fluxo de controle são os Diagramas de Estado Concorrentes, na qual múltiplos diagramas representam o fluxo de controle ao serem ativados simultaneamente, transferindo o controle entre si.

Palanque e Bastide [PB 94] relacionam as características a seguir como essenciais para modelagem das interfaces: existência de um formalismo, estrutura de dados e de controle, mecanismos de estruturação e capacidade de descrever paralelismo.

- Um sistema de interfaces necessita de uma representação formal que facilite sua especificação[6] e análise, ao invés de uma notação "*ad-hoc*". A necessidade deste formalismo vem da dificuldade em representar a funcionalidade das interfaces que constituem uma classe de sistemas em tempo real.
- Um sistema pode ser melhor compreendido se seus aspectos estruturais (ou estáticos) e comportamentais (ou dinâmicos) puderem ser representados através de uma notação única. Devem estar disponíveis mecanismos para definição de uma estrutura de dados e de uma estrutura de controle.
- O projeto de uma interface usuário-computador precisa ser bem compreendido, reutilizável e aberto para evolução. Mecanismos de estruturação devem estar disponíveis, de forma que os componentes construídos apresentem simplicidade, possibilitando sua expansão ou reutilização em diferentes contextos.
- Na maioria das interfaces, sobretudo onde há a possibilidade de interação com múltiplas janelas, a representação de concorrência é de suma importância, mesmo que o usuário tenha uma interação com o *software* através de um dispositivo puramente seqüencial, a exemplo do teclado.

Com a sofisticação de recursos e técnicas de interação o usuário assumiu uma postura mais ativa do que aquela na qual apenas respondia às demandas do sistema. O usuário passou a definir sua própria seqüência de procedimentos com um alto grau de liberdade na escolha de suas ações. Este novo modelo de interação introduziu um paralelismo na interação antes raramente encontrado. Um exemplo deste modelo é a introdução do conceito de janelas o qual

[6] Uma especificação formal consiste na descrição através de uma linguagem que possua uma sintaxe e semântica bem definida.

viabilizou a execução de tarefas no modo concorrente, isto é, cada tarefa evolui de forma independente, ao mesmo tempo em que sincroniza e coopera informações compartilhadas com as demais.

MacColl e Carrington classificam o software da interface com o usuário como sendo complexo, altamente interativo, multi-modal, concorrente e com requisitos de tempo-real [MC 99].

2.7 MODELOS FORMAIS

A análise formal possibilita a avaliação em cada estágio do ciclo de vida do projeto sem que sejam necessários custos adicionais de implementação, para tanto se fundamenta em modelos. O processo de análise de um modelo pode se fundamentar na sua execução. Para que a versão programada de um modelo possa vir a ser executada é necessário que o modelo tenha um formalismo matemático que represente as principais características das interfaces com o usuário, e disponha de ferramentas computadorizadas para sua construção e análise.

Neste trabalho existe o interesse particular por formalismos que possuam uma representação gráfica (normalmente extensões dos diagramas de transição de estados) para facilitar a compreensão do usuário em um projeto participativo. Uma outra característica necessária ao formalismo desejado é a possibilidade de verificar de forma automática aspectos relacionados à usabilidade da interface.

De posse de um modelo do sistema, o projetista deve ser capaz de verificar a sua consistência de duas formas: a partir de uma análise formal e a partir de uma análise quantitativa [PB 96]. Na análise formal o objetivo é assegurar que não há falhas no modelo. Esta análise pode ser realizada com o uso de ferramentas, que investigam propriedades gerais do modelo tais como:

- Consistência Léxica: está relacionada à existência dos objetos necessários à realização de uma tarefa. Por exemplo, se em um ponto da interface o usuário necessitar selecionar um botão, é necessário que este botão exista no modelo.
- Consistência Sintática: está relacionada ao sequenciamento das ações disponíveis no modelo. Assim, com o modelo deve ser possível realizar qualquer seqüência de ações que venha a ser requisitada.

Por outro lado, na análise quantitativa, o objetivo é realizar medidas de desempenho. Este tipo de análise resulta em dados do tipo: número de ações necessárias para atingir um certo objetivo, freqüência de ações e consistência das restrições temporais do modelo.

Na literatura podem ser ser encontrados vários formalismos que se propõem a representar a interação usuário-computador (gramáticas, diagramas de transição de estados, etc) [Shn 98].

2.7.1 StateCharts

Um exemplo de formalismo utilizado para representar as características de interfaces são os grafos *Statecharts*. Este formalismo consiste de uma linguagem visual para a especificação formal do comportamento de sistemas. Um grafo *statechart* é essencialmente um diagrama de estados finitos, estendido ao qual foi acrescentada a representação de hierarquia, concorrência, propriedades visuais e sincronização. Semelhante às máquinas de estado finito, *statecharts* representam o sistema através de um conjunto discreto de estados que representam o comportamento do sistema [Dru 97].

2.7.2 Redes de Petri

Redes de Petri é um outro formalismo utilizado na construção de modelos que representam as características das interfaces.

A escolha de redes de Petri como formalismo para modelar interfaces decorre de sua capacidade de descrever as características das interfaces tais como paralelismo, concorrência, assincronismo, não-determinismo. Estas redes possibilitam descrever eventos de forma concorrente e assíncrona, típicos de interfaces gráficas. São adequadas para especificar como os estados do sistema se modificam em conseqüência das ações do usuário. Apresentam uma representação formal e são passíveis de execução (que possibilita simulação). Além de um formalismo matemático, as redes de Petri possuem uma representação gráfica e permitem simular o comportamento do sistema modelado.

Avaliar aspectos de corretude, através da análise de propriedades da rede são típicos no formalismo de redes de Petri. Por exemplo, um dos aspectos da completitude da interface pode estar relacionado com a alcançabilidade do estado final (saída do sistema).

A descrição de sistemas complexos em redes de Petri, tais como as interfaces, com seus inúmeros estados e estruturas repetidas (a exemplo de janelas, botões e menus), torna-se bastante extensa em termos da estrutura da rede. Extensões de rede de Petri foram propostas de forma a introduzir mecanismos para simplificar os modelos obtidos a partir do enriquecimento de seu poder de descrição. Estas extensões constituem uma classe de rede de Petri denominada redes de Petri de alto nível [Mur 89].

Nas redes de alto nível, as fichas que denotam o estado do sistema podem representar tipos estruturados de dados ao invés da simples indicação binária que a presença da ficha representa nas redes clássicas. Esta modificação no significado da ficha permite que os modelos de sistemas complexos sejam simplificados através da fatoração de subestruturas semelhantes capazes de reconhecer as fichas sobre as quais atuam.

Uma das abordagens de alto nível é a rede de Petri Colorida (*CPN Coloured Petri Nets)* [Jen 92]. CPN é uma classe de redes de Petri de alto nível que se aplica à especificação, projeto, simulação, validação e implementação de sistemas de software. Estas redes dispõem de métodos formais para análise, verificação e validação.

No nosso contexto do trabalho, estas redes são particularmente interessantes devido à possibilidade de representar através de uma estrutura única sistemas nos quais o comportamento varia de acordo com o contexto, além da disponibilidade no nosso ambiente de trabalho de ferramentas de suporte a aplicação dos métodos formais citados. Assim este foi o formalismo escolhido para modelar o comportamento das interfaces neste trabalho.

A adequação destas redes à modelagem de interfaces pode ser constatada através de sua extensa utilização [Sch 95] [SE 96][RPC 96][RPC 98][PBSBD 93][PBS 95][PB 96].

Schlungbaum em [Sch 95][SE 96] utiliza uma notação própria para a modelagem de sistemas de interface com base em redes de Petri Colorida.

Fiorella em [RPC 96][RPC 98] utiliza redes de Petri colorida para construir um formalismo visual denominado XDM que suporta o projeto e avaliação da interação de sistemas em contextos específicos.

Palanque em [PBSBD 93] propõe uma extensão de redes de Petri – PNO (*Petri Net with Objects*) para modelar o diálogo em interfaces. Em [PBS 95] é proposto o formalismo ICO (*Interactive Cooperative Objects*) para construir o modelo de um sistema, o qual também é baseado em redes de Petri. Em [PB 96] é proposto um ciclo de vida de software com base no formalismo ICO.

Os modelos de interfaces construídos com o formalismo de redes de Petri normalmente são dependentes de um contexto, dificultando a reutilização do modelo para contextos diferentes.

A seguir passamos a descrever sucintamente as redes CPN.

Rede de Petri Colorida (CPN – *Coloured Petri Nets*)

Uma CPN é composta de uma estrutura, um conjunto de inscrições e um conjunto de declarações. A estrutura é um grafo dirigido e bipartido, no qual ao invés de associar-se pesos aos arcos, são utilizadas inscrições que correspondem a expressões. As expressões determinam dinamicamente, na ocorrência de uma transição, quantas e quais fichas podem ser removidas ou adicionadas aos lugares associados. As expressões são construídas a partir de constantes, variáveis e operadores previamente definidos. O conjunto de declarações de uma CPN serve para declarar a natureza dos elementos citados nas inscrições e portanto dos objetos representados no modelo. As inscrições e declarações são escritas na linguagem CPN-ML [UAA 96b].

Em uma CPN podem ser utilizadas inscrições associadas às transições, denominadas guardas. As guardas são expressões de natureza booleana e têm a função de restringir a ocorrência da transição. As guardas são representadas graficamente por inscrições entre colchetes ao lado das transições. As fichas presentes no modelo CPN transportam um valor, denominado cor, que pertence ao domínio do tipo de dados definido nas declarações. Assim, cada lugar na estrutura é associado a um conjunto de cores que indica o tipo de ficha que o lugar deve conter.

Uma marcação para uma CPN é uma distribuição de fichas nos lugares. Em cada marcação uma transição é dita habilitada se todos os seus lugares de entrada contiverem fichas suficientes para satisfazer às expressões dos arcos. Cada expressão é então avaliada para determinar quantas e quais fichas são necessárias nos lugares de entrada. Caso a

transição dispare, fichas são retiradas dos lugares de entrada e novas fichas são depositadas nos lugares de saída. A quantidade de fichas removidas/depositadas é função da avaliação das expressões dos arcos. Uma representação visual de uma CPN é apresentada na Figura 5.2 e uma descrição formal destas redes pode ser encontrada em [Jen 92].

A análise de redes de Petri se baseia na construção do espaço de estados (grafo de alcançabilidade), ou em técnicas algébricas lineares (equações de estado) [Mur 89]. Um grafo de alcançabilidade representa o conjunto de estados alcançáveis, e se presta para verificar um conjunto de propriedades da rede relativos ao contexto modelado.

Uma vez que uma rede de Petri pode ser representada por uma matriz de incidência e suas marcações por um conjunto de vetores, esta representação pode ser utilizada para caracterizar a dinâmica do sistema. Da matriz de incidência e dos vetores são derivadas equações algébricas lineares cujas soluções características são os lugares e as transições.

As redes de Petri são consideradas adequadas para a modelagem funcional e dinâmica de sistemas de interface por (1) possibilitarem a verificação da especificação, a análise e a simulação de modelos; (2) porque o comportamento dinâmico dos sistemas representados por elas pode ser visualizado e discutido mais facilmente com os usuários; (3) porque sua representação gráfica pode ser utilizada como documentação e principalmente por (4) sua fundamentação matemática que possibilita uma análise computacional dos modelos representados.

2.8 AVALIAÇÃO DE INTERFACES

O foco da avaliação de uma interface usuário-computador é a análise da interação do usuário com o sistema, a fim de adquirir informações sobre como os indivíduos interagem com o sistema e quais os problemas mais comuns.

A avaliação de interfaces pode ser realizada a partir de diferentes métodos e em fases distintas do ciclo de vida dos produtos [Tre 94b] [HH 93] [Shn 98] [Nie 93] [QT 97]. Pode ser realizada de maneira informal ou através de um processo rigoroso de testes. Seus resultados podem ser subjetivos, objetivos, ou ainda uma combinação destas duas formas. Ela pode se

fundamentar apenas na opinião do projetista através de uma avaliação heurística, no comportamento do usuário ou na sua opinião.

Há registros de que uma média de 48% dos recursos gastos no desenvolvimento de sistemas sejam utilizados na concepção e desenvolvimento da interface com o usuário, no entanto, deste montante apenas um pequeno percentual é dedicado à avaliação [Wil 94].

O processo avaliatório de interfaces pode ocorrer em diferentes fases do projeto e ao longo de todo o ciclo de vida do produto, levando a um contínuo refinamento. No entanto, pode ser bem mais eficiente se introduzido durante o desenvolvimento do software. De acordo com a época em que ocorre, a avaliação pode ser classificada como: Formativa e Somativa [HH 93].

A avaliação formativa ocorre antes que o sistema esteja concluído, começa no início do projeto e prossegue continuamente ao longo do seu desenvolvimento. Por outro lado, a avaliação somativa ocorre após a conclusão do projeto. A avaliação somativa é utilizada durante testes beta ou na comparação de um produto com outro similar.

A avaliação formativa é realizada em cada etapa de projeto, produzindo informações quantitativas que auxiliam os projetistas a comparar e estabelecer especificações quanto à usabilidade do sistema. Os dados resultantes podem ser utilizados para a tomada de decisões quanto ao refinamento da interface. Segundo Hix em [HH 93], a avaliação formativa produz os seguintes tipos de informação:

- Objetiva: consiste de medidas que podem ser diretamente observadas, tipicamente representam o desempenho do usuário enquanto utiliza a interface.
- Subjetiva: representa opiniões, usualmente do usuário, a respeito da facilidade de utilização da interface.
- Quantitativa: representa dados numéricos, tais como medidas ou estatísticas de desempenho. Os dados obtidos são de fundamental importância para verificação da convergência entre os testes e a especificação do produto.
- Qualitativa: consiste de resultados não numéricos, tais como lista de problemas encontrados pelos usuários durante a interação, e suas sugestões para melhoria do sistema.

Scapin em [SB 92] classifica os métodos de avaliação de interfaces, segundo a origem dos dados da avaliação, nas três categorias descritas a seguir.

Os métodos de avaliação baseados no desempenho do usuário. Estes métodos geram grandes volumes de informação tanto qualitativa quanto quantitativa que vão desde a opinião do usuário sobre o produto até o registro automático, ou não, de dados da interação, tais como: seqüências de execução de tarefas, seqüências alternativas, seqüências mais freqüentemente utilizadas e o tempo gasto na execução de cada tarefa. A eficácia desses métodos, quanto aos resultados da análise, depende da sistematização e representação dos dados disponíveis.

Os métodos de avaliação baseados em modelos teóricos objetivam prever o desempenho do usuário durante sua interação com o sistema.

E, os métodos baseados no julgamento do especialista que dependem da capacidade de análise do avaliador, o qual se apoia em critérios derivados de princípios e diretrizes de projeto de interfaces, padrões, além de sua própria experiência em avaliação.

Neste trabalho adotaremos a visão de Scapin, e recorremos aos métodos baseados no desempenho do usuário, com base no registro automático de dados e em métodos baseados em modelos teóricos da interação que objetiva antecipar falhas e corrigi-las.

2.8.1 Objetivos da Avaliação

Para que o processo de avaliação seja eficaz, seus objetivos devem estar sempre claros. Os objetivos da avaliação podem ser expressos sob a forma de questões que devem ser respondidas pelo projetista quando da análise de uma interface [Tre 94b]. Estas questões podem possuir graus de abstração bastante diferentes. A título de exemplo são apresentadas a seguir algumas questões típicas:

- A interface é ou não utilizável?
- Por que algumas das opções de navegação na interface não estão sendo utilizadas?
- Quais são as dificuldades freqüentemente encontradas pelos usuários?
- O desempenho do usuário corresponde à sua opinião sobre o produto?
- A apresentação das informações para o usuário pode ser melhorada?
- A interface permite ao usuário realizar sua tarefa de forma correta e eficiente?

A resposta a estas questões é função do método de avaliação utilizado. Na literatura podem ser encontrados vários métodos para a avaliação de interfaces, geralmente referenciados como métodos para avaliação da usabilidade (*Usability Evaluation Methods*) [Nie 93]. A seção a seguir descreve a classe de métodos baseados na observação, uma vez que são de interesse para este trabalho.

2.8.2 Métodos de Avaliação baseados no desempenho do usuário

A análise de protocolos corresponde ao registro do desempenho do usuário enquanto utiliza a interface com o objetivo de obter dados quantitativos acerca do desempenho dos participantes enquanto estes realizam suas tarefas. Os dados obtidos são extremamente importantes tanto para comparação entre várias sessões de utilização quanto para comparação entre diferentes usuários ou produtos. Este registro pode ocorrer através da utilização de lápis e papel, de áudio, vídeo, ou através de um registro (arquivo de *log*) utilizando o próprio computador e um software auxiliar de coleta.

O processo de observação direta envolve um ou mais avaliadores para o registro do desempenho do usuário durante a realização de uma tarefa definida. O observador pode ser ativo, questionando ou auxiliando o usuário quando da ocorrência de dificuldades, ou passivo. Uma desvantagem da observação, é que observadores humanos não são muito eficientes no processo de monitoração, podem facilmente distrair-se, além de apresentar dificuldade em registrar em tempo hábil, todas as atividades do usuário. Para solucionar estes problemas podem ser realizadas gravações de áudio e vídeo para registrar as ações do usuário, e servir como uma fonte para consulta posterior.

A observação é uma técnica poderosa que permite investigar como os usuários realizam suas tarefas, ou como as tarefas são decompostas em sub-tarefas. A desvantagem desta técnica é o seu caráter intrusivo que pode alterar o comportamento do usuário.

O registro em Áudio possibilita coletar informações de como o usuário realizou as tarefas. Porém, muitas vezes os dados coletados são insuficientes para realizar um diagnóstico. Para resolver este problema pode-se associar ao registro de áudio, o registro em vídeo e a captura automática das ações do usuário.

O Registro em Vídeo da interação, é uma técnica baseada no registro do desempenho do usuário que se mostra eficiente para realizar a análise de uma interface uma vez que podem existir muitos eventos simultâneos, ou eventos que acontecem tão rapidamente que ficam difíceis de serem percebidos e registrados pelo observador. Esta técnica no entanto apresenta limitações devido ao tempo prolongado de análise dos resultados e ao custo elevado das gravações e editoração do material. Portanto, sua utilização só é justificada em aplicações críticas e apenas para partes selecionadas da avaliação. É recomendada apenas após observações gerais que permitam isolar o problema, e deve ser aplicada apenas a trechos específicos da interação sem que constitua a fonte principal dos dados a serem capturados.

A escolha de um método deve se fundamentar no propósito da avaliação, nos objetivos específicos, no escopo da avaliação, nos aspectos a serem avaliados e nas técnicas e ferramentas de suporte ao processo avaliatório [Tre 94b]. Observa-se assim a complexidade da escolha com que se depara um avaliador de interfaces. A seguir são apresentadas alguns métodos e ferramentas para avaliação de interfaces.

2.8.3 Métodos e Ferramentas para a Captura de Dados da Interação

O processo de captura de informações diretamente da aplicação (*Software Logging*) se baseia na coleta automática, através do sistema computacional, de informações sobre como os usuários realizam as suas tarefas com o sistema sob avaliação. Dependendo dos objetivos, a coleta de dados pode ser realizada para um grande número de usuários em diferentes circunstâncias. Contudo, um problema na utilização desta técnica decorre do volume de informações que pode ser coletado e do tempo necessário para analisá-las, chegando a ser inviável uma análise sem o auxílio de ferramentas computacionais.

Tipicamente, o registro (*log*) da interface contém informações estatísticas acerca da freqüência de utilização de cada característica da interface. As informações coletadas podem revelar quais as características (módulos do sistema, cenários) que estão sendo mais utilizadas e identificar aquelas que estão sendo raramente utilizadas. Pode-se também observar a freqüência com que várias situações de erro e solicitações de ajuda acontecem, com o objetivo de comparar versões de protótipos ou de sistemas já desenvolvidos. No contexto desta pesquisa, esta técnica de avaliação será utilizada com o objetivo de coletar dados sobre a interação do usuário com protótipos.

Existem no mercado ferramentas para construção de protótipos com uma variedade de recursos para a construção interativa das interfaces. Todavia, a avaliação da usabilidade dos protótipos é uma tarefa ainda pouco contemplada pelos projetistas destes produtos, sendo raros os casos onde existem recursos para sua validação. Esta constatação conflita com recomendações encontradas na literatura, como o que sugere Shneiderman em [Shn 98], que afirma ser essencial a incorporação de mecanismos de avaliação nas ferramentas de desenvolvimento, com o objetivo de auxiliar os projetistas de interface a criar sistemas consistentes e com um maior grau de usabilidade.

A construção de um protótipo tem em essência o caráter transitório. O protótipo deve ser avaliado, e se forem encontrados problemas, estes devem ser corrigidos levando a um constante refinamento, o qual deve ser repetido até que soluções satisfatórias sejam obtidas.

Para que um protótipo possa ser considerado utilizável, ele deve proporcionar aos usuários um meio eficiente de realizar as tarefas para o qual foi projetado. Uma outra medida, não menos importante, é a medida da satisfação do usuário. Entretanto, a medida desta satisfação é subjetiva e bastante complexa, pois é uma função dos próprios usuários, das tarefas a serem realizadas e do ambiente de trabalho, e portanto não será considerada parte do escopo deste trabalho, que se concentrará na validação do protótipo do ponto de vista de sua usabilidade com base em um conjunto de métricas que serão discutidas na Seção 6.2.1.

Neste trabalho, a avaliação quantitativa se fundamenta na coleta de dados objetivos sobre a interação. Para tanto é utilizada uma ferramenta para coleta de dados oriundos da especificação da interface e da interação do usuário com o protótipo.

Método *Playback*

Uma das primeiras iniciativas de construção de ferramentas para *software logging* que merece destaque é aquela desenvolvida pela IBM na década de 80 [NS 84]. No Centro de Fatores Humanos da IBM foi desenvolvida a metodologia *Playback*, que destina-se a realizar um processo de avaliação de software e/ou documentação. A idéia central é que, enquanto um usuário está trabalhando com o sistema, a atividade do teclado é marcada, cronometrada e gravada em um segundo computador. Esse *log* gravado da interação é depois repassado pelo sistema hospedeiro para observação e análise do avaliador.

A avaliação da usabilidade é dividida em sessões de testes separadas. Uma sessão pode consistir de leitura e execução de exercícios de um capítulo de um tutorial, ou pode incluir a execução de uma tarefa para realização de testes de desempenho. Todas as digitações do usuário e o tempo cumulativo da sessão (em milisegundos) são gravados. Também são gravados, durante a sessão, os códigos e comentários do experimentador (juntamente com o tempo associado). Além disso, são coletadas as seguintes estatísticas e registradas na conclusão da sessão: tempo de início da sessão até a primeira digitação do usuário, tempo de início da sessão até a última digitação, tempo cumulativo durante a utilização de ajuda, tempo total da sessão, freqüência de uso de cada tecla de função e número de vezes que foi solicitada ajuda.

Projeto MUSiC

Um conjunto de métodos propostos para medir a usabilidade de sistemas foi desenvolvido pelo projeto europeu MUSiC (*Metrics for Usability Standards in Computing*) [Bev 97]. Estes métodos se destinam à especificação e avaliação da qualidade no uso e se fundamentam na realimentação destas informações sobre o projeto do sistema. Estes métodos são apoiados por um conjunto de ferramentas e técnicas que se fundamentam nos princípios da norma ISO 9241-11, a qual especifica as medidas de desempenho e satisfação do usuário. Uma de suas ferramentas de suporte, DRUM [MR 93], oferece recursos para coleta e análise de dados em vídeo. Outra ferramenta utilizada no MUSiC consiste em um questionário para avaliação da satisfação do usuário SUMI [KC 93].

Os métodos do projeto MUSiC, avaliam a extensão com que objetivos específicos de tarefas são alcançados e o tempo gasto para atingir estes objetivos. São também obtidas medidas do tempo gasto improdutivamente (por exemplo solucionando problemas e buscando ajuda), e dados acerca da localização destes problemas. Estas informações constituem um diagnóstico que ajuda a identificar onde estão problemas específicos e quais as mudanças necessárias.

No MUSiC, a eficácia com que um usuário utiliza um produto é medida pelo percentual completado da tarefa e pelo percentual de qualidade dos objetivos alcançados (o grau com que foram atingidos os objetivos da tarefa). Este valor muitas vezes é obtido a partir da média de valores da avaliação de sub-tarefas. A medida de eficiência relaciona a eficácia com o dispêndio de recursos (temporal, físico, intelectual, financeiro). Um exemplo de

eficiência é a 'eficiência temporal' que relaciona a eficácia com o dispêndio de tempo na realização da tarefa [Bev 97]. Estas medidas podem ser utilizadas para comparar:

- Produtos similares ou versões de um mesmo produto utilizadas pelo mesmo usuário realizando a mesma tarefa.
- Diferentes tipos de usuário de um mesmo produto.
- Diferentes tarefas realizadas pelo mesmo usuário no mesmo produto.

QC/Replay & USINE

Um método de avaliação baseado na análise do modelo de tarefa e em testes de usabilidade foi proposto por Lacerot e Paternó [LP 98]. Este método de avaliação consiste no mapeamento de ações físicas realizadas pelo usuário em tarefas representadas no modelo da tarefa. O método se baseia na hipótese de que a análise de tarefas e análise de erros cometidos durante a interação com um produto podem sugerir como melhorar a interface. Um dos propósitos do método foi reduzir o envolvimento do avaliador com a coleta de dados a partir do uso de ferramentas automáticas de coleta.

O método é apoiado por duas ferramentas: USINE e QC/Replay. O QC/Replay é um software comercial que registra a interação do usuário com objetos da interface (botões, menus, listas, etc.) e as ações elementares (pressionamento do mouse, de teclas, etc.). O registro é feito na forma de scripts que podem ser posteriormente "executados" para reproduzir a interação. Os registros do *log* são armazenados em um arquivo texto que pode ser editado pelo avaliador. A ferramenta USINE (*USer INterface Evaluator*) possibilita a associação entre ações do usuário no modelo da tarefa com ações do usuário no arquivo de *log*. Um outro aspecto interessante é a possibilidade de apresentar os resultados na forma texto ou na forma de gráficos.

Sherlock

Um outro trabalho que merece destaque é o *Sherlock* [MS 97], uma ferramenta de análise de consistência, que suporta a avaliação da distribuição espacial de objetos na tela e da terminologia utilizada nos diálogos. Apresenta de forma gráfica e compacta uma visão das propriedades visuais de caixas de diálogo. Inclui a análise de botões, dos pontos de vista: *layout*, cores, tamanhos e localização, para determinar a consistência do projeto. Se

fundamenta na evidência de que a consistência facilita a percepção humana e processos cognitivos tais como reconhecimento e memorização.

XVT

Uma solução comercial voltada para o desenvolvimento e avaliação de interfaces é o XVT [BSF 99]. Este produto é composto por ferramentas multi-plataformas, a exemplo de *Microsoft Windows*, *Windows NT*, *Windows 95*, *Macintosh*, *OS/2*, *OSF/Motif*. A ferramenta DSC, possibilita o desenvolvimento de sistemas de forma rápida. Ela possui um módulo interativo de construção de interfaces gráficas, e um gerador de código em linguagem C. As interfaces geradas podem ser manipuladas em qualquer plataforma XVT.

Neste mesmo contexto, existe uma outra ferramenta denominada *TRACK Defects* que realiza a monitoração de problemas no ambiente *Windows*. Esta ferramenta possibilita a monitoração de erros, gerenciamento de projetos, alterações de código, teste de produtos, atualização de versões, configurações do sistema, etc. Há também *TRACKWeb* que possibilita a administração de uma base de dados, acesso a informação, consultas e, geração de relatórios a partir de um navegador de páginas *web*. A ferramenta *TRACK Defects* foi desenvolvida para o gerenciamento de erros no ambiente Windows. Seu propósito é registrar a ocorrência dos erros de hardware ou software com o objetivo de facilitar o processo de manutenção de produtos interativos já em utilização. Esta ferramenta apresenta uma base de soluções para os defeitos a qual pode ser manipulada pela *internet* além de permitir a notificação automática de indivíduos que possam apoiar a solução de problemas.

Observer

O *Observer* [Nol 99] é um sistema destinado a coletar, analisar, apresentar e gerenciar dados oriundos de observação. Pode ser utilizado para registrar atividades, atitudes, movimentos, expressões faciais, interações sociais e qualquer outro aspecto do comportamento humano.

Projetado para capturar informações durante a interação, é o sucessor do lápis e papel. É possível realizar anotações diretamente no computador, ou codificar eventos em vídeo ou mídia digital. Antes da realização da coleta é utilizado o módulo de configuração, onde é possível especificar o que se deseja observar. Para cada projeto de observação, pode ser criada

uma configuração diferente. O módulo de registro de eventos realiza as observações, gravando automaticamente os dados em arquivos de observação.

Durante uma sessão de observação, o pressionamento das teclas é registrado, além do tempo de ocorrência, sendo possível também utilizar o mouse ou uma caneta como dispositivo de entrada. Durante as observações, pode-se tomar notas e realizar comentários, os quais são armazenados juntamente com os dados coletados.

Para uma análise detalhada, observações do comportamento são freqüentemente registradas em vídeo. O *Observer Video-Pro*, que possui capacidades de multimídia, é composto por componentes de software e hardware. Ele realiza a leitura dos eventos diretamente a partir do vídeo, possuindo funções de procura e edição.

Após a conclusão da coleta de dados, estão disponíveis funções de análise com as quais é possível verificar os dados através de tabelas de evento (uma lista cronológica com todos os eventos registrados), ou gerar relatórios com estatísticas com a freqüência e duração das atividades, além da estrutura seqüencial do processo ou da ocorrência dos eventos. Os resultados podem ser mostrados na tela, impressos em papel ou salvos em disco.

Ferramentas FAI

A construção de Ferramentas para Avaliação de Interfaces (FAI) no âmbito da Universidade Federal da Paraíba, iniciou com a construção de um sistema de prototipagem rápida de interfaces - AGILE, apresentado em [Pro 92] e [San 92], o qual previa o desenvolvimento de uma ferramenta para avaliação de interfaces neste ambiente. Esta ferramenta foi posteriormente desenvolvida por Nakayama [Nak 95].

As ferramentas desenvolvidas para *software logging* tipicamente não apresentam portabilidade, uma vez que a coleta das informações é realizada por um sistema de software de baixo nível, tal como os *drivers* de mouse e de teclado, ou através da modificação do software sob avaliação ou ainda por uma combinação destes. A modificação do software é uma forma mais eficiente por permitir realizar uma coleta seletiva, registrando apenas a ocorrência de eventos de interesse.

Por esta razão foram desenvolvidas na Universidade Federal da Paraíba ferramentas específicas para os ambientes DOS, Windows e Unix. A FAIDOS, a FAIWindows e a

FAIUnix respectivamente. Todas estas ferramentas têm características semelhantes, mas são voltadas para ambientes operacionais e hardware específicos.

A FAIDOS [Nak 95], é uma ferramenta para coleta de dados baseada na técnica de *software logging* construída para a validação de protótipos construídos no ambiente AGILE, o qual por sua vez trabalha no ambiente MS-DOS. O AGILE é um sistema de prototipagem de interfaces, que foi desenvolvido com o propósito de oferecer recursos aos projetistas de interface para construção e simulação de protótipos de interfaces do tipo texto com base em quatro tipos de diálogos: menu, pergunta e resposta, comando e formulário.

Na ferramenta desenvolvida por Nakayama, FAIDOS, o módulo coletor de dados, realiza dois tipos de coleta. O primeiro consiste na leitura do arquivo de protótipo que contém todas as informações da interface em análise, coletando informações tais como: os tipos de diálogos, as teclas utilizadas nos diálogos, os comandos disponíveis, etc. Este tipo de coleta é denominado de coleta estática. A segunda forma de coleta, denominada coleta dinâmica, acontece durante a interação do usuário com o protótipo e registra o tempo de execução de uma tarefa, o número de vezes que a ajuda foi solicitada durante um diálogo, o total de erros cometidos em cada diálogo, o tempo de execução de uma tarefa, o tempo total da sessão de avaliação, o total de erros cometidos na sessão, etc.

Devido à grande popularidade do ambiente *Windows*, foi desenvolvida uma outra ferramenta para este ambiente, a FAIWin descrita em Diniz [Din 96]. A FAIWin é uma ferramenta para coleta de dados baseada na técnica de *software logging* desenvolvida para a validação de protótipos construídos para o ambiente *Windows*.

Esta ferramenta permite o registro de dados estáticos sobre a especificação do protótipo e dados dinâmicos sobre a interação com o usuário. O módulo coletor estático registra dados do tipo: relação de cores utilizadas nas janelas, número total de cores em cada janela, total de objetos em cada janela, etc. Por outro lado, o módulo coletor dinâmico, registra informações oriundas da interação do usuário com o protótipo. Dentre estas informações destacam-se: taxa de utilização de comandos e taxa de solicitação de ajuda.

A ferramenta FAIUnix foi desenvolvida no contexto deste trabalho e será apresentada no Capítulo 6.

2.9 CONCLUSÕES

Para alguns projetistas de interface, o processo de avaliação pode não alcançar os resultados desejados, contudo os resultados obtidos da avaliação podem representar indicações valiosas sobre problemas existentes no projeto, além de apontar o caminho de novas alternativas de interação. Soluções que possam parecer insatisfatórias podem ser revistas e os problemas associados podem ser eliminados a cada novo ciclo de projeto. Assim, no capítulo seguinte será apresentado um método de concepção de interfaces que contempla a avaliação desde os primeiros estágios do processo de concepção.

3 MÉTODO PARA CONCEPÇÃO DE SOFTWARE INTERATIVO

A concepção da interface do software interativo envolve o sequenciamento de tarefas e ações do usuário, a realimentação do sistema, o projeto de telas e a funcionalidade do sistema, o conteúdo e métodos de acesso à informação, o projeto de objetos, *layout* da tela e estilos de interação. Assim, o projeto da interação se desenvolve no nível conceitual e no nível de percepção. O projeto no nível conceitual consiste em identificar as funções necessárias, sequenciar estas funções e definir o fluxo da interação. O projeto no nível perceptivo, consiste no projeto da representação visual para o usuário.

Neste Capítulo será apresentado um método para o projeto da interação que aborda tanto o nível conceitual quanto o nível perceptivo. Este método se fundamenta nas atividades de análise de modelos informais, na verificação de modelos formais e na validação de protótipos.

3.1 DESCRIÇÃO DO MÉTODO

O método aqui apresentado adota a prototipação e iteratividade do modelo espiral. Modelos formais e informais são utilizados para melhor definir o problema e refinar os requisitos da interface de um produto.

O método se desenvolve em um conjunto de etapas que se apoiam na construção e avaliação dos modelos. A relação entre os domínios dos modelos adotados e as etapas do método é ilustrada na Figura 3.1.

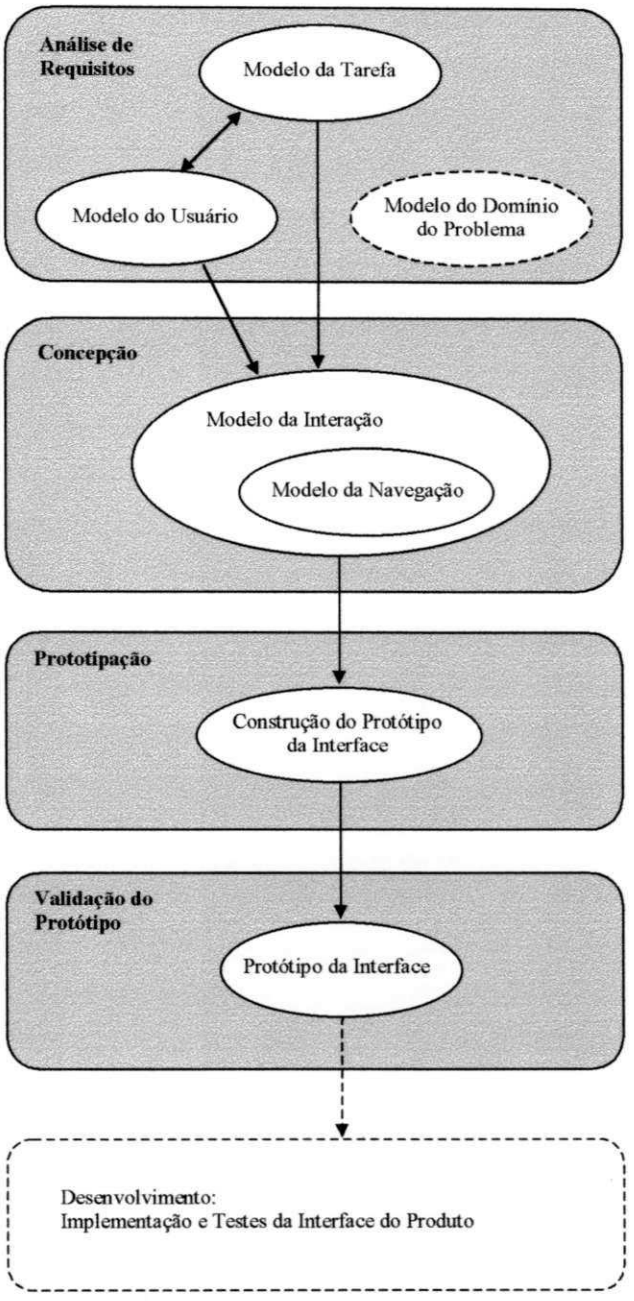

Figura 3-1: Relação entre os domínios dos modelos e etapas do método

No desenvolvimento de interfaces, a análise de requisitos se apoia na análise de necessidades, análise do usuário, análise da tarefa e na análise funcional. A análise de requisitos define os objetivos básicos, os propósitos e as características desejadas para o sistema, resultando em um modelo do domínio do problema que descreve o que usuário será capaz de fazer com o sistema.

Uma vez que a formalização da especificação auxilia na compreensão do *software*, o método sugere o conceito da verificação formal da especificação antes da construção de protótipos, antecipando problemas de usabilidade que se propagariam para o protótipo e subseqüentemente para o produto que dele evoluísse.

A combinação da verificação de modelos e a validação de protótipos resulta no aumento da capacidade de análise do método permitindo confrontar resultados destas duas abordagens.

O método se apoia na avaliação de cada uma das etapas que compõem a especificação da interface. As etapas não apresentam restrições de interdependência ou de seqüência, salvo na primeira iteração. Por exemplo, não se pode construir um protótipo para o qual não tenha sido analisada e modelada a tarefa.

3.2 APRESENTAÇÃO DAS ETAPAS

As etapas do método se desenvolvem como mostra a Figura 3-1. Estas etapas compreendem: a construção e avaliação do modelo da tarefa, construção e avaliação do modelo da interação, construção e verificação do modelo da navegação e construção e validação do protótipo.

Semelhantemente ao ciclo de vida no modelo estrela, neste método, o ciclo de concepção é centrado na avaliação, de tal modo que os resultados de cada etapa são avaliados antes de se prosseguir para a próxima etapa.

Na Figura 3-2 é mostrada a relação entre os domínios dos modelos, e as ferramentas de suporte utilizadas nas diferentes etapas do método. A ferramenta *Design/CPN* é utilizada durante a avaliação do modelo da navegação, verificando aspectos do modelo construído em redes de Petri colorida. A ferramenta *DevGuide* é utilizada para construção do protótipo da interface, e a ferramenta FAIUnix é utilizada para avaliar os aspectos estruturais e comportamentais da interface. As ferramentas fornecem suporte a realização das etapas e serão apresentadas em detalhes nos capítulos seguintes. O método é composto pelas etapas descritas a seguir.

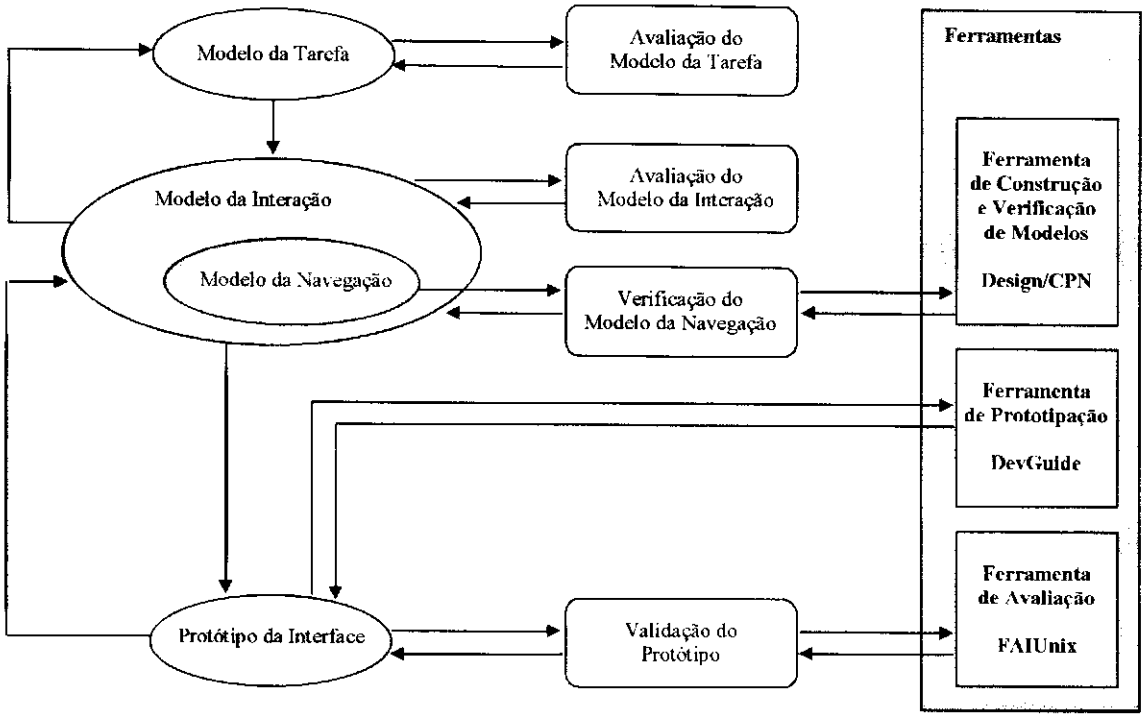

Figura 3-2: Ciclo de Concepção no Método

O algoritmo que descreve as etapas da Figura 3-2, que será detalhado a seguir no texto, consiste em:

Início da Concepção

Etapa 1. Análise das características do usuário (Levantamento do seu perfil)

Etapa 2. Construção do Modelo da Tarefa

Etapa 3. Avaliação da Consistência do Modelo da Tarefa:

Se (requisitos satisfatórios) então continua na (Etapa 4)

Senão retorna para (Etapa 2)

Etapa 4. Concepção do Modelo da Interação

Etapa 5. Avaliação da Consistência e Completitude do Modelo da Interação:

Se (requisitos satisfatórios) então continua na (Etapa 6)

Senão retorna-se para (Etapa 4) e modifica-se o Modelo da Interação ou retorna-se para (Etapa 2) e modifica-se o Modelo da Tarefa (caso as alterações no modelo da interação não sejam suficientes).

Etapa 6. Construção do Modelo da Navegação

Etapa 7. Verificação do Modelo da Navegação:

Se (requisitos satisfatórios) então continua na (Etapa 8)

Senão retorna-se para (Etapa 6) e modifica-se o Modelo da Navegação ou retorna-se para (Etapa 4) e modifica-se o Modelo da Interação (caso as alterações no modelo da navegação não sejam suficientes) ou retorna-se para (Etapa 2) e modifica-se o Modelo da Tarefa (caso as alterações no modelo da interação ainda não tenha sido suficientes)

Etapa 8. Construção do Protótipo

Etapa 9. Validação do Protótipo quanto a Aspectos Ergonômicos do Projeto Visual:

Se (requisitos satisfatórios) então continua na (Etapa 10)

Senão retorna-se para (Etapa 8) e modifica-se apenas a estrutura dos diálogos

Etapa 10. Preparação do Protótipo para Validação

Etapa 11. Validação do Protótipo junto ao Usuário Final:

Se (requisitos satisfatórios) então finaliza-se o processo

Senão retorna-se para (Etapa 4) e modifica-se o Modelo da Interação ou retorna-se para (Etapa 2) e modifica-se o Modelo da Tarefa (caso as alterações no modelo da interação não sejam suficientes)

Fim da Concepção

3.2.1 Análise das Características do Usuário.

A análise das características do usuário define a classe de usuários em termos das tarefas que serão realizadas e do conhecimento necessário para realizá-las. O resultado é um conjunto de definições denominado "perfil do usuário" que contém as características relevantes para a interação. Este perfil é obtido a partir da observação do usuário em seu ambiente de trabalho, se possível realizando tarefas similares, ou ainda a partir de questionários ou entrevistas com representantes do grupo de usuários. A análise do usuário se fundamenta no modelo sintático-semântico [Shn 98] descrito no Capítulo 2.

Entrada	Saída
Resultado do levantamento do perfil dos usuários (questionários, entrevistas, observação)	Perfil do usuário (descrição textual das características do usuário relevantes ao projeto)

3.2.2 Construção do Modelo da Tarefa

A análise da tarefa possibilita obter uma descrição completa das tarefas, subtarefas e métodos envolvidos na utilização do sistema. No método proposto, a análise da tarefa é realizada utilizando-se o método de descrição MAD [SG 89], descrito no Capítulo 2. A análise resulta em uma decomposição *top-down* com uma descrição detalhada da tarefa, envolvendo as seqüências necessárias, o fluxo da informação e o papel do usuário no procedimento, conduzido a uma decisão sobre o que pode ser automatizado.

Entrada	Saída
Perfil do usuário, descrição de funcionalidades requeridas no sistema	Modelo da tarefa (descrição gráfica e textual da tarefa em MAD)

3.2.3 Avaliação da Consistência do Modelo da Tarefa

Esta etapa se fundamenta na descrição da tarefa e consiste na análise de consistência do modelo da tarefa, ou seja, na verificação da completitude da representação da tarefa (relações temporais, restrições entre sub-tarefas).

Entrada	Saída
Modelo da Tarefa	Modelo da Tarefa revisado com relação a critérios de consistência

3.2.4 Concepção do Modelo da Interação

De posse do Modelo da Tarefa e do Perfil do Usuário já é possível realizar o projeto no nível conceitual, identificando as funções necessárias, o sequenciamento destas funções, a

definição do fluxo da interação, os mecanismos de interação, os objetos com os quais o usuário deverá interagir e a alocação tarefa/função que decide quais partes da tarefa serão realizadas pelo sistema e quais serão realizadas pelo usuário. Estas decisões constituem o modelo da interação.

A construção do modelo da interação, que será descrita no Capítulo 4, fundamenta-se no perfil do usuário (nível de conhecimento da tarefa, capacidade de aprendizado, motivações, etc) para escolher os objetos de interação e a forma como eles serão utilizados (comportamento dos objetos). No escopo deste trabalho, são tratadas as interfaces baseadas em janelas, botões e menus, devido a grande popularidade destes objetos e à possibilidade de modelar outros elementos de forma similar. Por exemplo, a seleção de um elemento de uma lista pode ser modelada como a escolha de uma opção de menu; a escolha de um elemento para realizar a navegação entre janelas pode ser modelado como o pressionamento de um botão.

Entrada	Saída
Modelo da tarefa, perfil do usuário, descrição de funcionalidades do sistema	Modelo da Interação (descrição textual tabular do projeto da interface destacando objetos da interação, ações realizáveis sobre estes objetos e a alocação de objetos às janelas de interação.

3.2.5 Avaliação da Consistência e Completitude do Modelo da Interação

Nesta etapa é realizada uma análise ergonômica do Modelo da Interação, verificando-se o número de opções atribuídas a cada menu e comparando-se com recomendações de projeto (diretrizes e padrões) para o contexto tratado. Também são verificados aspectos relacionados a consistência dos objetos, ou seja, tarefas semelhantes devem ser realizadas com a manipulação de objetos de interação semelhantes de forma a manter a consistência do projeto e reduzir a carga cognitiva sobre o usuário.

Entrada	Saída
Modelo da interação	Modelo da interação corrigido

3.2.6 Construção do Modelo da Navegação

A representação do Modelo da Navegação é feita utilizando-se redes de Petri colorida (CPN), na qual estão definidos os objetos de interação e o seu comportamento do ponto de vista de navegação entre janelas. A construção do Modelo da Navegação é detalhada no Capítulo 5.

Entrada	Saída
Modelo da interação	Modelo da navegação em CPN

3.2.7 Verificação do Modelo da Navegação

A verificação da correção no comportamento dos objetos de interação é realizada a partir da verificação formal do modelo da navegação. Este processo possibilita verificar se este modelo atende a um conjunto de propriedades de usabilidade definidas no contexto deste trabalho, tais como: a existência de caminhos diretos ou indiretos entre pontos da interação, o acesso à saída do sistema a partir de qualquer ponto da interação, o acesso à ajuda, entre outras. No processo de verificação das propriedades da rede que representa o modelo da navegação, é utilizada a ferramenta *Desgin/CPN* . A verificação do modelo da navegação é detalhada no Capítulo 5.

Entrada	Saída
Modelo da navegação	Modelo da navegação verificado segundo um conjunto de propriedades que refletem as características desejáveis para a interface, do ponto de vista ergonômico

3.2.8 Construção do protótipo

A construção do protótipo se fundamenta no modelo da interação e no modelo da navegação. Na construção do protótipo da interface, é utilizada a ferramenta de prototipação *DevGuide* que apoia o desenvolvimento de interfaces para o ambiente *UNIX/OpenLook*. O arquivo de protótipo contém todas as definições e comportamento dos objetos, definidos no

modelo da interação. O detalhamento da construção de um protótipo é apresentado no Capítulo 6.

Entrada	Saída
Modelo da interação, modelo da navegação	Arquivo contendo as definições do protótipo da interface

3.2.9 Validação do Protótipo quanto a Aspectos Ergonômicos do Projeto Visual

Esta etapa é realizada sem a participação do usuário e permite ao projetista analisar aspectos estruturais de especificação da interface (a exemplo da estrutura de menus), e de aspectos relacionados ao projeto visual (a exemplo da consistência no uso de cores). Consiste na utilização da ferramenta FAIUnix, construída no contexto deste trabalho, para analisar o arquivo de protótipo e obter informações sobre o projeto visual, que são então comparadas com recomendações ergonômicas. Neste trabalho foi adotado um conjunto de métricas derivado das métricas propostas por Mahajan e Schneiderman em [MS 97].

Entrada	Saída
Arquivo do protótipo da interface	Arquivo de coleta contendo um resumo das características estruturais/visuais da interface relativas a janelas e objetos da interação (menus e botões)

Além da análise dos aspectos estruturais e visuais face a um conjunto de métricas, nesta etapa também pode ser verificada a consistência do protótipo em relação ao modelo da interação, como será ilustrado no Capítulo 7.

3.2.10 Preparação do Protótipo para Validação

Nesta etapa é gerado o código do protótipo que será utilizado durante a coleta de dados da interação com o usuário. A preparação do protótipo da interface consiste na modificação do arquivo original do protótipo de modo a introduzir apontadores para as rotinas de captura automática de eventos da interação. Do arquivo de protótipo modificado é gerado o

protótipo executável, instrumentado para coleta, que será utilizado durante as sessões de interação com o usuário.

Entrada	Saída
Arquivo do protótipo, delimitação do escopo da coleta de dados	Arquivo de protótipo modificado e protótipo executável para captura dos dados da interação

3.2.11 Validação do Protótipo junto ao Usuário Final.

Uma vez definidas as tarefas que serão realizadas pelos usuários para validação do protótipo, nesta etapa são realizadas as sessões de teste com os usuários. A validação de protótipos é realizada com base na captura de dados da interação através da coleta automática de dados. As rotinas de coleta gravam informações com base na técnica de *Software Logging* sobre a ocorrência de eventos do teclado e do mouse, registrando o instante de tempo de ocorrência e o objeto da interação que deu origem ao evento.

Entrada	Saída
Protótipo executável para captura dos dados da interação	Arquivo de *log* da captura dos dados, contendo o registro de todos os eventos monitorados de acordo com a ordem cronológica de ocorrência

A partir destes dados, o projetista é capaz de analisar quantas e quais foram as tarefas concluídas, a ordem de execução das tarefas, a seqüência utilizada na execução da tarefa, quantos erros foram cometidos, em quais situações e quantas vezes foi necessária a ajuda, como o usuário utilizou o tempo durante a realização da tarefa, o padrão de utilização de recursos.

Nesta fase, o método recomenda a consulta ao usuário para obter a sua opinião sobre o produto no que diz respeito a aspectos como facilidade de uso, facilidade de aprendizado, etc.

3.3 CONCLUSÕES

Uma vez que a aplicação do método tem por objetivo o refinamento da especificação inicial de uma interface, propõe-se que seja aplicado de forma iterativa. O processo iterativo consiste na realimentação de informações entre as etapas de concepção e avaliação até que objetivos de usabilidade sejam atingidos, conforme é descrito no Capítulo 7.

Dentre os aspectos que se pretende investigar com este método de concepção, está a avaliação de características que podem ser consideradas adequadas do ponto de vista das propriedades dos modelos mas que podem ser consideradas inadequadas do ponto de vista do usuário quando da validação do protótipo. Um exemplo disto é a investigação da freqüência de utilização de caminhos projetados, algums das quais podem jamais vir a ser utilizados pelo usuário, seja por desconhecimento ou pela dificuldade, ou ainda por serem desnecessários no contexto do problema (do ponto de vista do usuário).

Os Capítulos 4, 5 e 6 detalham a aplicação do método ao estudo de caso enquanto o Capítulo 7 ilustra o seu caráter iterativo.

4 ANÁLISE E PROJETO DA INTERAÇÃO

Na análise da interação entre um usuário e a interface de um sistema é imprescindível a compreensão das características tanto do usuário quanto da tarefa a ser realizada. Este capítulo trata a análise destes dois elementos como as primeiras etapas do método de concepção de interfaces.

O modelo da tarefa, além de prover um conhecimento necessário sobre a tarefa, antes do início do projeto serve como referência para troca de informações entre especialistas envolvidos no projeto.

Para o perfil do usuário, foi adotado o Modelo Sintático-Semântico do Comportamento do Usuário, proposto por Shneiderman & Mayer [Shn 98]. A escolha se fundamentou no fato deste modelo atender às necessidades de descrição do usuário para os propósitos do trabalho. Ou seja, necessitávamos apenas um perfil que pudesse apoiar as decisões do projeto da interação.

4.1 LEVANTAMENTO DO PERFIL DO USUÁRIO

A partir do conhecimento do usuário é construída uma lista de atributos que representam o seu perfil, tais como: potencial, limitações e habilidades para realização da tarefa, preferências por dispositivos, motivações para uso do sistema. Estas e outras características devem ser apoiadas por informações obtidas através de questionários, observação do uso de produtos similares, entrevistas, etc. Neste trabalho estas informações foram obtidas a partir da observação informal do usuário interagindo com produtos da

categoria. As características foram escolhidas com base na relevância para o projeto em particular. Foi então traçado o perfil descrito a seguir:

Perfil do Usuário de um Navegador

Freqüência de execução da Tarefa – elevada
Motivação para realização da tarefa – facilitar o trabalho, oportunidades de lazer

CONHECIMENTO SEMÂNTICO:
Conhecimento de navegação na web – bom
Conhecimento de um Software de Navegação – bom
Conhecimento sobre o uso de Computadores – bom

CONHECIMENTO SINTÁTICO:
Uso de teclado, uso do mouse - bom
Conhecimento de terminologia específica da Web - regular

ESTILO COGNITIVO:
Aprendizado: por tentativa e erro.
Estilo de resolução de problemas: por iniciativa própria.
Retenção do aprendizado: elevada.
Níveis de curiosidade, persistência e inovação: elevados

De acordo com o perfil do usuário, são tomadas decisões sobre a forma de interação a ser adotada, sobre os dispositivos de interação que melhor se adequam a esse perfil, e sobre os mecanismos de ajuda projetados. Por exemplo, para um usuário inexperiente com software de navegação pode-se projetar o mecanismo de interação por seleção em menu, enquanto que para um usuário mais experiente, pode ser mais adequado projetar a interação por linhas de comando. No caso do *Browser-B*, as decisões de projeto tomadas com base no perfil do usuário serão apresentadas mais adiante no trabalho por ocasião da construção do modelo da interação.

4.2 Construção do Modelo da Tarefa

No processo de análise da tarefa é feita uma decomposição da tarefa de forma hierárquica geralmente resultando em uma representação arbórea até o nível de tarefas consideradas elementares (ou primitivas), no nível das folhas. Cada tarefa elementar deve corresponder a uma função elementar do sistema, e cada tarefa não elementar deve ser

simplificada para que possa ser realizada através das funções elementares. Neste nível são identificadas ações sobre o conjunto de objetos da interface, a exemplo do pressionamento de um botão ou seleção de uma opção de *menu*.

Um problema crucial consiste em combinar as características das tarefas com o projeto da interface. A estruturação da tarefa em subtarefas determina, por exemplo, quantas janelas serão necessárias para realizá-la. Associada ao perfil do usuário, esta informação influencia a escolha do estilo de interação (ex. menu ou comando) e a seqüência de ações a ser realizada pelo usuário.

Neste trabalho, para a modelagem de tarefas foi adotado MAD descrito no capítulo 2. Este método se mostrou adequado para identificar e selecionar os elementos de tarefa que são relevantes para o projeto, e conduzir à escolha adequada de elementos da interface (apresentação física dos objetos da interface, mecanismos de interação).

4.2.1 Descrição MAD do *Browser-B*

O exemplo a ser utilizado para a modelagem de interfaces, é um navegador de acesso a páginas da *web*. Uma análise detalhada do domínio foi realizada para obter uma estrutura de tarefas. Desta análise foi concluído que o navegador proposto deveria permitir ao usuário realizar as seguintes tarefas: consultar páginas (navegando entre as páginas visitadas), atualizar favoritos, solicitar ajuda e sair do sistema.

A seguir nas Figuras 4-1 e 4-2, é apresentada a árvore hierárquica para a tarefa "Navegar no *Browser*" segundo o método de descrição MAD. Os objetos e ações são tratados no nível conceitual, deixando o nível de diálogo e detalhes de apresentação para a fase do projeto da interface que se inicia com a construção do modelo da interação.

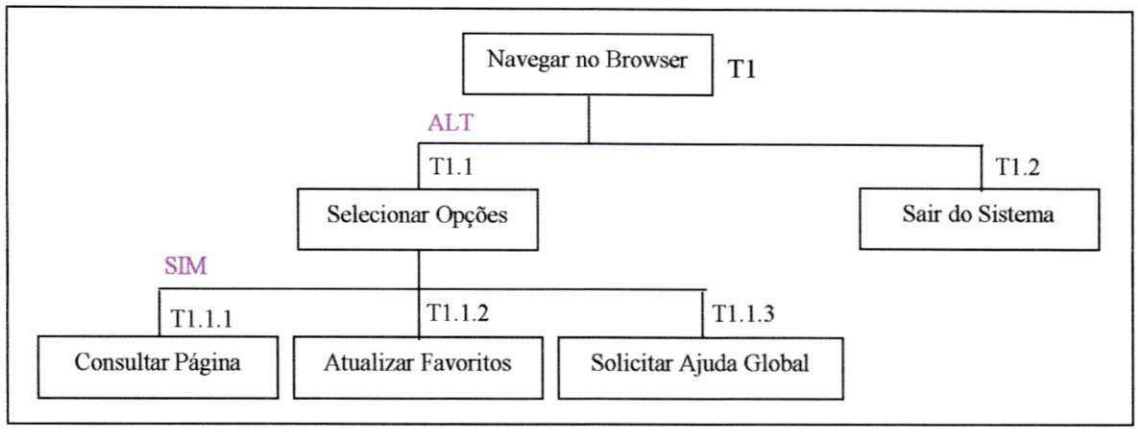

Figura 4-1: Tarefa Navegar no *Browser*

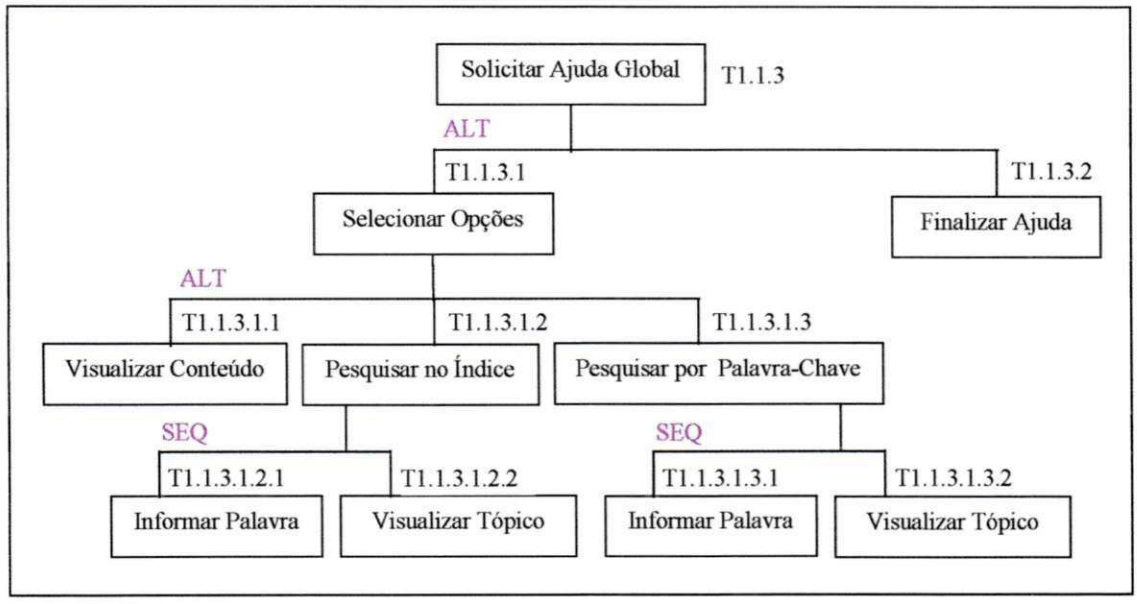

Figura 4-2: Tarefa Solicitar Ajuda Global

Descritores da Tarefa

A seguir nas Tabelas 4-1 à 4-6, são apresentados alguns dos descritores para a tarefa navegar no *Browser* segundo MAD.

Tabela 4-1: Descritor da Tarefa Navegar no *Browser*

TAREFA: T1 - Navegar no *Browser*	
ESTADO INICIAL: Navegador (*Browser*), Usuário	**ESTADO FINAL:** Navegador (*Browser*), Usuário, Informação
OBJETIVO: Realizar Navegação no Navegador (*Browser*)	
PRÉ-CONDIÇÕES: Navegador ativo AND Usuário disponível	**PÓS-CONDIÇÕES:** Navegador ativo AND Usuário disponível, Informação disponível
NÍVEL SUPERIOR:	**TAREFA COMPOSTA (estrutura):** Selecionar opções **ALT** Sair do sistema
TAREFA ELEMENTAR:	

Tabela 4-2: Descritor da Tarefa - Selecionar Opções (do Navegador)

TAREFA: T1.1 - Selecionar Opções (do Navegador)	
ESTADO INICIAL: Opções do Navegador: Selecionar Opções, Sair do sistema	**ESTADO FINAL:** Opções do Navegador: Consultar páginas, Atualizar favoritos, Solicitar ajuda global
OBJETIVO: Selecionar e ativar opções de interação com o Navegador	
PRÉ-CONDIÇÕES: Opções ativas: Selecionar Opções do Navegador AND Sair do Sistema	**PÓS-CONDIÇÕES:** Opções ativas: Consultar páginas AND Atualizar favoritos AND Solicitar ajuda global
NÍVEL SUPERIOR: T.1 Navegar no *Browser*	**TAREFA COMPOSTA (estrutura):** Consultar Página **SIM** Atualizar Favoritos SIM Solicitar Ajuda Global
TAREFA ELEMENTAR:	

Tabela 4-3: Descritor da Tarefa - Solicitar Ajuda global

TAREFA: T1.1.3 - Solicitar Ajuda Global	
ESTADO INICIAL: Opções do Navegador: Consultar páginas, Atualizar favoritos, Solicitar ajuda global.	**ESTADO FINAL:** Opções do Navegador: Selecionar Opções, Finalizar Ajuda.
OBJETIVO: Ter acesso à ajuda global sobre o uso do Navegador	
PRÉ-CONDIÇÕES: Opções ativas: Consultar páginas AND Atualizar favoritos AND Solicitar ajuda global	**PÓS-CONDIÇÕES:** Opções ativas: Selecionar Opções AND Finalizar Ajuda
NÍVEL SUPERIOR: T1.1 - Selecionar Opções (do Navegador)	**TAREFA COMPOSTA (estrutura):** Selecionar Opções **ALT** Finalizar Ajuda
TAREFA ELEMENTAR:	

Tabela 4-4: Descritor da Tarefa – Selecionar opções (de Ajuda)

TAREFA: T1.1.3.1 - Selecionar opções (de Ajuda)	
ESTADO INICIAL: Opções do Navegador: Selecionar opções, Finalizar Ajuda	**ESTADO FINAL:** Opções do Navegador: Visualizar conteúdo, Pesquisar no índice, Pesquisar por Palavra chave
OBJETIVO: Selecionar uma das opções de ajuda global	
PRÉ-CONDIÇÕES: Opções ativas: Selecionar opções AND Finalizar Ajuda	**PÓS-CONDIÇÕES:** Opções ativas: Visualizar conteúdo AND Pesquisar no índice AND Pesquisar por Palavra chave
NÍVEL SUPERIOR: T1.1.3 - Solicitar Ajuda Global	**TAREFA COMPOSTA (estrutura):** Visualizar Conteúdo **ALT** Pesquisar no índice **ALT** Pesquisar por palavra-chave
TAREFA ELEMENTAR:	

Tabela 4-5: Descritor da Tarefa - Pesquisar no índice

TAREFA: T1.1.3.1.2 - Pesquisar no índice	
ESTADO INICIAL: Opções do Navegador: Visualizar conteúdo, Pesquisar no índice, pesquisar por palavra-chave.	**ESTADO FINAL:** Opções do Navegador: Informar palavra, Visualizar Tópico
OBJETIVO: Obter ajuda global sobre o uso do Navegador a partir de pesquisa no índice	
PRÉ-CONDIÇÕES: Opções ativas: Visualizar conteúdo AND Pesquisar no índice AND pesquisar por palavra-chave	**PÓS-CONDIÇÕES:** Opções ativas: Informar palavra AND Visualizar Tópico
NÍVEL SUPERIOR: T1.1.3.1 - Selecionar opções	**TAREFA COMPOSTA (estrutura):** Informar palavra **SEQ** Visualizar tópico
TAREFA ELEMENTAR:	

Tabela 4-6: Descritor da Tarefa – Visualizar Tópico

TAREFA: T1.1.3.1.2.2 - Visualizar Tópico	
ESTADO INICIAL: Opções do Navegador: Informar palavra, Visualizar Tópico	**ESTADO FINAL:** Opções do Navegador: Informar palavra, Visualizar Tópico
OBJETIVO: Visualizar um tópico na Ajuda Global	
PRÉ-CONDIÇÕES: Opções ativas: Informar palavra AND Visualizar Tópico.	**PÓS-CONDIÇÕES:** Opções ativas: Informar palavra AND Visualizar Tópico
NÍVEL SUPERIOR: T1.1.3.1.2 – Pesquisar no Índice	**TAREFA COMPOSTA (estrutura):**
TAREFA ELEMENTAR: Movimentar o cursor sobre o texto da Ajuda Global	

4.3 VALIDAÇÃO DO MODELO DA TAREFA

A avaliação da usabilidade durante os estágios iniciais do projeto é essencial para que este se desenvolva adequadamente. Problemas de usabilidade no projeto conceitual (no domínio dos objetos e tarefas) estão entre os mais difíceis de corrigir a medida que o projeto se desenvolve. Problemas neste nível fatalmente se propagam até o projeto da apresentação e do diálogo.

A análise do modelo da tarefa se fundamenta na sua descrição e consiste na verificação da completitude da representação das tarefas (relações temporais, restrições).

No Apêndice A é apresentada a árvore hierárquica do *Browser-B* em MAD. A partir da análise do modelo original, observou-se que na tarefa "Atualizar Favoritos" não foi considerada a possibilidade de "Desfazer uma Exclusão" ou "Mudança de Nome" (Renomear) em "Organizar Favoritos". Com a inclusão destas opções a Figura 4-3 é modificada resultando na Figura 4-4.

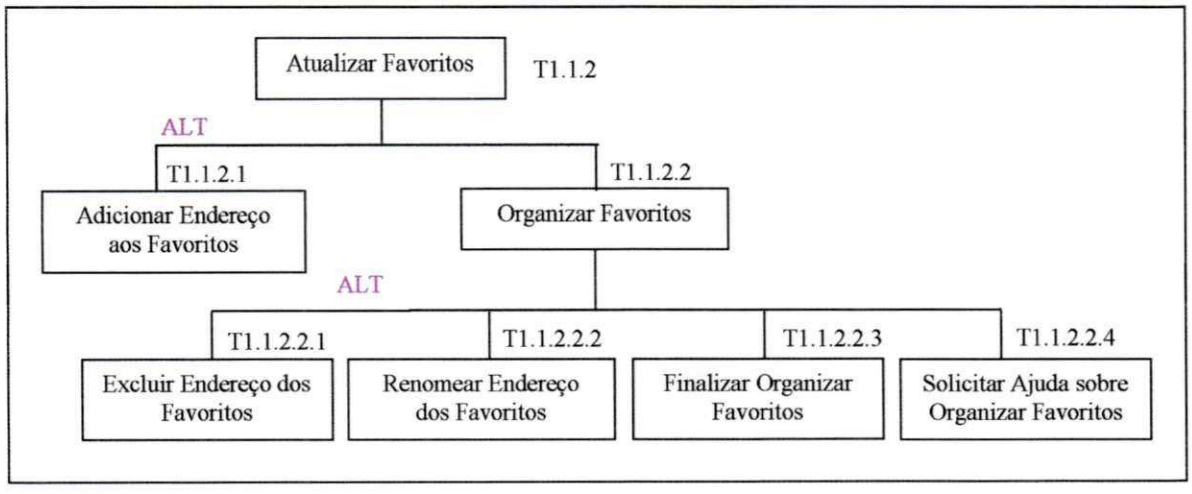

Figura 4-3: Tarefa Atualizar Favoritos (Original)

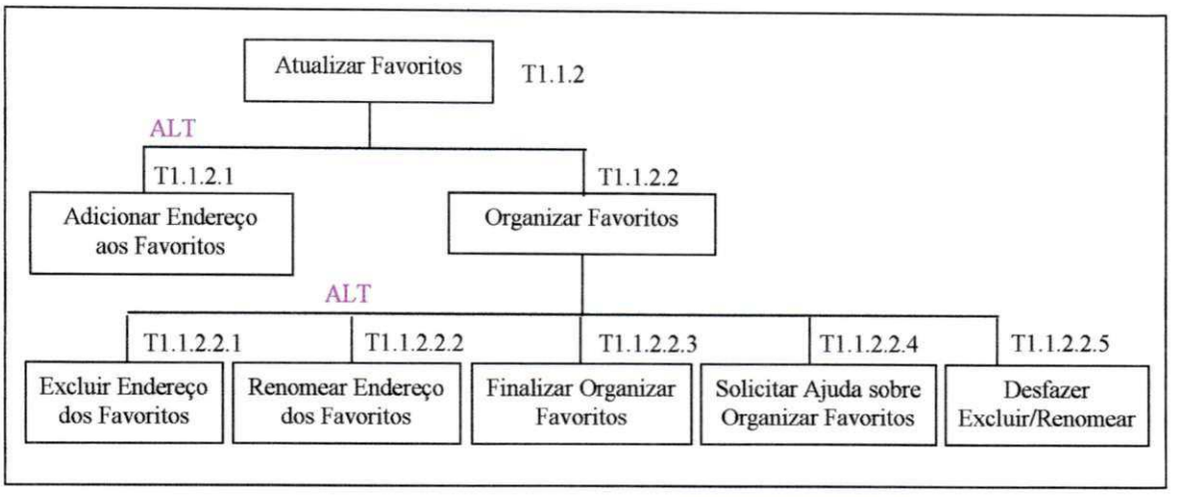

Figura 4-4: Tarefa Atualizar Favoritos (Modificada)

Resolver problemas de usabilidade ainda nesta fase (projeto conceitual) é mais simples do que durante o desenvolvimento do protótipo operacional. O exemplo apresentado ilustra como o modelo da tarefa possibilita a avaliação nos estágios iniciais do projeto.

4.4 CONSTRUÇÃO DO MODELO DA INTERAÇÃO

A construção do modelo da interação se fundamenta na análise das características do usuário, da tarefa e do sistema. Observa-se que embora existam muitos métodos para a modelagem de sistemas, dificilmente os modelos são utilizados em fases posteriores do projeto. Palanque em [PBSBD 93] sugere a necessidade de um campo comum que possibilite relacionar o modelo da tarefa com o modelo da interação, sendo esta a abordagem adotada neste trabalho.

4.4.1 Modelo da Interação do *Browser- B*

O principal objetivo da análise da tarefa foi a construção de um modelo de tarefa que fosse útil para a construção do modelo da interação. A partir da descrição da tarefa utilizando-se o método MAD, foi feita a associação entre os objetos da tarefa e os objetos de interação propostos no projeto da interface. Esta seção apresenta a construção do modelo da interação do *Browser-B*, conforme as etapas descritas a seguir.

Modelo da Interação – Etapa I

Nesta fase é construída uma tabela na qual estão representados: identificação da tarefa, as ações e os objetos envolvidos na tarefa aos quais são associados à técnica proposta para realizar a interação (seleção por menu, manipulação de texto, ativação por botão), o objeto de interface proposto para disparar a realização da tarefa (opção de menu, botão, campo de texto) e uma janela onde o objeto da interface será representado. A seguir está representado o resultado desta primeira etapa para construção do modelo da interação do *Browser-B*. Nesta etapa são consideradas apenas as tarefas que demandam uma ação do usuário e por conseguinte uma representação na interface (a exemplo das tarefas elementares). Nas Tabelas 4-7 e 4-8 é apresentado o Modelo da Interação - Etapa I.

As técnicas de interação consideradas, foram aquelas normalmente encontradas em qualquer padrão de interfaces: manipulação de texto, entrada/saída de mensagens e seleção por menu/ativação por botão para seleção de opções.

Tabela 4-7: Modelo da Interação - Etapa I

Id. Tarefa	Tarefa		Técnica de Interação	Objeto da Interface	Janela
	Ação	Objeto da Tarefa			
T1.1.1.1	Abrir	Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.3	Salvar	Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.4	Imprimir	Página	Sel. por Menu	Opção de Menu	Principal
T1.2	Sair	Do Sistema	Sel. por Menu	Opção de Menu	Principal
T1.1.1.1.1.1.1.2.1	Recortar	Texto	Sel. por Menu	Opção de Menu	Principal
T1.1.1.1.1.1.1.2.2	Copiar	Texto	Sel. por Menu	Opção de Menu	Principal
T1.1.1.1.1.1.1.2.3	Colar	Texto	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.1.2	Parar	Carregamento da Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.1.1	Atualizar	Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.1.3	Ver	Cód Fonte da Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.2.1	Voltar	Para a Pág Anterior	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.2.2	Avançar	Para a Pág Seguinte	Sel. por Menu	Opção de Menu	Principal
T1.1.2.1	Adicionar	Endereço aos Favoritos	Sel. por Menu	Opção de Menu	Principal
T1.1.2.2	Organizar	Favoritos	Sel. por Menu	Opção de Menu	Principal
T1.1.3.1.1	Visualizar	Conteúdo	Sel. por Menu	Menu	Principal
T1.1.3.1.2	Pesquisar	Índice	Sel. por Menu	Opção de Menu	Principal
T1.1.3.1.3	Pesquisar	Palavras-Chave	Sel. por Menu	Opção de Menu	Principal
T1.1.3	Solicitar Ajuda	Global	Sel. por Menu	Opção de Menu	Principal
T1.1.1.1.2	Solicitar Ajuda	Sobre Abrir Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.3.1.2	Solicitar Ajuda	Sobre Salvar Página	Sel. por Menu	Opção de Menu	Principal
T1.1.1.2.4.2	Solicitar Ajuda	Sobre Imprimir Página	Sel. por Menu	Opção de Menu	Principal
T1.1.2.1.2	Solicitar Ajuda	Sobre Adicionar Endereço aos Favoritos	Sel. por Menu	Opção de Menu	Principal
T1.1.2.2.1.2	Solicitar Ajuda	Sobre Excluir Endereço dos Favoritos	Sel. por Menu	Opção de Menu	Principal
T1.1.2.2.4	Solicitar Ajuda	Sobre Organizar Favoritos	Sel. por Menu	Opção de Menu	Principal
T1.1.1.1.1.1.1.1	Editar	Endereço	Manipulação Texto	Campo de Texto	Principal

Tabela 4-8: Modelo da Interação - Etapa I (Continuação)

Id. Tarefa	Tarefa		Técnica de Interação	Objeto da Interface	Janela
	Ação	Objeto da Tarefa			
T1.1.2.2.1.1.1.1	Identificar	Nome do Endereço	Manipulação Texto	Campo de Texto	Excluir Favoritos
T1.1.2.2.1.1.1.2.1	Confirmar Identificação	Do Nome	Ativação por Botão	Botão	Excluir Favoritos
T1.1.2.2.1.1.1.2.2	Cancelar Identificação	Do Nome	Ativação por Botão	Botão	Excluir Favoritos
T1.1.2.2.1.2	Solicitar Ajuda	Sobre Excluir Endereço dos Favoritos	Ativação por Botão	Botão	Excluir Favoritos
T1.1.2.2.1.1.3	Ir Para	Página Inicial	Ativação por Botão	Botão	Excluir Favoritos
T1.1.2.1.1.1.1.1	Identificar	Nome do Endereço	Manipulação Texto	Campo de Texto	Adicionar Favoritos
T1.1.2.1.1.1.2.1.1	Confirmar Identificação	Do Nome	Ativação por Botão	Botão	Adicionar Favoritos
T1.1.2.1.1.1.2.1.2	Cancelar Identificação	Do Nome	Ativação por Botão	Botão	Adicionar Favoritos
T1.1.2.1.2	Solicitar Ajuda	Sobre Adicionar Endereço aos Favoritos	Ativação por Botão	Botão	Adicionar Favoritos
T1.1.2.2.2.1.1.1	Identificar	Nome Original	Manipulação Texto	Campo de Texto	Renomear Favoritos
T1.1.2.2.2.1.1.2	Identificar	Nome Desejado	Manipulação Texto	Campo de Texto	Renomear Favoritos
T1.1.2.2.2.1.2.1	Confirmar Identificação	Dos Nomes	Ativação por Botão	Botão	Renomear Favoritos
T1.1.2.2.2.1.2.2	Cancelar Identificação	Dos Nomes	Ativação por Botão	Botão	Renomear Favoritos
T1.1.1.2.3.1.1.1.1.1	Identificar	Diretório	Manipulação Texto	Campo de Texto	Salvar Página
T1.1.1.2.3.1.1.1.1.2	Identificar	Nome	Manipulação Texto	Campo de Texto	Salvar Página
T1.1.1.2.3.1.1.1.2.1	Confirmar Identificação	Do Caminho	Ativação por Botão	Botão	Salvar Página
T1.1.1.2.3.1.1.1.2.2	Cancelar Identificação	Do Caminho	Ativação por Botão	Botão	Salvar Página
T1.1.1.2.3.1.2	Solicitar Ajuda	Sobre Salvar Página	Ativação por Botão	Botão	Salvar Página
T1.1.1.2.3.2	Imprimir	Página	Ativação por Botão	Botão	Salvar Página
T1.1.1.2.4.1.1.1.1	Identificar	Impressora	Manipulação Texto	Campo de Texto	Imprimir Página
T1.1.1.2.4.1.1.1.2.1	Especificar	Página Inicial	Manipulação Texto	Campo de Texto	Imprimir Página
T1.1.1.2.4.1.1.1.2.2	Especificar	Página Final	Manipulação Texto	Campo de Texto	Imprimir Página
T1.1.1.2.4.1.1.1.3	Definir	Num de Cópias	Manipulação Texto	Campo de Texto	Imprimir Página
T1.1.1.2.4.1.1.2.1	Confirmar	Impressão	Ativação por Botão	Botão	Imprimir Página
T1.1.1.2.4.1.1.2.2	Cancelar	Impressão	Ativação por Botão	Botão	Imprimir Página
T1.1.1.2.4.2	Solicitar Ajuda	Sobre Imprimir Página	Ativação por Botão	Botão	Imprimir Página
T.1.1.3.1.2.1	Informar	Palavra	Manipulação Texto	Campo de Texto	Ajuda
T.1.1.3.1.3.1	Informar	Palavra	Manipulação Texto	Campo de Texto	Ajuda
T1.1.3.2	Finalizar	Ajuda	Ativação por Botão	Botão	Ajuda
T1.1.3.1.1	Visualizar	Conteúdo	Ativação por Botão	Botão	Ajuda
T1.1.3.1.2	Pesquisar	No Índice	Ativação por Botão	Botão	Ajuda
T1.1.3.1.3	Pesquisar	Por Palavra-Chave	Ativação por Botão	Botão	Ajuda
T1.1.3.1.2.2	Visualizar	Tópico	Ativação por Botão	Botão	Ajuda
T1.1.3.1.3.2	Visualizar	Tópico	Ativação por Botão	Botão	Ajuda
T1.1.1.1.1.1.1.1	Editar	Endereço	Manipulação Texto	Campo de Texto	Abrir Página
T1.1.1.1.1.1.2.1	Confirmar Identificação	Do Endereço	Ativação por Botão	Botão	Abrir Página
T1.1.1.1.1.1.2.2	Cancelar Identificação	Do Endereço	Ativação por Botão	Botão	Abrir Página
T1.1.1.1.2	Solicitar Ajuda	Sobre Abrir Página	Ativação por Botão	Botão	Abrir Página
T1.1.2.2.1	Excluir	Endereço dos Favoritos	Ativação por Botão	Botão	Organizar Favoritos
T1.1.2.2.2	Renomear	Endereço dos Favoritos	Ativação por Botão	Botão	Organizar Favoritos
T1.1.2.2.3	Finalizar	Organizar Favoritos	Ativação por Botão	Botão	Organizar Favoritos
T1.1.2.2.4	Solicitar Ajuda	Sobre Organizar Favoritos	Ativação por Botão	Botão	Organizar Favoritos
T1.1.2.2.5	Desfazer	Excluir / Renomear Favoritos	Ativação por Botão	Botão	Organizar Favoritos

Modelo da Interação – Etapa II

Durante esta etapa, são descritas as janelas (Tabela 4-9) e agrupados os objetos de uma mesma classe (Tabelas 4-10 à 4-16), a exemplo de botões e menus.

Tabela 4-9: Modelo da Interação - Descrição das Janelas

Descrição de Janelas		
Id da Janela	*Descrição*	*Objetos*
winBW	Janela Principal do Navegador	Botão, Menus, Campo de Texto, Painel de Texto
winAJ	Janela de Ajuda	Botão, Campo de Texto, Painel de Texto
winAP	Janela Abrir Página	Botão, Campo de Texto, Mensagem
winSP	Janela Salvar Página	Botão, Campo de Texto
winIP	Janela Imprimir Página	Botão, Campo de Texto
winOF	Janela Organizar Favoritos	Botão, Lista de Elementos
winRF	Janela Renomear Favoritos	Botão, Campo de Texto
winEF	Janela Excluir Favoritos	Botão, Campo de Texto, Mensagem
winAF	Janela Adicionar Favoritos	Botão, Campo de Texto, Mensagem

O agrupamento de elementos no modelo da interação é feito de modo a permitir uma análise ergonômica da solução proposta. Em função do agrupamento, é possível analisar aspectos diferentes. Por exemplo: agrupando por elementos de interação é possível analisar aspectos como o número de opções propostas para cada menu e comparar com as diretrizes encontradas na literatura ou mesmo guias de estilo específicos para o projeto em andamento. Ou ainda, agrupando por tarefa, é possível analisar a consistência dos métodos propostos para sua realização (tarefas semelhantes realizadas por objetos semelhantes).

O agrupamento por classe de objetos é mostrado nas Tabelas 4-10 à 4-16. Nestas tabelas, estão representadas a identificação e descrição dos objetos de interação, além do seu comportamento. O comportamento pode resultar na navegação entre janelas e/ou na atualização das opções de interação disponíveis para o usuário. Quando resultar em uma mudança de janela, a representação corresponde à Navegação (*Navigation*). Se ocorrer uma situação de retorno, a representação corresponde a Desfazer o contexto ou Fechar a janela ativa (*undo & close*). Se a saída do sistema for desejada, a representação corresponde ao "*exit*".

As situações que resultam apenas na atualização das opções de interação, correspondem a Não_Navegação (*No Navigation*), podendo ocorrer de duas formas:

- *No_Navigation* = Não Navegação com alteração de opções disponíveis
- *No_Navigation* * = Não Navegação mantendo-se as opções disponíveis

Tabela 4-10: Modelo da Interação - Descrição dos Menus

Descrição de Menus					
Id da Janela	*Id do Menu*	*Opções*	*Id da Opção*	*Ativa*	*Comportamento*
WinBW	mArq	Abrir Pagina	o_abrpag	winAP	*Navigation*
		Salvar Pagina	o_savpag	winSP	*Navigation*
		Imprimir Pagina	o_imppag	winIP	*Navigation*
		Sair	o_sair	-	*Exit*
	mEdt	Recortar	o_recort	-	*No_Navigation* *
		Copiar	o_copiar	-	*No_Navigation* *
		Colar	o_colar	-	*No_Navigation* *
	mExb	Parar	o_parar	-	*No_Navigation* *
		Atualizar	o_atualz	-	*No_Navigation* *
		Ver Código Fonte	o_codfnt	-	*No_Navigation* *
	mIrP	Voltar	o_voltar	-	*No_Navigation* *
		Avançar	o_avanc	-	*No_Navigation* *
	mFav	Adicionar	o_adcfav	winAF	*Navigation*
		Organizar	o_orgfav	winOF	*Navigation*
	mAjd	Conteúdo	o_hcont	smCnt	*No_Navigation*
		Índice	o_hindex	winAJ	*Navigation*
		Pesquisar	o_pesq	winAJ	*Navigation*
	smCnt	Geral	o_hgeral	winAJ	*Navigation*
		Abrir Pagina	o_habrpag	winAJ	*Navigation*
		Salvar Pagina	o_hsavpag	winAJ	*Navigation*
		Imprimir Pagina	o_himppag	winAJ	*Navigation*
		Adicionar a Favoritos	o_hadcfav	winAJ	*Navigation*
		Organizar Favoritos	o_horgfav	winAJ	*Navigation*
		Excluir Favoritos	o_hexcfav	winAJ	*Navigation*

Tabela 4-11: Modelo da Interação - Descrição dos Botões

Descrição dos Botões				
Id da Janela	*Id do Botão*	*Descrição*	*Ativa*	*Comportamento*
WinBW	BArqBW	Arquivo	mArq	*No_Navigation*
	BEdtBW	Editar	mEdt	*No_Navigation*
	bExbBW	Exibir	mExb	*No_Navigation*
	bIrPBW	Ir Para	mIrP	*No_Navigation*
	bFavBW	Favoritos	mFav	*No_Navigation*
	bAjdBW	Ajuda	mAjd	*No_Navigation*
	aVltBW	Voltar	-	*No_Navigation* *
	aAvnBW	Avançar	-	*No_Navigation* *
	aStpBW	Parar	-	*No_Navigation* *
	aRefBW	Atualizar	-	*No_Navigation* *

Tabela 4-12: Modelo da Interação - Descrição dos Botões (Continuação)

Descrição dos Botões				
Id da Janela	*Id do Botão*	*Descrição*	*Ativa*	*Comportamento*
winAJ	cntAJ	Conteúdo	-	*No_Navigation* *
	indAJ	Índice	-	*No_Navigation* *
	psqAJ	Pesquisar	-	*No_Navigation* *
	exbAJ	Exibir	-	*No_Navigation* *
	fchAJ	Fechar	-	*Undo & Close*
winAP	cnfAP	Confirmar	-	*Undo & Close*
	canAP	Cancelar	-	*Undo & Close*
	hlpAP	Ajuda	winAJ	*Navigation*
winSP	prnSP	Imprimir	winIP	*Navigation*
	cnfSP	Confirmar	-	*Undo & Close*
	canSP	Cancelar	-	*Undo & Close*
	hlpSP	Ajuda	winAJ	*Navigation*
winIP	cnfIP	Confirmar	-	*Undo & Close*
	canIP	Cancelar	-	*Undo & Close*
	hlpIP	Ajuda	winAJ	*Navigation*
winOF	desOF	Desfazer	desOF	*No_Navigation* *
	renOF	Renomear	winRF	*Navigation*
	delOF	Excluir	winEF	*Navigation*
	hlpOF	Ajuda	winAJ	*Navigation*
winRF	cnfRF	Confirmar	-	*Undo & Close*
	canRF	Cancelar	-	*Undo & Close*
winEF	gtmEF	Pág Inicial	winBW	*Navigation*
	cnfEF	Confirmar	-	*Undo & Close*
	canEF	Cancelar	-	*Undo & Close*
	hlpEF	Ajuda	winAJ	*Navigation*
winAF	cnfAF	Confirmar	-	*Undo & Close*
	canAF	Cancelar	-	*Undo & Close*
	hlpAF	Ajuda	winAJ	*Navigation*

Tabela 4-13: Modelo da Interação - Descrição dos Campos de Texto

Descrição de Campos de Texto				
Id da Janela	*Id do Campo*	*Descrição*	*Ativa*	*Comportamento*
winBW	endBW	Endereço da Página	-	-
winAJ	fndAJ	Procurar	-	-
winAP	endAP	Abrir	-	-
winSP	dirSP	Diretório	-	-
	arqSP	Nome do Arquivo	-	-
winIP	impIP	Impressora	-	-
	pbegIP	Paginas de	-	-
	pendIP	Ate	-	-
	ncpIP	Numero de Copias	-	-
winRF	norgRF	Nome Original	-	-
	ndesRF	Nome Desejado	-	-
winEF	nomEF	Nome	-	-
winAF	nomAF	Nome	-	-

Tabela 4-14: Modelo da Interação - Descrição dos Painéis de Texto

Descrição de Painéis de Texto				
Id da Janela	*Id do Painel*	*Descrição*	*Ativa*	*Comportamento*
winBW	pagBW	Apresentação da Página	-	-
winAJ	txtAJ	Apresentação da Ajuda	-	-

Tabela 4-15: Modelo da Interação - Descrição de Listas

Descrição de Listas				
Id da Janela	*Id da Lista*	*Descrição*	*Ativa*	*Comportamento*
winOF	lstOF	Lista dos Favoritos existentes	-	-

Tabela 4-16: Modelo da Interação - Descrição de Mensagens

Descrição de Mensagens				
Id da Janela	*Id Mensagem*	*Descrição*	*Ativa*	*Comportamento*
WinAP	msgAP	Digite o Endereço	-	-
WinEF	msgEF	Deseja Excluir?	-	-
winAF	msgAF	A Pagina será adicionada aos seus favoritos	-	-

4.5 VALIDAÇÃO DO MODELO DA INTERAÇÃO

No Modelo da Interação é possível avaliar o número de objetos em cada janela, a exemplo do número de botões existentes, ou analisar a hierarquia dos menus, verificando se existem muitos subníveis, o que pode indicar a necessidade de uma divisão melhor das subtarefas.

Pode-se ainda verificar as alternativas para a realização de uma mesma tarefa, ou seja, é possível identificar todas as formas que conduzem a uma mesma tarefa, avaliando o excesso ou ausência de caminhos alternativos.

No modelo estudado pode-se verificar por exemplo que na janela principal do Navegador (winBW), o número de botões é igual a 10, o que representa o limite superior recomendado para um número de opções simultâneas que devem ser apresentadas ao usuário [Shn 98].

O modelo da interação será utilizado na construção do modelo da navegação apresentado no Capítulo 5 e na construção do protótipo apresentado no Capítulo 6.

4.6 CONCLUSÕES

Neste capítulo foi apresentado como o método MAD foi utilizado para representar a associação entre uma tarefa e as metas do usuário e, como estas metas podem ser atingidas a partir de uma seqüência de ações, ou cenário da interação. Foi também ilustrado como, a partir do modelo de tarefa e do perfil do usuário, é possível construir um modelo de interação no qual são associados os dispositivos e mecanismos de interação às ações realizadas pelo usuário. O próximo Capítulo trata das etapas de análise do modelo da interação, destacando a avaliação dos mecanismos de navegação projetados.

5 MODELO E ANÁLISE DA NAVEGAÇÃO

A avaliação da qualidade das interfaces usuário-computador pode ser mais efetiva se baseada em um método de análise formal que represente suas características. Esta representação pode então ser utilizada para avaliar os sistemas modelados e levar a recomendações para eliminar problemas encontrados e sugerir novas alternativas de interação.

Este capítulo apresenta uma representação da navegação em interfaces de caráter genérico que foi proposta com o objetivo de sintetizar, independente dos dispositivos de interação, as situações de transição mais comuns em uma interface com o usuário. Em seguida é apresentada a etapa de construção do modelo da navegação em rede de Petri colorida . Este modelo foi construído a partir da representação genérica. O modelo da navegação é então instanciado com a interface do *Browser-B* e o capítulo conclui com uma discussão sobre a adequação do modelo aos propósitos deste trabalho.

5.1 REPRESENTAÇÃO GENÉRICA DA NAVEGAÇÃO

Esta seção apresenta uma representação genérica da navegação, que foi proposta com o objetivo de sintetizar, independente dos dispositivos de interação, as situações de transição mais comuns em uma interface com o usuário. A independência de dispositivos, neste contexto, significa que uma ação solicitada pelo usuário resulta na execução de um procedimento que independe do dispositivo ou mecanismo de interação utilizado. Assim, se um usuário solicita uma ação de "*undo*" (desfazer) através de um comando no teclado, um comando de voz, seleção de um item de menu ou através do acionamento de um botão, a ação resultante é a mesma, *undo*.

A partir do estudo das características das interfaces gráficas observa-se que a transição entre os estados da interface[7] durante uma interação pode ocorrer de forma direta ou indireta, condicional ou incondicional, mantendo o contexto atual da interação, causando uma mudança de contexto ou ainda restaurando o contexto anterior [STF 97]. À cada estado da interação está associado um contexto que compreende as informações do estado atual e um histórico da navegação contendo informações sobre os estados anteriores (de onde venho, onde estou, para onde posso ir).

Estrutura da Navegação

Como já foi mencionado no Capítulo 2, Hix e Hartson [HH 93], declaram que um modelo matemático que represente sistemas de interface deve modelar as seguintes situações: seqüência, iteração, escolha, ações independentes, capacidade de interrupção, concorrência e intervalos de espera, as quais passamos a descrever:

Seqüência: Duas tarefas estão relacionadas no tempo e em seqüência se uma das tarefas puder ser completamente concluída antes do início da outra.

Iteração: Existe uma iteração em uma tarefa se a mesma for realizada mais de uma vez.

Escolha: Existe uma escolha entre duas tarefas se o usuário puder realizar uma seleção (com igual probabilidade) para execução de apenas uma das tarefas. Se a escolha puder ser refeita, trata-se de uma escolha repetida.

Ações independentes: Existe uma relação de independência entre tarefas se o usuário puder executar as tarefas sem nenhum tipo de restrição quanto à ordem a ser seguida.

Capacidade de interrupção: Existe a capacidade de interrupção entre tarefas em situações nas quais as tarefas podem ser interrompidas por outras tarefas antes que tenham sido completadas.

Concorrência: Existe uma concorrência entre tarefas se duas ou mais tarefas, puderem ser realizadas simultaneamente. Concorrência é uma relação temporal que deve ser considerada na análise das interfaces. Por exemplo, no navegador um usuário pode realizar interação com múltiplas tarefas, pode imprimir uma página enquanto está salvando outra.

Intervalos de espera: Existe um atraso, ou intervalo de espera, entre tarefas se uma tarefa puder ser completada, um certo intervalo de tempo, antes da outra tarefa ser iniciada.

---

[7] Estados da interface podem também ser referenciados no texto como estados da interação.

Para os propósitos deste trabalho, considera-se que as situações descritas a seguir representam de forma genérica a navegação na interface que está associada à transição entre estados da interface durante uma interação. Na descrição das situações, os estados da interação são também referenciados como pontos da interação e são representados como nós no diagrama da navegação descrito adiante. A transição entre dois estados da interação pode ocorrer de forma direta ou indireta como será discutido a seguir.

Retorno na interação - Esta situação corresponde à navegação do ponto atual para um ponto anterior na interação. Ela pode provocar o resgate do contexto anterior (*undo*) ou manter o efeito das operações realizadas (*close*). A recuperação do contexto pode ocorrer de duas formas: *Undo Simples* que restaura o contexto do ponto antecessor e devolve-lhe o controle da interação, e o *Undo Sucessivo*, que recupera o contexto de um ponto antecessor na navegação a partir de uma seqüência de *undos simples*. A recuperação do contexto pretendido se dá em um procedimento passo a passo, no qual os contextos intermediários vão sendo recuperados. De forma semelhante, a operação de fechar (*close*) provoca o retorno na navegação porém mantém os resultados das operações já realizadas. Tanto as operações de *undo sucessivo* quanto as operações de *close sucessivo* têm como limite o estado (ponto) inicial da interação.
Avanço na interação - Esta situação decorre da navegação para um ponto subseqüente na interação. A transição pode ocorrer de forma incondicional ou mediante o atendimento a restrições:

- Deslocamento incondicional – consiste na transição entre pontos da interação independentemente do estado atual da interação.
- Deslocamento condicional – consiste na navegação de um ponto da interação para outro subseqüente ou para um ponto qualquer, dependendo da condição ser ou não satisfeita.

Retorno ao início da interação – Consiste na navegação de um ponto qualquer da interação para o ponto inicial da interação.
Saída do sistema – Consiste na navegação de um ponto qualquer para o estado final da interação.

Estas situações são representadas em um diagrama mostrado na Figura 4-1. A notação utilizada no diagrama é descrita em seguida.

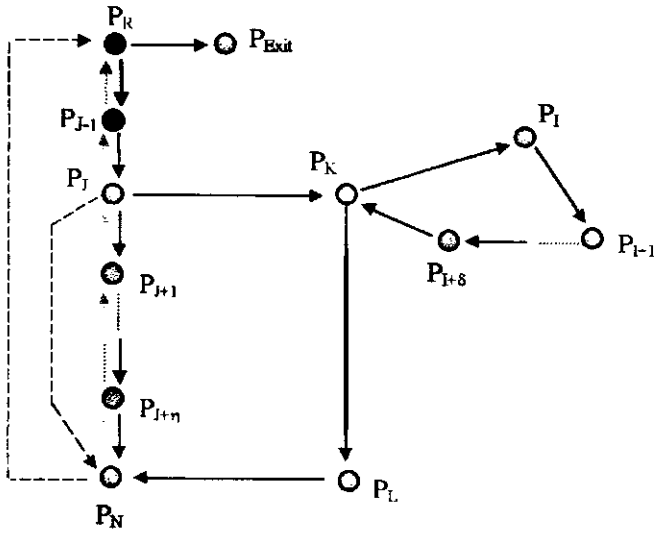

Figura 5-1: Representação Genérica da Navegação

Notação adotada no diagrama:

- ● representa os pontos da interação (nós do diagrama)
- ● representa o ponto inicial da interação (nó raiz)
- ——► representa os caminhos entre pontos da interação
- ----------► representa os caminhos de retorno na interação
- ---------► representa os caminhos de acesso direto a um ponto da interação

Neste diagrama apresentado na Figura 5-1, um nó representa um ponto ou estado da interação. A partir de um nó P_J da interação, é possível:

- Deslocar-se para um nó subseqüente P_{J+1}.
- Retornar ao nó antecedente P_{J-1} através de um *undo* ou *close*.
- Retornar a um nó qualquer através de *undos* ou *close* sucessivos.
- Retornar diretamente ao nó raiz P_R.
- Deslocar-se diretamente para um nó P_K independentemente de sua localização.

Na figura estão representados os seguintes caminhos de navegação na interação:

⇒ Navegação incondicional entre nós:

$P_R \longrightarrow P_{J-1} \longrightarrow P_J \longrightarrow P_{J+1}$

Ex: Realização da próxima ação disponível na seqüência da interação.

⇒ Navegação condicional entre nós:

$P_R \longrightarrow P_{J-1} \longrightarrow P_J \longrightarrow P_K$

Ex: Seleção de uma dentre um conjunto de ações disponíveis.

⇒ Navegação de Retorno (*Undo*) ou Fechamento (*Close*) simples:

$P_R \longrightarrow P_{J-1} \longrightarrow P_J \cdots P_{J-1}$

Ex: Retorno na interação para correção de um erro (retorno simples)

⇒ Navegação de Retorno (*Undo*) ou Fechamento (*Close*), sucessivos:

$P_R \longrightarrow P_{J-1} \longrightarrow P_J \longrightarrow P_{J+1} \cdots P_J \cdots P_{J-1} \cdots P_R$

Ex: Retorno a um ponto qualquer da interação desfazendo o efeito causado pela última ação na interação.

⇒ Navegação direta entre nós da interação

$P_R \longrightarrow P_{J-1} \longrightarrow P_J \longrightarrow P_N$

Ex: Solicitação de ajuda em um ponto qualquer da interação.

⇒ Navegação de retorno (direto) à raiz:

$P_R \longrightarrow P_{J-1} \longrightarrow P_J \longrightarrow P_N \dashrightarrow P_R$

Ex: Retorno à janela principal a partir de um ponto qualquer da interação.

5.2 Construção de um Modelo de Navegação

A partir da representação genérica da navegação foi adotado o formalismo de redes de Petri colorida (CPN), para modelar a navegação.

Para construir e simular o modelo em CPN, foi utilizada a ferramenta *Design/CPN* [MSC 93]. Esta ferramenta possui capacidades que possibilitam criar, modificar, organizar, executar, depurar, examinar e validar redes de Petri colorida. A ferramenta possui:

- Um editor para criação e alteração de redes de Petri colorida (CPN - *Coloured Petri Nets*).
- Um verificador de sintaxe para validar as CPN.
- Um simulador para executar as CPN.
- Capacidades de monitoração interativa da execução e depuração.
- Facilidades para organizar uma rede de Petri colorida em módulos hierárquicos.
- Facilidades gráficas para mostrar os resultados da simulação.

Para facilitar a descrição do processo de modelagem em CPN o modelo da navegação foi instanciado para o *Browser-B* apresentado no Capítulo 4.

A interface do Navegador é composta por janelas (*windows*), botões (*buttons*) e *menus*. No modelo da navegação apresentado na Figura 5-2, considera-se que ao contexto da interação estão associados o estado atual da interação e um histórico contendo os estados anteriores. Um estado da interação corresponde a uma janela ativa e uma lista de ações que o usuário pode realizar neste ponto da interação. Desta forma, no modelo de navegação aqui apresentado, a interação com o usuário pode resultar na navegação entre janelas e/ou na atualização da lista de ações disponíveis. Quando da ação do usuário resultar uma mudança de janela, a representação correspondente no modelo de navegação foi denominada de "navegação". Por outro lado, quando da ação do usuário resultar apenas uma mudança na lista de ações disponíveis para o usuário sem que haja mudança de janelas, a representação correspondente no modelo de navegação foi denominada de "não navegação". Se ocorrer uma mudança para um estado anterior na interação, a representação corresponde a "*undo & close*" e finalmente se ocorrer a saída do sistema, a representação corresponde ao "*exit*".

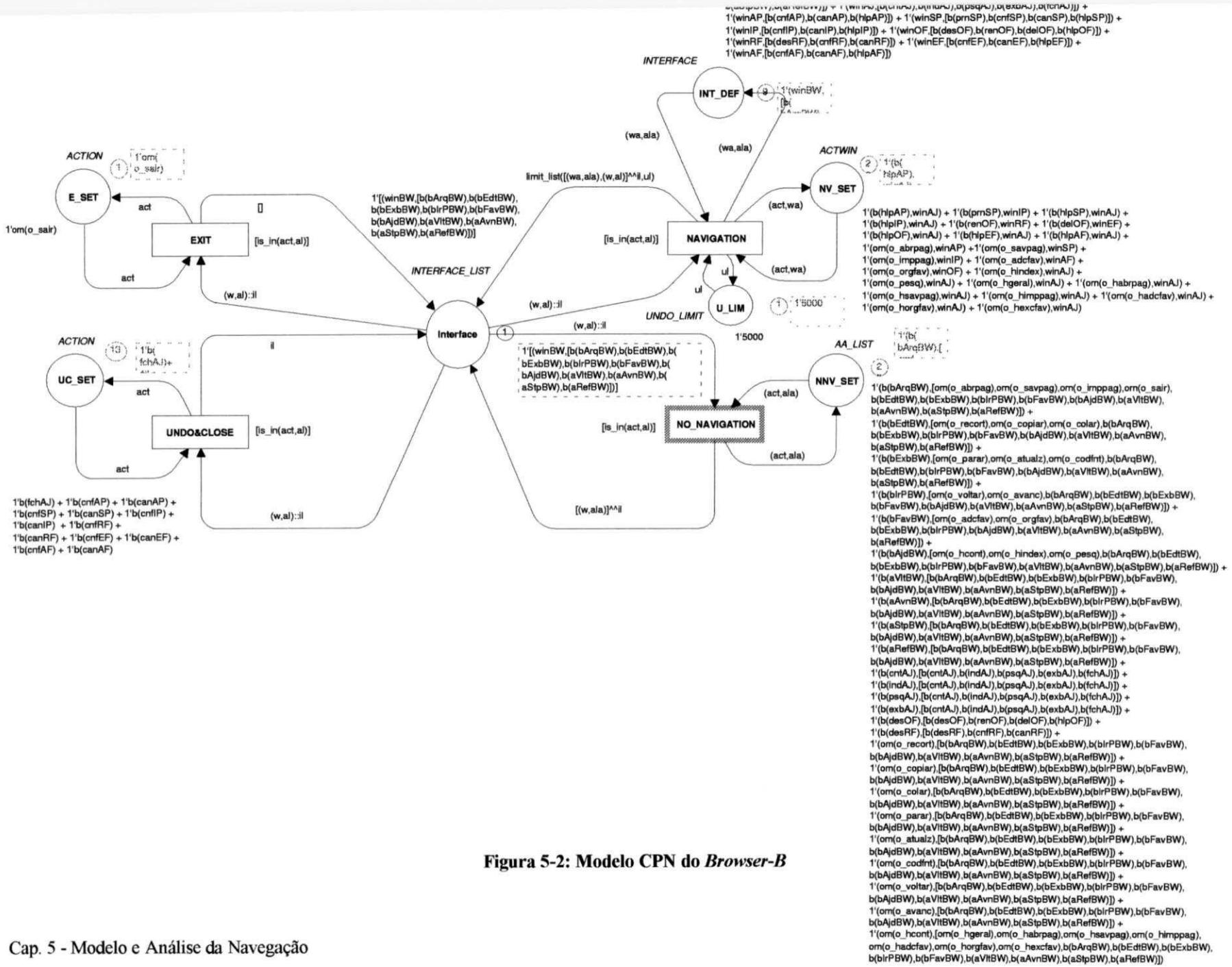

Figura 5-2: Modelo CPN do *Browser-B*

Tabela 5-1: Declarações Globais

```
color WINDOW          = with winBW | winAJ | winAP | winSP | winIP | winOF | winRF | winEF | winAF;
color BUTTON          = with bArqBW | bEdtBW | bExbBW | bIrPBW | bFavBW | bAjdBW | aVltBW | aAvnBW |
                             aStpBW | aRefBW | cntAJ | indAJ | psqAJ | exbAJ | fchAJ | cnfAP | canAP |
                             hlpAP | prnSP | cnfSP | canSP | hlpSP | cnfIP | canIP | hlpIP | desOF | renOF |
                             delOF | hlpOF | desRF | cnfRF | canRF | cnfEF | canEF | hlpEF |  cnfAF |
                             canAF | hlpAF;
color MENU            = with mArq | mEdt | mExb | mIrP | mFav | mAjd | smCnt;
color OP_MENU         = with o_abrpag | o_savpag | o_imppag | o_sair | o_recort | o_copiar | o_colar |
                             o_parar | o_atualz | o_codfnt | o_voltar | o_avanc | o_adcfav | o_orgfav |
                             o_hcont | o_hindex | o_pesq | o_hgeral | o_habrpag | o_hsavpag | o_himppag |
                             o_hadcfav | o_horgfav | o_hexcfav;
color ACTION          = union b:BUTTON + om:OP_MENU;
color ACTWIN          = product ACTION * WINDOW;
color ACTION_LIST     = list ACTION;
color AA_LIST         = product ACTION * ACTION_LIST;
color INTERFACE       = product WINDOW * ACTION_LIST;
color INTERFACE_LIST = list INTERFACE;
color UNDO_LIMIT      = int;

fun is_in(elem,[]) = false |
        is_in(elem,x::y) = if elem=x then true else is_in(elem,y);

fun take (_, 0)    = []
  | take([], _)    = raise Subscript
  | take(h::t, n) = h::take(t, n-1) handle Overflow => raise Subscript;

fun limit_list(ilist:INTERFACE_LIST, u_lim:int):INTERFACE_LIST =
  if length(ilist) > u_lim then take(ilist,u_lim) else ilist;

var w: WINDOW;
var wa: WINDOW;
var bt: BUTTON;
var om1: OP_MENU;
var act: ACTION;
var al: ACTION_LIST;
var ala: ACTION_LIST;
var il: INTERFACE_LIST;
var ul: UNDO_LIMIT;
```

O modelo CPN da representação genérica da navegação possui sete lugares e quatro transições. As transições no modelo representam as ações apresentadas na Seção 5.1. Na Tabela 5-1 são apresentadas as declarações do modelo com seu conjunto de cores, funções, variáveis e constantes, utilizadas nas operações que envolvem a rede. Através das cores são representados os objetos da interface considerados na representação genérica da navegação. A seguir é apresentado o conjunto de cores do modelo da navegação.

cor *WINDOW*	→ representa o conjunto das janelas existentes
cor *BUTTON*	→ representa o conjunto dos botões existentes
cor *MENU*	→ representa o conjunto dos menus existentes
cor *OP_MENU*	→ representa o conjunto das opções de menu
cor *ACTION*	→ representa a união de um botão ou uma opção de menu
cor ACTWIN	→ cor auxiliar representando um par (ação, janela)
cor *ACTION_LIST*	→ representa uma lista de ações, que contém botões ou opções de menu disponíveis

cor AA_LIST → cor auxiliar representando um par (ação, lista de ações)
cor *INTERFACE* → representa um par (janela, lista de ações)
cor *INTERFACE_LIST* → representa uma lista de interfaces
cor UNDO_LIMIT → representa o limite de elementos para a lista de interfaces

O conjunto de lugares, exemplificado com o *Browser-B*, é descrito a seguir:

Interface: Contém uma ficha da cor *INTERFACE LIST* que representa o estado da interface. Esta estrutura de dados é uma lista utilizada como uma pilha, de forma a permitir que sejam realizadas operações de retorno na interação. Na lista de interfaces o primeiro elemento representa sempre o estado atual da interface, enquanto que os demais elementos da lista representam, na ordem inversa de ocorrência, os estados anteriores da interface percorridos pelo usuário durante a interação. Na Tabela 5-2 é mostrado a ficha para a marcação inicial do *Browser-B*.

Tabela 5-2: Exemplo da Ficha *Interface_List* (Marcação Inicial)

```
1`[(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]]|
```

U_LIM: Contém uma ficha da cor *UNDO_LIMIT* que representa uma limitação para o tamanho da lista de interfaces, restringindo assim o número permitido de operações de retorno na interação. Esta ficha é mostrada na Tabela 5-3.

Tabela 5-3: Lugar de Limitação do Tamanho da Lista de Interfaces (U_LIM)

```
1`5000
```

NNV_SET: Contém fichas da cor *AA_LIST*. Neste lugar estão presentes as ações que não provocam navegação na interface, ou seja, apenas mudam as ações disponíveis para o usuário. Esta ficha é mostrada na Tabela 5-4.

Tabela 5-4: Lugar "*No_Navigation*" (NNV_SET)

```
1`(b(bArqBW),[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),
   b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bEdtBW),[om(o_recort),om(o_copiar),om(o_colar),b(bArqBW),b(bExbBW),b(bIrPBW),b(bFavBW),
   b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bExbBW),[om(o_parar),om(o_atualz),om(o_codfnt),b(bArqBW),b(bEdtBW),b(bIrPBW),b(bFavBW),
   b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bIrPBW),[om(o_voltar),om(o_avanc),b(bArqBW),b(bEdtBW),b(bExbBW),b(bFavBW),b(bAjdBW),
   b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bFavBW),[om(o_adcfav),om(o_orgfav),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bAjdBW),
   b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bAjdBW),[om(o_hcont),om(o_hindex),om(o_pesq),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),
   b(bFavBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(aVltBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(aAvnBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(aStpBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(aRefBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(cntAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(indAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(psqAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(exbAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(desOF),[b(desOF),b(renOF),b(delOF),b(hlpOF)]) +
1`(b(desRF),[b(desRF),b(cnfRF),b(canRF)]) +
1`(om(o_recort),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_copiar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_colar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_parar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_atualz),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_codfnt),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_voltar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_avanc),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_hcont),[om(o_hgeral),om(o_habrpag),om(o_hsavpag),om(o_himppag),om(o_hadcfav),
   om(o_horgfav),om(o_hexcfav),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)])
```

NV_SET: Contém fichas da cor *ACTWIN*. Neste lugar estão presentes as ações que provocam navegação na interface. Esta ficha é mostrada na Tabela 5-5.

Tabela 5-5: Lugar *Navigation* (NV_SET)

```
1`(b(hlpAP),winAJ) + 1`(b(prnSP),winIP) + 1`(b(hlpSP),winAJ) +
1`(b(hlpIP),winAJ) + 1`(b(renOF),winRF) + 1`(b(delOF),winEF) +
1`(b(hlpOF),winAJ) + 1`(b(hlpEF),winAJ) + 1`(b(hlpAF),winAJ) +
1`(om(o_abrpag),winAP) +1`(om(o_savpag),winSP) +
1`(om(o_imppag),winIP) + 1`(om(o_adcfav),winAF) +
1`(om(o_orgfav),winOF) + 1`(om(o_hindex),winAJ) +
1`(om(o_pesq),winAJ) + 1`(om(o_hgeral),winAJ) + 1`(om(o_habrpag),winAJ) +
1`(om(o_hsavpag),winAJ) + 1`(om(o_himppag),winAJ) + 1`(om(o_hadcfav),winAJ) +
1`(om(o_horgfav),winAJ) + 1`(om(o_hexcfav),winAJ)
```

INT_DEF: Contém fichas da cor *INTERFACE*. Neste lugar está presente a definição da interface, ou seja, o conjunto de janelas e a lista de ações correspondentes a cada janela. Esta ficha é mostrada na Tabela 5-6.

Tabela 5-6: Lugar "*Interface_Definition*" (INT_DEF)

```
1`(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) + 1`(winAJ,[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(winAP,[b(cnfAP),b(canAP),b(hlpAP)]) + 1`(winSP,[b(prnSP),b(cnfSP),b(canSP),b(hlpSP)]) +
1`(winIP,[b(cnfIP),b(canIP),b(hlpIP)]) + 1`(winOF,[b(desOF),b(renOF),b(delOF),b(hlpOF)]) +
1`(winRF,[b(desRF),b(cnfRF),b(canRF)]) + 1`(winEF,[b(cnfEF),b(canEF),b(hlpEF)]) +
1`(winAF,[b(cnfAF),b(canAF),b(hlpAF)])
```

UC_SET: Contém fichas da cor *ACTION*. Neste lugar estão presentes as ações que provocam retorno na interação, ou seja, situações de desfazer (*undo*) e fechar (*close*). Esta ficha é mostrada na Tabela 5-7.

Tabela 5-7: Lugar "*Undo & Close*" (UC_SET)

```
1`b(fchAJ) + 1`b(cnfAP) + 1`b(canAP) + 1`b(cnfSP) + 1`b(canSP) + 1`b(cnfIP) +
1`b(canIP)  + 1`b(cnfRF) + 1`b(canRF) + 1`b(cnfEF) + 1`b(canEF) +
1`b(cnfAF) + 1`b(canAF)
```

E_SET: Contém fichas da cor *ACTION*. Neste lugar estão presentes as ações que provocam saída do sistema, finalizando a interação. Esta ficha é mostrada na Tabela 5-8.

Tabela 5-8: Lugar "*Exit*" (E_SET)

```
1`om(o_sair)
```

Esta representação da navegação se propõe a ser de caráter genérico de modo a representar interfaces baseadas em janelas, menus e botões. Desta forma, a definição dos lugares, conforme foi apresentado, foi concebido com o propósito de possibilitar a modelagem de diferentes situações destas interfaces de forma rápida, simples e sem demandar conhecimentos específicos em redes de Petri para construção destes modelos.

Para representar a navegação de uma outra interface neste modelo, é necessário apenas adequar o conjunto de cores de modo a "instanciar" as janelas, botões, menus e a lista dos elementos de navegação. A simplicidade deste procedimento dispensa o projetista de conhecimentos aprofundados da ferramenta *Design/CPN* [MSC 93] e da linguagem *CPN-ML* [UAA 96b].

O conjunto de transições não necessita modificações para representar a navegação em outras interfaces que não a do *Browser-B*. As transições representam de forma genérica as situações de saída do sistema (*EXIT*), retorno na interação (*UNDO & CLOSE*), avanço na interação (*NAVIGATION*) e alteração das opções disponíveis (*NO_NAVIGATION*).

Para o disparo das transições, foram definidas três funções para a rede: *is_in*, *take* e *limit_list*. Estas funções são descritas a seguir:

```
fun is_in ( elem, [ ] ) = false |
      is_in ( elem, x :: y ) = if elem = x then true else is_in ( elem, y );
```

- Esta função verifica se um elemento se encontra na lista. Possui um caráter genérico porque nela não estão definidos os tipos dos elementos, podendo ser utilizada para verificar elementos inteiros, strings, etc. Esta função é utilizada para verificar se uma determinada ação se encontra na lista de ações.

```
fun take ( _, 0 )   = [ ]
      | take([], _)= raise Subscript
      | take(h::t, n) = h::take(t, n-1) handle Overflow => raise Subscript;
```

- Esta função retorna os *n* primeiros elementos de uma lista. É utilizada pela função *limit_list* para manter sob controle o tamanho da lista de interfaces, ou seja, limitando o crescimento de acordo com restrições físicas de implementação.

```
fun limit_list ( ilist:INTERFACE_LIST, u_lim:int ): INTERFACE_LIST =
      if length(ilist) > u_lim then take(ilist, u_lim) else ilist;
```

- Esta função limita o elemento *ilist* a um tamanho não superior a *u_lim*. Esta função é utilizada para manter a lista de interfaces com um tamanho máximo, definido pela ficha no lugar U_LIM.

Associada a cada uma das transições, existe uma restrição que deve ser satisfeita antes da habilitação da transição. Esta definição adicional é denominada "Guarda" e consiste de

uma expressão booleana, cujo resultado é Verdadeiro ou Falso. As guardas são utilizadas para introduzir as restrições associadas a cada transição. Para avaliação das guardas, em cada transição é verificado se a ação no topo da lista (ações para a janela ativa) encontra-se presente no lugar associado o qual contém os elementos de interface destinados aquela situação em particular.

Por exemplo, para que a transição *UNDO & CLOSE* dispare, é necessário que para a janela ativa esteja disponível uma ação correspondente, e que deve estar presente no lugar *UC_SET*. Quando esta transição disparar, o controle retorna à janela anterior removendo o elemento do topo da lista de interfaces.

Durante a simulação, percebeu-se para o caso do navegador, a possibilidade do usuário atingir a janela "Excluir Favoritos" (winEF) e retornar para a janela principal (winBW), retornando para o ponto inicial da interação, o que é desejado. Contudo, esta operação pode repetir-se indefinidamente conduzindo a um *loop* infinito. Como conseqüência, ocorre uma explosão de estados na rede, uma vez que o estado nunca se repete.

A passagem por um mesmo ponto da interface difere dos anteriores pelo fato de trabalhar-se com uma lista (necessária para representar situações de *undo*), onde a cada nova ação é acrescentado o elemento correspondente. Para evitar que isto ocorra, pode-se incluir restrições quanto a caminhos repetidos, mas isto pode interferir na realização de determinadas tarefas. Apesar de reconhecer a importância da inclusão destas restrições, elas não serão tratadas neste trabalho. Para efeitos de simulação, o botão "*GoTo Main*" (ligação entre a janela winEF e winBW) será retirado da modelagem e esta condição de retorno não será tratada na análise da navegação.

Para ilustrar o disparo das transições e a alteração no valor da ficha *INTERFACE_LIST*, é apresentado a seguir na Tabela 5-9 uma seqüência de interação. A esta seqüência denominou-se Plano da Tarefa ou Cenário, que consiste de uma seqüência de passos na interação necessários para realizar a tarefa.

Tabela 5-9: Exemplo de seqüência de interação (Plano da Tarefa ou Cenário)

(1)	A janela ativa inicialmente é a janela principal (winBW).
(2)	É selecionada a opção "Abrir Página" no menu Arquivo (mArq), conduzindo à janela Abrir Página (winAP).
(3)	Nesta janela é digitado o endereço desejado e pressionado o botão de confirmação (cnfAP), retornando à janela principal.
(4)	É selecionada a opção "Salvar Página" no menu Arquivo (mArq), conduzindo à janela Salvar Página (winSP).
(5)	Na janela winSP é pressionado o botão de Imprimir Página (prnSP), conduzindo à janela Imprimir Página (winIP).
(6)	Na janela winIP é pressionado o botão de confirmação (cnfIP), retornando à janela winSP.
(7)	Na janela winSP é pressionado o botão de cancelar (canSP), retornando à janela principal.
(8)	É selecionada a opção "Abrir Página" no menu Arquivo (mArq), conduzindo à janela Abrir Página (winAP), onde é confirmada a abertura de uma nova página (cnfAP), e retornado o controle para à janela principal.
(9)	É pressionado o botão Voltar (aVltBW).
(10)	É pressionado o botão Avançar (aAvnBW).
(11)	É selecionada a opção "Imprimir Página" no menu arquivo (mArq), conduzindo a janela Imprimir Página (winIP).
(12)	É pressionado o botão de confirmação (cnfIP), retornando para à janela principal.
(13)	É pressionado o botão Parar Carregamento da Página (aStpBW).
(14)	É solicitada Ajuda através da seleção da opção de menu "Índice" do menu Ajuda (mAjd).
(15)	É pressionado o botão de finalização da Ajuda (fchAJ), retornando para à janela principal.
(16)	É selecionada a opção "Sair" no menu arquivo (mArq), finalizando a execução do sistema.

A única ficha que teve seu valor modificado foi a ficha *INTERFACE_LIST*. As demais não foram alteradas devido à estrutura da rede. A representação do exemplo, para a ficha *INTERFACE_LIST* é:

(1)	`1`[(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW), b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]`

- Esta ficha representa a marcação inicial, que define a janela winBW como janela ativa e que as ações disponíveis para o usuário são representadas pelos botões bArqBW, bEdtBW, bExbBW, bIrPBW, bFavBW, bAjdBW, aVltBW, aAvnBW, aStpBW e aRefBW os quais podem ser selecionados.

(2)	1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]
	1`[(winAP,[b(cnfAP),b(canAP),b(hlpAP)]),(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Na simulação é pressionado o botão Arquivo (act = b(bArqBW)), fazendo com que o menu seja exibido e, em seguida é selecionada a opção Abrir Página (act = om(o_abrpag)). Com o pressionamento do botão bArqBW, a transição *NO_NAVIGATION* dispara aumentando o número de opções para o usuário. Com a seleção da opção o_abrpag a transição *NAVIGATION* dispara conduzindo a uma nova janela. O novo elemento incorporado à lista de interfaces define que a janela winAP é agora a janela ativa.

(3)	1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Com o pressionamento do botão de confirmação (act = b(cnfAP)) a transição *UNDO & CLOSE* dispara, removendo o elemento do topo da lista e definindo a janela WinBW como janela ativa.

(4)	1`[(winSP,[b(prnSP),b(cnfSP),b(canSP),b(hlpSP)]),(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Selecionado-se a opção Salvar Página (act = om(o_savpag)) , a transição *NAVIGATION* dispara, conduzindo à janela winSP.

(5)	1`[(winIP,[b(cnfIP),b(canIP),b(hlpIP)]),(winSP,[b(prnSP),b(cnfSP),b(canSP),b(hlpSP)]),(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Selecionando-se o botão Imprimir Página (act = b(prnSP)), a transição *NAVIGATION* dispara novamente tornando a janela winIP ativa.

(6)	1`[(winSP,[b(prnSP),b(cnfSP),b(canSP),b(hlpSP)]),(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão de confirmação (act = b(cnfIP)), a transição *UNDO & CLOSE* dispara removendo o elemento do topo da lista e definindo a janela winSP como janela ativa.

(7)	1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão de cancelar (act = b(canSP)), a transição *UNDO & CLOSE* dispara novamente removendo o elemento do topo da lista e definindo a janela winBW como ativa.

(8)	1`[(winAP,[b(cnfAP),b(canAP),b(hlpAP)]),(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]
	1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Selecionando-se a opção Abrir Página (act = om(o_abrpag)), é disparada a transição *NAVIGATION* conduzindo à janela winAP. Nesta janela é confirmada a abertura da nova página (act = b(cnfAP)) disparando a transição *UNDO & CLOSE*, removendo o elemento do topo da lista e tornando a janela winBW ativa.

(9)	1`[(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão Voltar (act = b(aVltBW)), é disparada a transição NO_NAVIGATION, simulando a permanência na janela winBW, sendo modificada apenas a lista de opções para o usuário.

(10)	1`[(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão Avançar (act = b(aAvnBW)), é disparada novamente a transição *NO_NAVIGATION*, permanecendo na mesma janela e com as mesmas ações disponíveis.

(11)	1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]
	1`[(winIP,[b(cnfIP),b(canIP),b(hlpIP)]),(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão Arquivo (act = b(bArqBW)), é disparada a transição *NO NAVIGATION*, alterando as opções disponíveis. Selecionando a opção Imprimir Página (act = om(o_imppag)), a transição *NAVIGATION* dispara conduzindo à janela winIP

(12)	1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão de confirmação (act = b(cnfIP)), é disparada a transição *UNDO & CLOSE*, removendo o elemento do topo da lista e tornando a janela winBW ativa.

(13)	1`[(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão Parar (act = b(aStpBW)), é disparada a transição *NO_NAVIGATION*, permanecendo na janela winBW, sendo alterada apenas a lista de opções para o usuário.

(14)	1`[(winBW,[om(o_hcont),om(o_hindex),om(o_pesq),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]
	1`[(winAJ,[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]),(winBW,[om(o_hcont),om(o_hindex),om(o_pesq),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão Ajuda (act = b(bAjdBW)), é disparada a transição *NO_NAVIGATION*, aumentando o número de opções disponíveis para o usuário. Selecionando-se a opção Índice (act = om(o_hindex)), é disparada a transição *NAVIGATION* conduzindo à janela winAJ.

(15)	1`[(winBW,[om(o_hcont),om(o_hindex),om(o_pesq),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]

- Pressionando-se o botão de finalizar a Ajuda (act = b(fchAJ)), é disparada a transição *UNDO & CLOSE*, removendo o elemento do topo da lista e retornando à janela winBW.

(16)	`1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]`
	`1`[]`

- Pressionando-se o botão Arquivo (act = b(bArqBW)), é disparada a transição *NO_NAVIGATION*, alterando a quantidade de opções disponíveis. Dentre estas opções é selecionada a opção de Saída do sistema (act = om(o_sair)), que dispara a transição *EXIT*, eliminando todos os elementos da lista de interfaces e desabilitando todas as transições, finalizando desta forma a simulação.

Os resultados da simulação apresentados acima foram obtidos com a ferramenta *Design/CPN* através de uma simulação interativa. Além da simulação foi verificado um conjunto de propriedades da rede que modela a navegação, as quais serão discutidas na próxima seção.

5.3 ANÁLISE DA NAVEGAÇÃO

Aos mecanismos de navegação disponíveis ao usuário podem ser associados requisitos de qualidade tais como:

- Existência de caminhos diretos ou indiretos entre pontos da interação que viabilizem a realização das tarefas.
- Existência de caminhos de acesso otimizados, que não sejam complexos ou extensos.
- Acesso à saída do sistema de qualquer ponto da interação.
- Reiniciabilidade – acesso ao ponto de início de modo que a mesma tarefa possa ser realizada mais de uma vez.
- Acesso à ajuda de qualquer ponto da interação.
- Reversibilidade do caminho percorrido entre dois pontos da interação quando a tarefa demandar.
- Existência de caminhos alternativos entre dois pontos da interação para realização da tarefa.
- Inexistência de bloqueios na navegação.

A partir das características da navegação de um modelo de tarefa e dos requisitos de qualidade da navegação que se aplicarem, pode ser feita uma análise de modo a verificar se os mecanismos propostos satisfazem aos requisitos de navegação da tarefa [ST 98].

5.3.1 Análise de caminhos

As características que podem ser analisadas a partir do modelo construído neste trabalho foram: existência de caminhos, acesso à saída do sistema, reiniciabilidade, acesso à ajuda, reversibilidade do caminho, existência de caminhos alternativos. Uma vez que o modelo da navegação não trata a ativação e desativação de objetos da interface tais como: botões ou opções de menu, não é possível analisar a existência de bloqueios na navegação.

Na ferramenta *Design/CPN* existem consultas padrões que podem ser executadas sem a necessidade de implementação de código. Uma das funções disponíveis determina a seqüência de ocorrência entre uma marcação inicial e uma marcação final. Esta função, denominada de '*Reacheable*' é essencial para a análise de caminhos, contudo apresenta apenas o caminho de menor comprimento.

Para sistemas de interfaces, além de identificar o menor caminho (caminho ótimo para realização da tarefa), é interessante saber-se o conjunto de todos os possíveis caminhos entre dois pontos da interface, com o objetivo de averiguar o excesso ou ausência de caminhos alternativos para a realização da tarefa. Desta forma, foi escrita a função '*AllPath*', que retorna todos os caminhos entre duas marcações da rede, uma vez que a ferramenta não dispõe de uma função para esta finalidade. Os parâmetros são as marcações inicial e final para pesquisa, e o tamanho do caminho a ser analisado. O valor zero faz com que sejam retornados os caminhos de qualquer comprimento. A função *Allpath*, escrita na linguagem CPN-ML, é mostrada na Tabela 5-10.

Esta função só deve ser aplicada nos casos em que o grafo de ocorrência seja finito. Inicialmente são verificados todos os nós de saída adjacentes ao nó correspondente a marcação inicial da pesquisa (função *OutNodes* da ferramenta *Design/CPN*). A partir do exame destes nós de saída, vai sendo formada uma lista com o caminho em direção ao nó correspondente a marcação final da pesquisa, respeitando-se o comprimento especificado e com a restrição de que um nó que já se encontre previamente na lista não seja repetido.

Tabela 5-10: Função AllPath

```
001  fun member (x, [])    = false
002    | member (x, h::t) = x = h orelse member (x,t);
003
004  fun AllPathAux (aBegin, aEnd, aOrdem, aList, aAux) =
005    let val n = ref 1 and k = ref 0 and
006        mOutNodes = ref [] and mSavPath = ref [] and
007        mList = ref aList and mAux = ref aAux
008    in
009      ( mSavPath := !mAux;
010        mAux := !mAux @ (aBegin :: nil);
011
012        if aBegin = aEnd andalso (aOrdem = 1 orelse length(!mAux) = aOrdem) then
013          ( mList := !mList @ (!mAux :: nil) )
014        else
015          ( mOutNodes := OutNodes(aBegin);
016            n := length(!mOutNodes);
017            while !k < !n do
018              if member( nth(!mOutNodes, !k), !mAux ) then
019                ( k := !k + 1 )
020              else
021                ( mList := AllPathAux( nth(!mOutNodes, !k),aEnd,aOrdem,!mList,!mAux);
022                  k := !k + 1 )
023          );
024
025        mAux := !mSavPath
026      );
027
028      !mList
029    end;
030
031  fun AllPath(nB, nE, nOrd) =
032    let val mList = ref [] and mAux = ref []
033    in
034      ( mList := AllPathAux(nB, nE, nOrd+1, !mList, !mAux) );
035      !mList
036    end;
```

A função apresenta recursividade e termina quando for atingido o nó final ou quando não existirem mais nós para serem expandidos. Contudo, apresenta resultados enumeráveis uma vez que utiliza um número finito de nós de saída (Função *OutNodes*) e não existe a reavaliação de um nó que já esteja previamente contido no caminho sob análise.

No quadro a seguir é mostrada a utilização desta função entre as marcações 1 e 12 do espaço de estados da rede do *Browser-B*. A função retornou todos os caminhos independentemente do comprimento.

AllPath(1,12,0)

Retorna: [[1,7,12], [1,6,7,12], [1,6,5,7,12], [1,6,5,4,7,12], [1,6,5,4,3,7,12], [1,6,5,4,3,2,7,12], [1,6,5,4,3,2,8,7,12], [1,6,5,4,2,7,12], [1,6,5,4,2,3,7,12], [1,6,5,4,2,8,7,12], [1,6,5,4,2,8,3,7,12], [1,6,5,3,7,12], ...]: Node list list

Com esta função é possível analisar: a existência de pelo menos um caminho, a existência de caminhos alternativos, a adequação do número de caminhos existentes, a extensão dos caminhos projetados e a complexidade dos caminhos disponíveis.

5.3.2 Propriedades do Modelo da Navegação

As propriedades que serão tratadas na análise deste trabalho, são as propriedades dinâmicas que caracterizam o comportamento da rede. Como já foi mencionado, métodos formais são utilizados para provar estas propriedades. A seguir estas propriedades são apresentadas informalmente.

Alcançabilidade - uma marcação da rede é dita alcançável a partir de uma outra marcação, se existe uma seqüência de disparos que transforma a primeira marcação na segunda. Esta é uma base formal para estudar as propriedades dinâmicas de um sistema.

Limitação – os limites superior e inferior de um lugar na rede delimitam o número de fichas de uma cor neste lugar.

Estado ou Marcação Recorrente – uma marcação recorrente (*home*) é uma marcação para a qual é sempre possível retornar.

Vivacidade – estabelece que um conjunto de elementos de ligação permanece ativo. Esta propriedade está diretamente relacionada à ausência de bloqueios (*deadlocks*).

Eqüidade – estabelece quão freqüentemente as transições ocorrem. Assim, se duas transições se encontram com uma relação justa, o número de vezes que uma dispara enquanto a outra não dispara é limitado.

Considerando que a navegação foi representada pelo modelo CPN apresentado na Seção 5.2, estas características podem ser expressas em termos de propriedades da rede, conforme é ilustrado a seguir.

Reversibilidade na interação

O retorno na interação não se aplica necessariamente a todos os seus estados. Esta característica pode ser verificada na rede através da propriedade de alcançabilidade, utilizando as consultas disponíveis na ferramenta *Design/CPN*, ou através da função *AllPath*. Ao realizar uma consulta com a função '*Reacheable*', disponível na ferramenta *Design/CPN*, é

apresentado apenas um dos caminhos (o menor) entre duas marcações. A função *AllPath* é mais apropriada porque apresenta uma lista contendo todos os caminhos entre duas marcações (dois pontos da interação), possibilitando verificar a adequação dos caminhos existentes.

Caminhos de acesso a pontos específicos

Por razões de usabilidade, o acesso à mecanismos de ajuda deve ser permitido a partir de qualquer ponto da interação. A análise desta característica da navegação é possível a partir da análise das marcações da rede. Partindo de um estado M_k deve-se verificar se é possível alcançar o estado M_j, no qual o primeiro elemento[8] da ficha *INTERFACE_LIST* corresponde à solicitação de ajuda.

Reiniciabilidade

A maioria dos sistemas de interface possui uma janela principal onde inicia a interação. Para verificar se o sistema é reinicializável (possui a capacidade de retornar ao estado inicial) deve-se verificar a existência da marcação inicial na lista de marcações recorrentes (*home markings*) mostradas no relatório da ferramenta *Design/CPN*. Do ponto de vista de redes de Petri, uma marcação recorrente representa uma marcação alcançável à partir de qualquer marcação da rede, mediante uma seqüência de disparos.

Observe que se todas as marcações da rede forem marcações recorrentes, pode-se interpretar como a existência de caminhos de acesso entre todos os pontos da interface, característica que pode ser encontrada em algumas situações de interação. Para que esta característica venha a ser analisada é necessário eliminar todas as saídas do sistema, retirando todas as fichas do lugar *E_SET*. Nesta situação, pode-se então verificar quais são as novas marcações recorrentes.

- Se todas as marcações forem recorrentes, existem caminhos de acesso entre todos os pontos da interação.
- Se ao retirar as marcações correspondentes ao estado *EXIT*, ainda existirem marcações mortas (*dead markings*) pode-se concluir que existem situações que comprometem a navegabilidade do sistema. Ou seja, situações a partir das quais o usuário não tem opções de navegação.

[8] Botão de ajuda, opção de menu, texto de um comando

Caminhos para acesso à saída do sistema

Nos sistemas de interface podem existir uma ou mais opções de saída. Para verificar o acesso à saída do sistema a partir de qualquer ponto da interação, deve-se verificar se todas as marcações mortas constantes no relatório da ferramenta *Design/CPN*, correspondem à saída do sistema.

A saída do sistema é caracterizada pelo disparo da transição *EXIT*. Quando ocorre o disparo desta transição, a lista de interfaces é esvaziada fazendo com que a marcação correspondente seja uma marcação morta (*dead marking*), ou seja, não estarão disponíveis opções de navegação. Para que o usuário seja capaz de finalizar a interação em qualquer ponto da navegação, esta marcação morta deve ser sempre alcançável, ou seja, ela deve ser morta e ao mesmo tempo recorrente. Para verificar a ocorrência do disparo desta transição, verificam-se os nós de entrada e analisa-se a expressão do arco, buscando a ação (pressionamento de botões, seleção de opções de menu) que levou à saída do sistema.

5.4 ANÁLISE DA NAVEGAÇÃO NO *BROWSER-B*

Após o cálculo do grafo de ocorrência [UAA 96a], a ferramenta *Design/CPN* gera um arquivo texto contendo um relatório padrão. Dentre as informações fornecidas, aquelas que merecem destaque para a análise dos tópicos acima são:

O Relatório da ferramenta *Design/CPN*

- Estatísticas – tamanho e *status* do grafo de ocorrência.

  No grafo de ocorrência são representadas todas as marcações alcançáveis. Os nós representam cada uma das marcações, enquanto que os arcos representam a existência de um elemento de ligação entre os nós. O *Status* pode ser *Full* ou *Partial*, indicando se o grafo foi completamente construído ou não. O valor de *secs* indica o tempo decorrido durante a simulação.
- Propriedades de Limitação.

  Para a rede modelada, exibe os valores dos limites superior e inferior do número de fichas para cada lugar.
- Propriedades de Recorrência (*Home*).

  Exibe a lista de todas as marcações recorrentes.

- Propriedades de Vivacidade (marcações mortas).

  Mostra a lista de todas as marcações mortas (*dead markings*).

Na Tabela 5-11 é apresentado o resultado obtido na ferramenta *Design/CPN*.

Tabela 5-11: Resultados da Ferramenta *Design/CPN*

```
Statistics
------------------------------------------------------------------------
  Occurrence Graph
    Nodes:  26
    Arcs:   168
    Secs:   1
    Status: Full


Boundedness Properties
------------------------------------------------------------------------
  Best Integers Bounds    Upper       Lower
  New'E_SET 1             1           1
  New'INT_DEF 1           9           9
  New'Interface 1         1           1
  New'NNV_SET 1           25          25
  New'NV_SET 1            23          23
  New'UC_SET 1            13          13
  New'U_LIM 1             1           1


Home Properties
------------------------------------------------------------------------``
  Home Markings:  None


Liveness Properties
------------------------------------------------------------------------
  Dead Markings:  [12]
```

Como pode-se observar no resultado da ferramenta, não existe um estado recorrente ("*home marking*"). Para o caso de interfaces, o estado de saída do sistema (*exit*) deve estar acessível a partir de qualquer ponto da interface. Conclui-se então que existe falta de algum componente de interligação entre janelas da interface, que possibilite acesso a saída do sistema.

Para verificar o que está errado, é feita uma análise do modelo da interação e do modelo da tarefa, realizando as devidas correções. Este método será descrito com maiores detalhes no Capítulo 7.

5.5 CONCLUSÕES

Neste capítulo foi apresentada uma estrutura genérica de navegação para sistemas de interface utilizando o formalismo de redes de Petri e, foi proposto um conjunto de propriedades da rede que estão associadas a requisitos ergonômicos da navegação. Nesta etapa do método é possível detectar problemas de concepção da estrutura de navegação, por exemplo a impossibilidade de alcançar a saída do sistema a partir de um ponto da navegação. Tendo analisado o modelo da navegação, as próximas etapas do método se fundamentam na análise da usabilidade com base na interação do usuário com um protótipo.

6 PROTOTIPAÇÃO E VALIDAÇÃO DA INTERFACE

A coleta automática de dados pode ser utilizada para diagnosticar problemas que não podem ser detectados durante o processo de observação dos usuários. O registro dos dados pode revelar estatísticas acerca de uma ou várias ações em particular, como por exemplo a utilização de um elemento da interface. Uma boa parte das informações obtidas com a utilização desta técnica poderia ser obtida através da observação direta, porém o observador humano tende a não perceber uma série de acontecimentos que muitas vezes são de extrema importância para o sequenciamento da interação. A coleta automática de dados registra o que aconteceu durante a interação sem sofrer a influência da opinião do avaliador ou do usuário sobre estas ocorrências.

Este capítulo apresenta a etapa de validação de protótipos com base em métodos de avaliação não intrusivos (métodos que não alteram o comportamento do usuário durante a interação). Para apoiar a validação foi desenvolvida uma ferramenta para a captura de dados da interação. Neste capítulo também é descrita a ferramenta de coleta desenvolvida para o método e os procedimentos para sua utilização. Finalmente é apresentada a validação do protótipo do estudo de caso descrito no Capítulo 4.

6.1 CONSTRUÇÃO DE UM PROTÓTIPO

O *OpenWindows* é um ambiente de janelas que utiliza interfaces gráficas. Este ambiente utiliza uma interface baseada no padrão *OpenLook* e opera em estações de trabalho SUN, com um sistema operacional compatível com o sistema operacional UNIX. Para o

desenvolvimento de interfaces neste ambiente, foi utilizada a ferramenta *Devguide - OpenWindows Developer's Guide* [Sun 91].

O *Devguide* é uma ferramenta que apresenta muitos recursos para a construção de interfaces, tornando este processo simples e rápido. Possibilita a construção de interfaces através de uma programação visual e interativa. Pode ser utilizado por pessoas que não sejam programadores tais como projetistas de interfaces e gerenciadores de projetos.

O *Devguide*, oferece elementos pré-definidos para a construção das interfaces: *Base Windows, Pop-up Windows, Control Areas, Canvas Panes, Term Panes, Text Panes, Buttons, Messages, Settings, Text Fields, Sliders, Gauges, Lists* e *Menus*. Uma descrição detalhada destes elementos e como utilizá-los na construção de um protótipo se encontra em [ST 95] e em [ST 96a].

As interfaces geradas pelo *DevGuide*, são armazenadas em arquivos que possuem uma estrutura interna de acordo com a sintaxe GIL (*Graphical Intermediate Language*). Um arquivo GIL armazena a definição da interface utilizando uma linguagem de descrição geral que identifica cada elemento através de uma lista de propriedades. Estes arquivos podem ser recuperados posteriormente para modificação do protótipo ou para a geração do código fonte da interface. Na figura 6-1, é mostrado o procedimento para criar uma interface utilizando-se o *Devguide*.

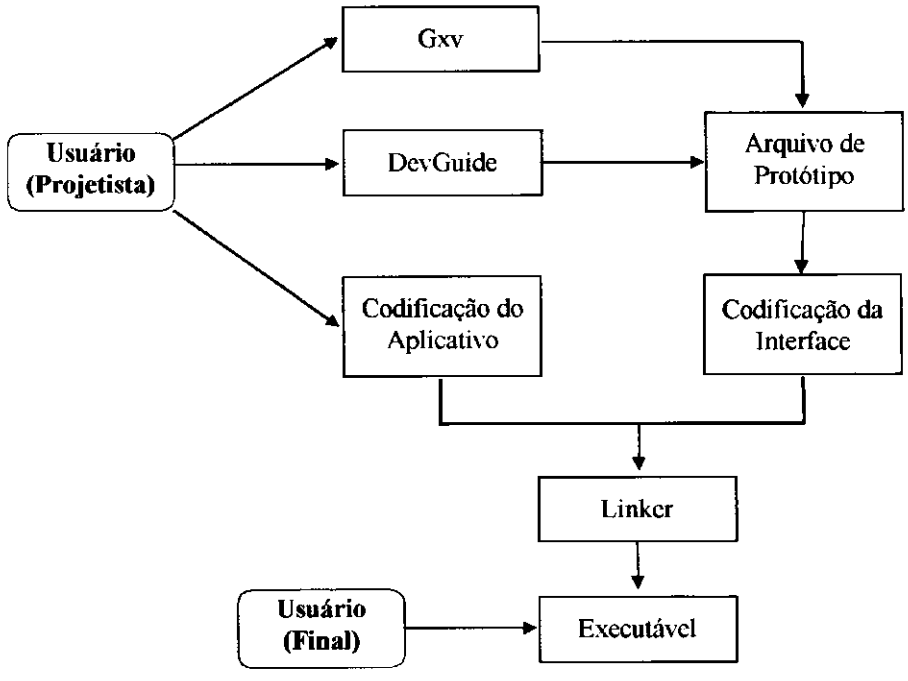

Figura 6-1: Procedimento para Construção de um Protótipo

Após obtenção do arquivo de protótipo, a partir do programa **Gxv** (*Devguide's companion program*), é possível ler os arquivos de interfaces gerados pelo *Devguide* e realizar as chamadas às rotinas do *Xview*, incluindo todo o código fonte em linguagem C necessário para criar a interface. O *Xview* (***X** Window-System-based **V**isual/**I**ntergrated **E**nvironment for **W**orkstations*) [Hel 91], é um *toolkit* para construção de interfaces gráficas que proporciona ao usuário um conjunto pré-definido de objetos de interface, tais como *canvas*, *scrollbars*, *menus* e *control panels.*

Após a geração do código fonte da interface, e dispondo-se também do código fonte do aplicativo, utiliza-se um *linker* para gerar o programa executável que pode ser utilizado pelo usuário final.

6.1.1 O Protótipo do *Browser-B*

A partir do modelo da interação do *Browser-B*, descrito no Capítulo 4 foi construído o protótipo correspondente utilizando o *DevGuide.* O protótipo do *Browser-B* é apresentado nas Figuras 6-2 à 6-10.

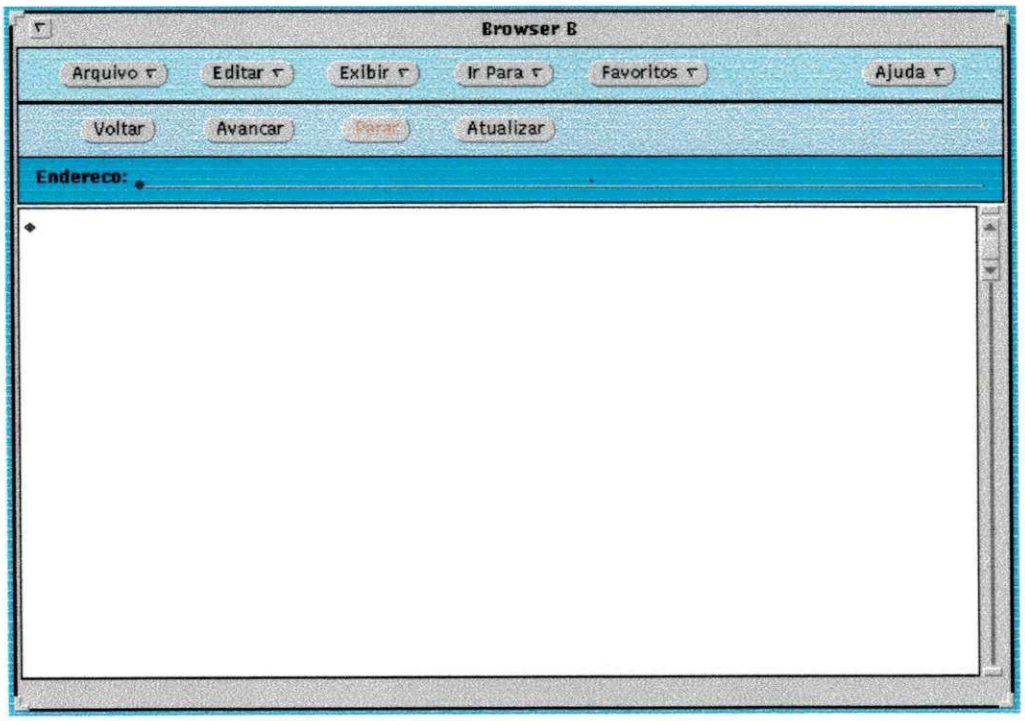

Figura 6-2: winBW - Janela Principal do Navegador

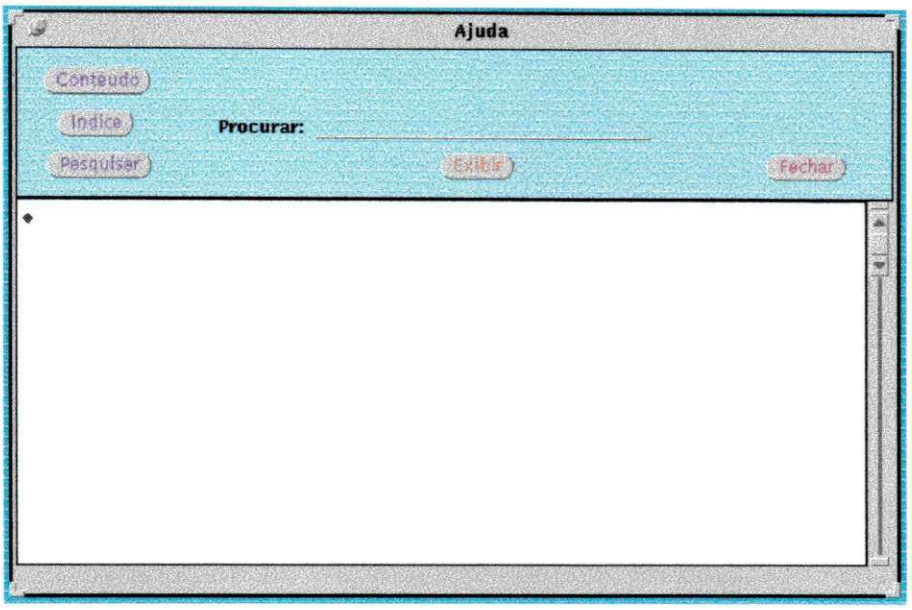

Figura 6-3: winAJ - Janela de Ajuda

Figura 6-4: winAP - Janela Abrir Página

Figura 6-5: winSP - Janela Salvar Página

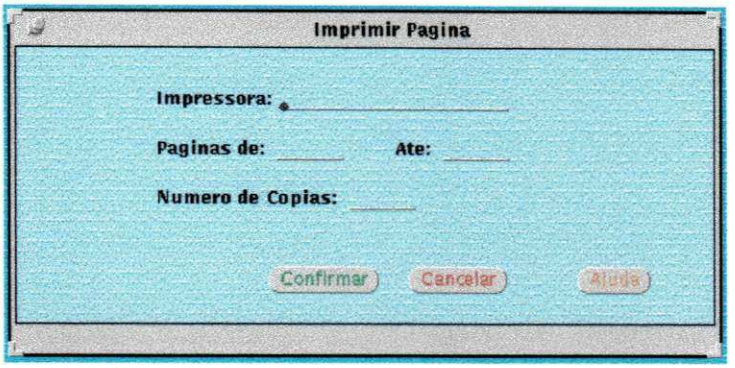

Figura 6-6: winIP - Janela Imprimir Página

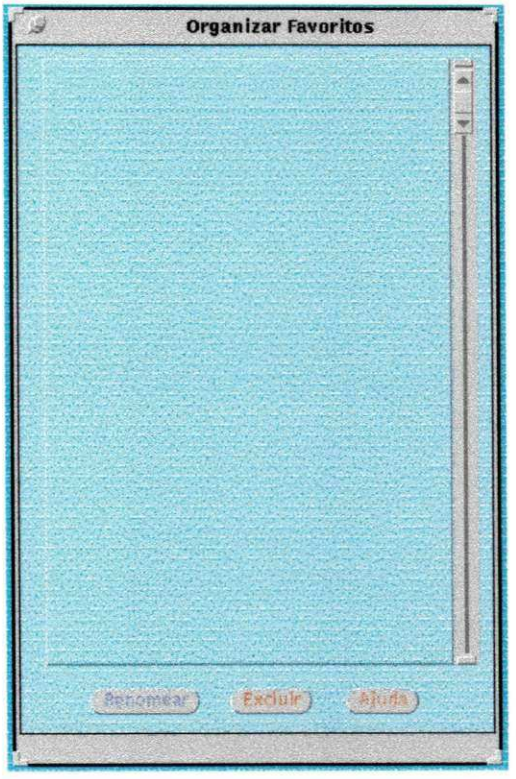

Figura 6-7: winOF - Janela Organizar Favoritos

Figura 6-8: winRF - Janela Renomear Favoritos

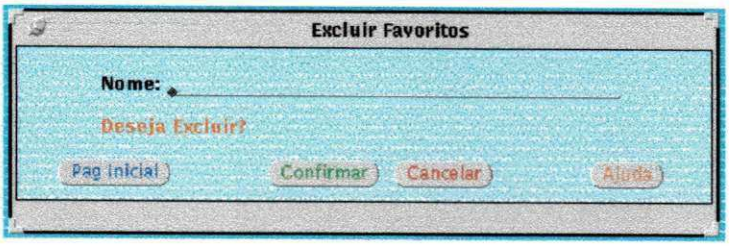

Figura 6-9: winEF - Janela Excluir Favoritos

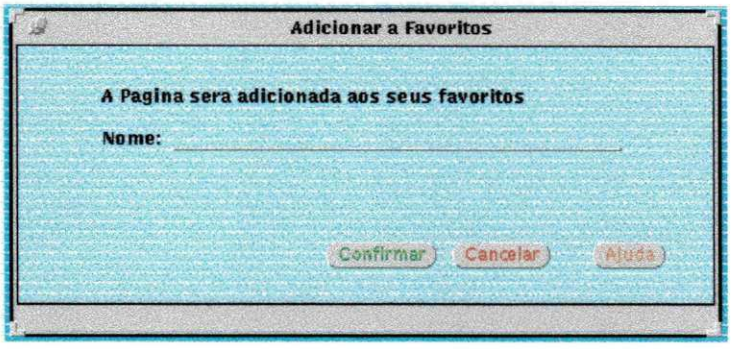

Figura 6-10: winAF - Janela Adicionar Favoritos

Após a construção do protótipo de uma interface, é utilizada a ferramenta FAIUnix para apoiar a sua validação conforme será descrito nas próximas seções.

6.2 A FERRAMENTA FAIUNIX

A ferramenta apoia a avaliação de interfaces projetadas para ambiente UNIX [ST 95][ST 96a]. O ambiente de construção da ferramenta foi o *OpenWindows* [Sun 91].

A ferramenta FAIUnix desenvolvida neste trabalho é composta de dois módulos de coleta de dados: um coletor de dados estático e um coletor de dados dinâmico, discutidos nas seções 6.2.1 e 6.2.2 respectivamente.

Inicialmente na ferramenta, para a utilização dos coletores, é necessário informar qual o arquivo a ser utilizado, dependendo da funcionalidade desejada. A ferramenta trabalha com arquivos de interfaces, que contém a identificação dos arquivos de protótipo que constituem a interface em estudo, com arquivos de protótipo com o objetivo de obter uma nova definição

da interface com a utilização de procedimentos de captura dos eventos, e com arquivo de resultados para análise e visualização ordenada do que foi coletado.

Após a definição do arquivo de trabalho pode-se realizar a visualização deste arquivo ou selecionar com qual dos coletores se deseja trabalhar, conforme Figura 6-11.

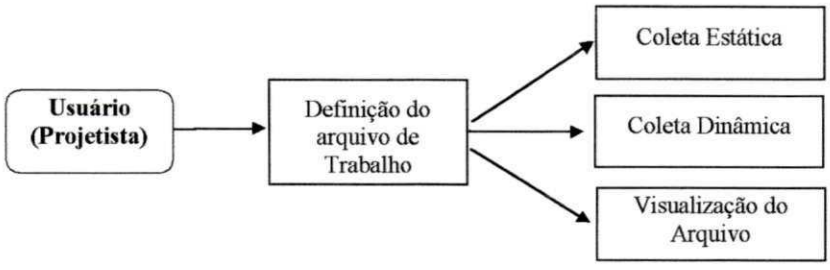

Figura 6-11: Recursos da ferramenta FAIUnix

A janela inicial da FAIUnix é mostrada na Figura 6-12. Na Figura 6-13 é mostrada a visualização do arquivo e na Figura 6-14 é mostrado o processo de seleção do módulo coletor.

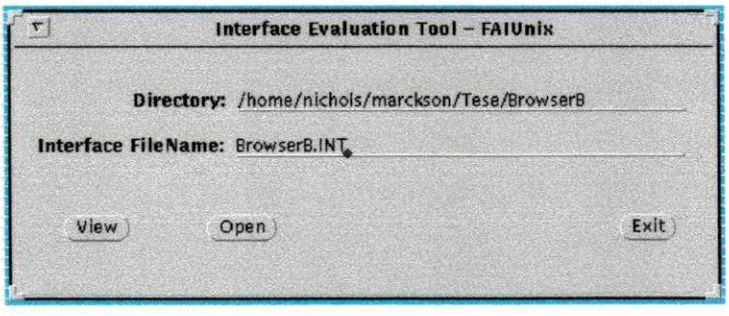

Figura 6-12: Janela Inicial da ferramenta FAIUnix

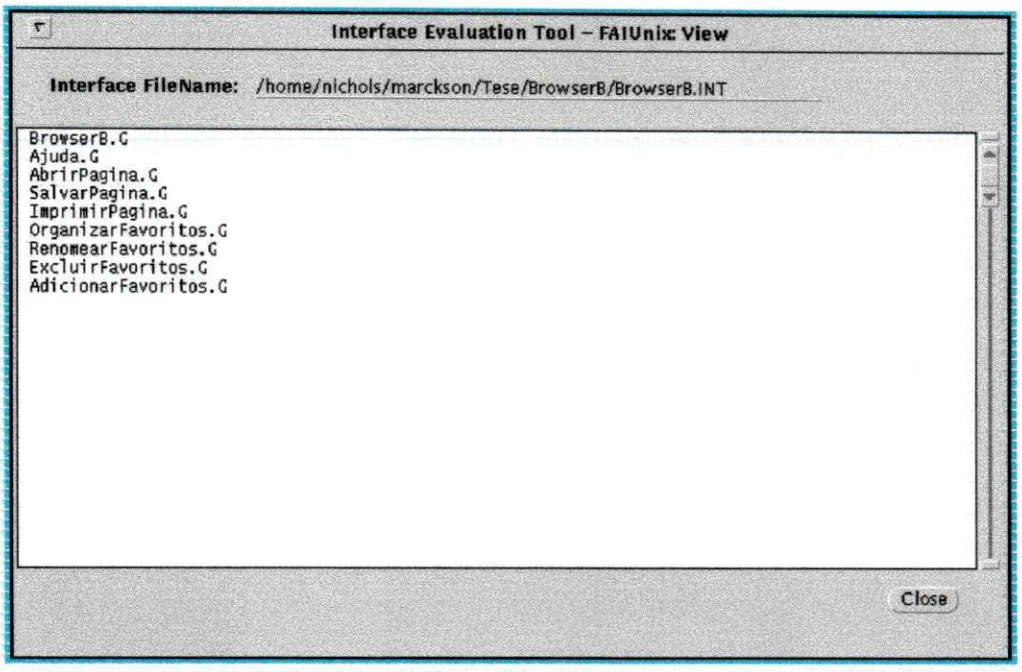

Figura 6-13: Janela de Visualização do arquivo

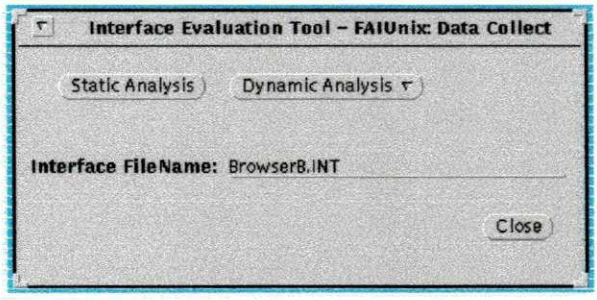

Figura 6-14: Seleção do módulo Coletor

6.2.1 Coletor de Dados Estáticos

O módulo coletor de dados estáticos realiza a coleta de dados sobre um protótipo construído no *DevGuide*. Esta coleta é realizada sem a participação do usuário e permite ao projetista observar questões estruturais de especificação da interface. A partir dele é possível obter informações sobre a distribuição espacial dos objetos na tela, com destaque especial para menus e botões e a utilização de cores no projeto.

Este módulo coletor, não é um módulo dedicado ao *DevGuide*, podendo coletar dados de qualquer arquivo construído de acordo com a sintaxe GIL. O *DevGuide* foi utilizado devido à sua disponibilidade na plataforma de desenvolvimento.

O processo de coleta deve ser realizado de acordo com as etapas descritas na Figura 6-15:

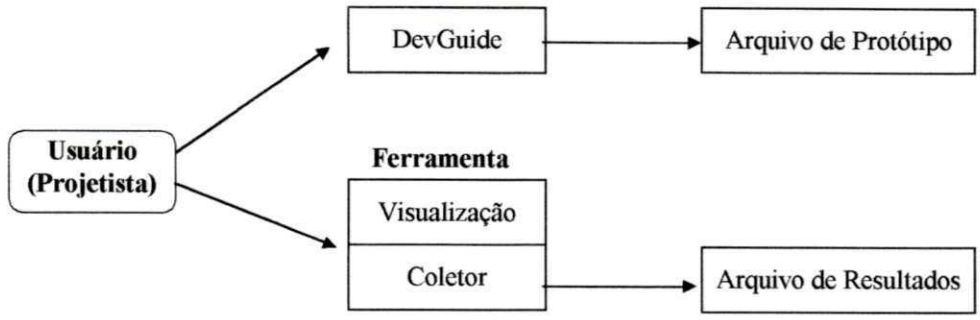

Figura 6-15: Etapas da coleta de dados estáticos

Inicialmente o projetista deve utilizar o *DevGuide* para gerar o protótipo da interface. Uma vez obtido o arquivo de protótipo, contendo todas as definições, o projetista deve utilizar o coletor para registrar as informações de interesse contidas no arquivo de protótipo. Uma vez ativado o coletor, é gerado um arquivo de resultados contendo um relatório sobre a interface sob avaliação. Os resultados da coleta podem então ser visualizados na própria ferramenta, sem a necessidade de software auxiliar, conforme é mostrado na Figura 6-16.

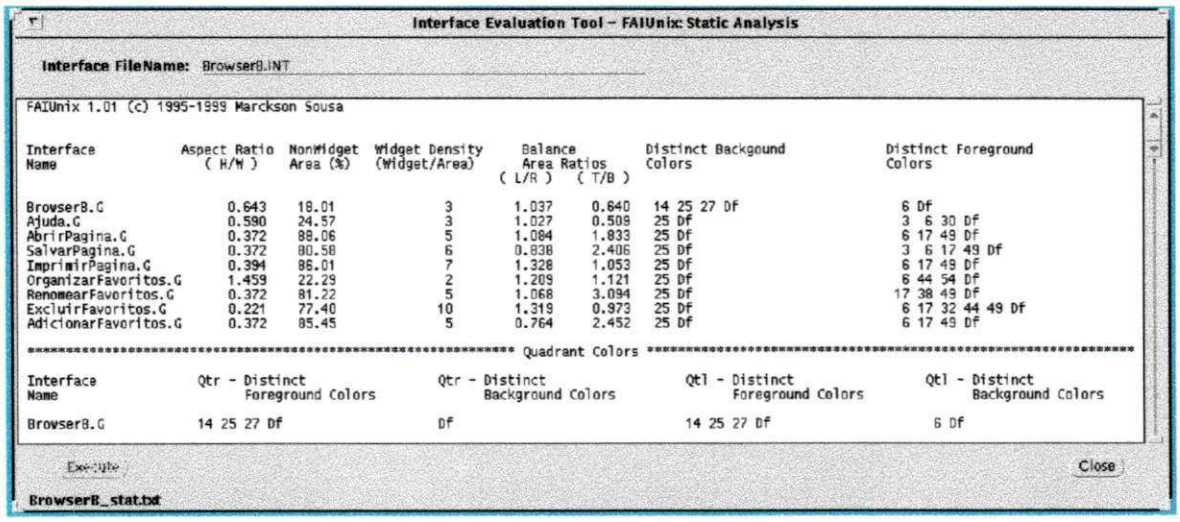

Figura 6-16: Visualização do arquivo de resultados da Coleta Estática

Avaliação do Projeto Visual

Os parâmetros de interesse na coleta estática estão relacionados à distribuição da informação e cores nas janelas do protótipo, além da definição dos menus e botões. Para os propósitos do coletor, do ponto de vista de área ocupada e cor de representação, considerou-se que os elementos de uma interface correspondem a janelas. Isto é, uma janela compreende um espaço na tela, o qual pode ser usado para entrada e saída de informações.

Para avaliar a distribuição das cores apresentadas ao usuário, analisa-se a área de cada janela a partir da análise de seus quadrantes (análogo aos quadrantes matemáticos). Assim é possível avaliar o projeto a partir da comparação dos resultados da ferramenta com recomendações sobre o projeto visual [Shn 98], averiguando a sua adequação.

A seguir, na Tabela 6-1, são listados os itens coletados por este módulo.

Tabela 6-1: Itens obtidos através da coleta estática (Distribuição da Informação)

Distribuição da informação nas Janelas
Identificação da interface
Proporções da Janela (*Aspect Ratio*).
Percentual de ocupação da Janela (*NonWidget*).
Densidade dos Objetos na Janela(*Widget Density*).
Distribuição dos Objetos na Janela (*Balance Area Ratios*).
Utilização de Cores
Distribuição da informação nos Quadrantes
Identificação da Interface
Cores distintas presentes nos quadrantes (*foreground* e *background*)

As métricas adotadas neste trabalho foram um subconjunto das métricas propostas por Mahajan e Schneiderman em [MS 97].

Proporções da Janela

Proporções da Janela (*Aspect Ratio*) – razão entre altura e a largura de uma janela. Os valores desejáveis para esta métrica, de acordo com Schneiderman, devem estar na faixa de 0.5 até 0.8. Janelas com funcionalidades semelhantes devem apresentar a mesma relação.

$$P = \frac{Altura}{Largura} \qquad (6.1)$$

Percentual de Ocupação da Janela

Percentual de Ocupação da Janela (*Nonwidget Area*) - razão entre a área não ocupada e a área total da janela, expressa como uma porcentagem. Valores próximos de 100 indicam uma ocupação elevada e valores < 30 podem sinalizar a necessidade de reprojeto da janela.

$$PO = \frac{Área\_desocupada}{Área\_totaldajanela} \qquad (6.2)$$

Densidade dos Objetos na Janela

Densidade dos Objetos na Janela (*Widget Density*) – número de objetos no nível mais alto da interação, dividido pela área total da janela (multiplicado por 100.000 para efeito de normalização). Este número mede o grau de ocupação da janela. Valores acima de 100 significam que um número relativamente grande de objetos se concentra em uma pequena área da janela.

$$DO = \frac{Númerodeobjetos}{Área\_totaldajanela} 100.000 \qquad (6.3)$$

Distribuição dos Objetos na Janela

Distribuição dos Objetos na Janela (*Balance Area Ratios*) – Medida do equilíbrio na distribuição de objetos na janela. Esta métrica é representada por duas medidas:

Equilíbrio horizontal (T/B) – razão entre as áreas ocupadas na metade superior da tela e na metade inferior.

$$Eh = \frac{Área\_ocupadaSup}{Área\_ocupadaInf} \qquad (6.4a)$$

Equilíbrio vertical (L/R) – razão entre as áreas ocupadas na metade esquerda da tela e na metade direita.

$$Ev = \frac{Área\_ocupadaEsq}{Área\_ocupadaDir} \qquad (6.4b)$$

Valores elevados entre 4.0 e 10.0 indicam que a distribuição de objetos na tela não está bem equilibrada. O limite superior 10, representa uma tela em branco ou quase em branco (ex. uma tela com apenas um objeto de dimensões irrelevantes face a dimensão da tela).

Utilização de Cores

Utilização de Cores – o propósito desta métrica é avaliar a consistência no uso de cores ao longo do projeto. Além de permitir a análise do número de cores na janela, possibilita avaliar a combinação utilizada. De forma a facilitar a análise, para as cores disponíveis na ferramenta de construção de protótipo *DevGuide*, foi associado um número inteiro para cada cor.

Esta métrica é representada por duas listas que relacionam todas as cores utilizadas na janela:

Relação das Cores de Fundo (*Distinct Background Color*) – lista de elementos que representam as cores de fundo utilizada na janela

Relação das Cores de Frente (*Distinct Foreground Color*) – lista de elementos que representam as cores de frente utilizadas na janela.

Avaliação da Estrutura do Diálogo

Além das informações sobre a distribuição da informação, apresentada na Tabela 6-1, é também possível avaliar a estrutura de menus e botões. Considerando o número de objetos envolvidos no projeto da interação, uma avaliação mais cuidadosa da estrutura de elementos como menus e botões torna-se sujeita a erros quando feita sem o auxílio de uma ferramenta. Portanto, foi incluído na FAIUnix um recurso para relacionar todos os objetos destas

categorias e suas características, de modo a facilitar a análise. As métricas utilizadas nesta avaliação são diretrizes de projeto tais como as diretrizes de Smith e Moiser [SM 86]. As informações sobre a estrutura do diálogo são mostradas na Tabela 6-2.

Além da análise do uso de cores, os aspectos que podem ser analisados são aqueles tratados na análise do modelo da interação, porém agora eles passam a ser analisados no contexto do protótipo. Assim, é possível verificar se o protótipo corresponde ao modelo da interação proposto originalmente. As características visuais destes objetos já foram tratadas na seção anterior.

Tabela 6-2: Itens obtidos através da coleta estática (Avaliação da Estrutura do Diálogo)

Estrutura dos Menus
Identificação da Interface
Identificação do menu
Número de colunas do menu
Número de itens do menu
Se existe um submenu associado
Estrutura dos Itens de menu
Identificação da Interface
Identificação do menu
Identificação da opção de menu
Cor de frente (*foreground*)
Identificação do menu ativado por esta opção
Estrutura dos Botões
Identificação da Interface
Identificação do botão
Cor de frente (*foreground*)
Identificação do menu associado ao botão

6.2.2 Coletor de Dados Dinâmicos

O coletor dinâmico realiza a coleta de informações oriundas da interação do usuário com um protótipo da interface, possibilitando uma verificação de aspectos comportamentais do projeto da interação. O módulo de Coleta de Dados Dinâmicos permite gravar informações com base na técnica de *Software Logging*, a partir da ocorrência de eventos do teclado e do mouse considerando o instante de tempo e o objeto que provocou a ocorrência do evento.

O processo de coleta deve ser realizado de acordo com as etapas mostradas na Figura 6-17 e discutidas a seguir:

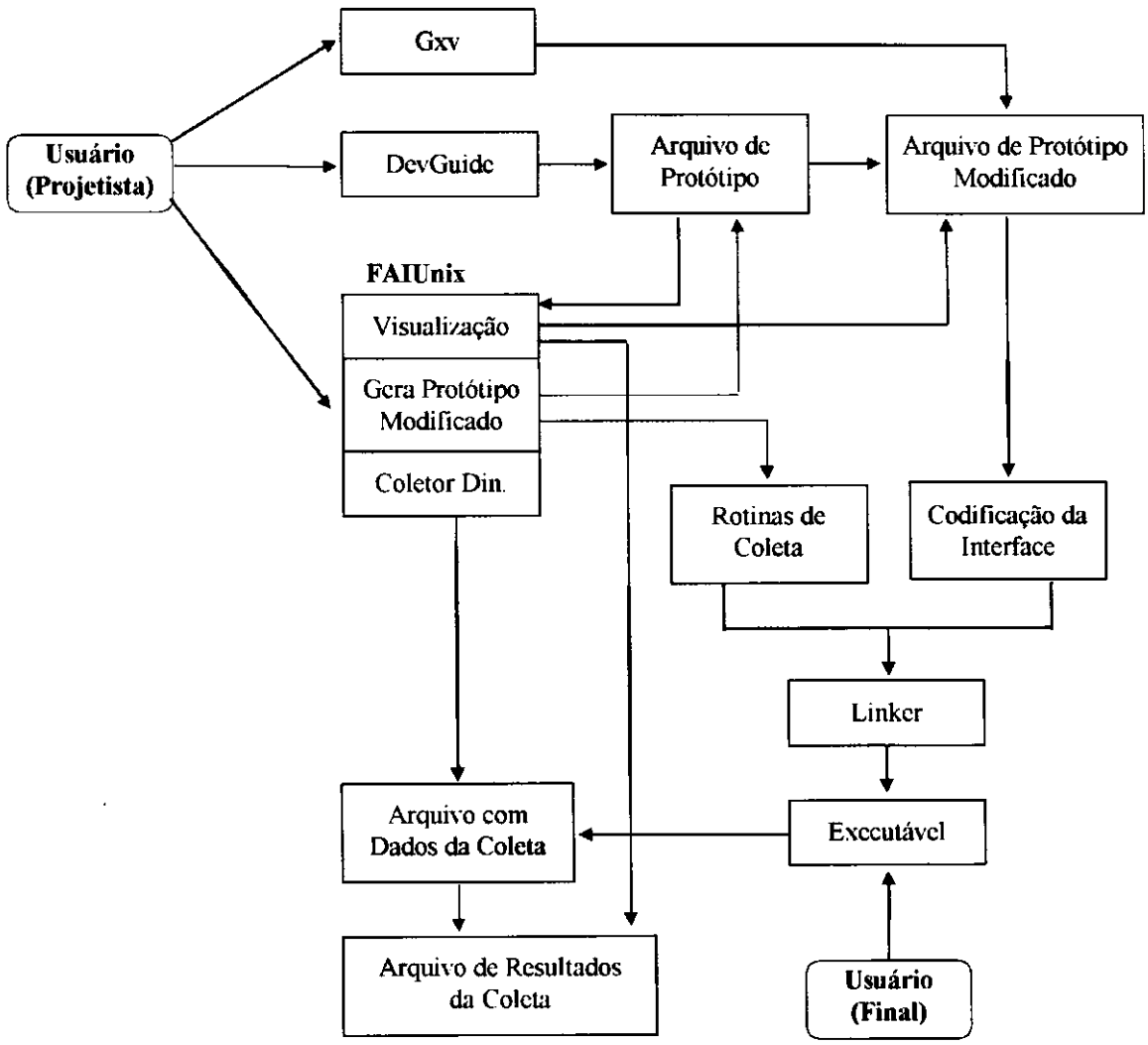

Figura 6-17: Etapas da coleta de dados dinâmicos

6.2.3 Preparação da Coleta

Inicialmente é gerado o arquivo de protótipo da interface utilizando o *DevGuide*. Em seguida é utilizado o coletor para inserir apontadores para as rotinas de captura automática dos eventos no arquivo de protótipo, neste passo é gerado automaticamente todo o código auxiliar necessário.

A próxima etapa consiste em utilizar o software Gxv para gerar o código fonte da interface. Este código deve então ser ligado (*linked)* ao código que contém as funções de coleta. Após a "linkagem", é gerado um programa executável o qual pode ser utilizado para a captura dos dados quando da utilização do protótipo pelo usuário final.

Na Figura 6-18 é mostrada a janela onde pode ser gerado o novo arquivo de protótipo através da seleção do botão "*New Prototype*", a restauração do arquivo anterior de protótipo (*Undo*) através da seleção do botão "*Restore Old Prototype*".

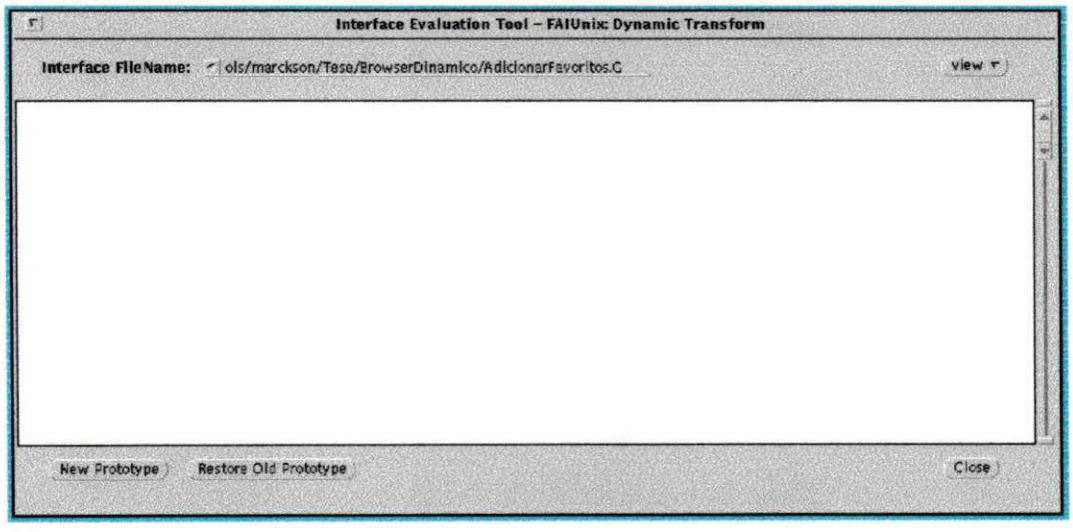

Figura 6-18: Janela de Geração do Protótipo Modificado

Nesta janela da ferramenta é possível visualizar os arquivos de protótipo original e modificado, o programa auxiliar gerado e os resultados da coleta.

6.2.4 Coleta de Dados

Uma vez definidas as tarefas que serão realizadas pelos usuários para validação do protótipo, nesta etapa são realizadas as sessões de teste com usuários para captura dos dados. Antes de iniciar a sessão de testes o módulo coletor é ativado, e são solicitados: a identificação do protótipo e do arquivo de coleta, o nome do projetista, do usuário, da sessão de utilização e do tempo limite para realização da coleta de dados, para que seja gerado um arquivo de resultados. Na Figura 6-19 é mostrada a entrada de informações necessárias ao início de uma sessão de coleta.

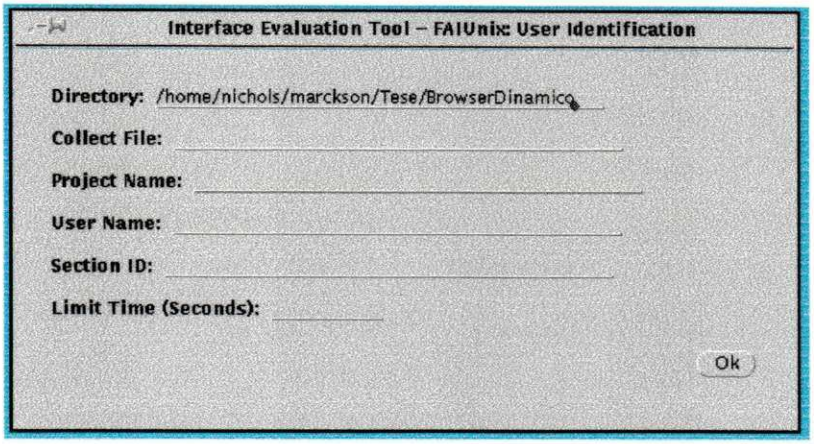

Figura 6-19: Janela de Identificação para o início da sessão de Teste

A ferramenta FAIUnix, oferece suporte ao projetista para avaliar a extensão com que os objetivos da tarefa foram atingidos e o tempo gasto para atingi-los. Tendo definido os parâmetros de interesse para a avaliação, a ferramenta FAIUnix foi desenvolvida para coletar as informações listadas na Tabela 6-3.

Tabela 6-3: Itens obtidos através do coletor dinâmico

Informações sobre as Janelas
Identificação da interface
Identificação do objeto
Número de vezes em que o objeto foi acessado
Informações sobre a sessão
Número de solicitações de ajuda
Cenário percorrido na realização de uma tarefa
Tempo utilizado na consulta à ajuda
Tempo de realização de uma tarefa

6.2.5 Apresentação dos resultados

Finalizada uma sessão de interação os dados coletados pela ferramenta podem ser apresentados na forma de um arquivo de *log* ou na forma de um relatório[9]. Um arquivo de coleta estará disponível para consulta e análise e pode ser visualizado como é mostrado na Figura 6-20.

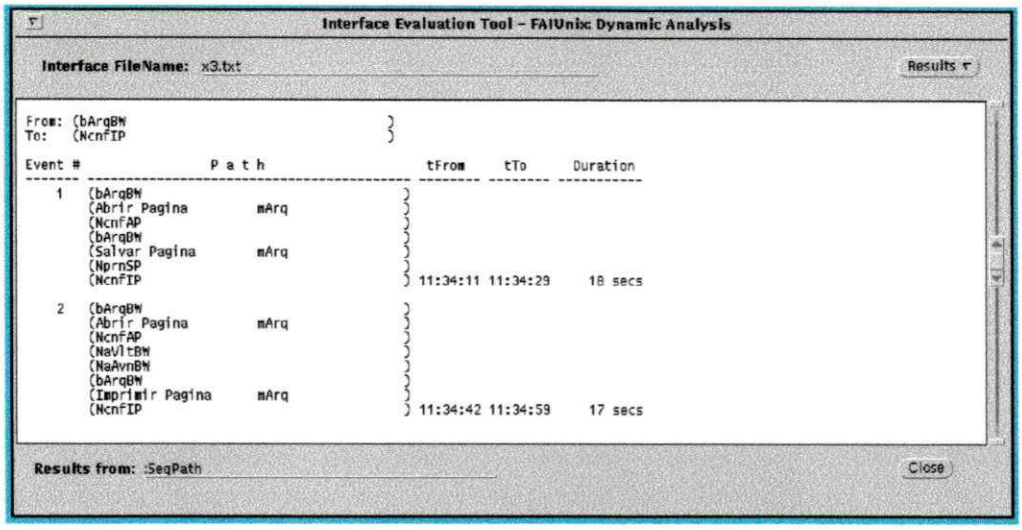

Figura 6-20: Janela com Resultados da Coleta Dinâmica

As opções de visualização dos dados permitem selecionar no arquivo de resultados as informações de interesse tais como informações sobre botões, menus ou todas as informações coletadas. O mecanismo de visualização dos resultados, também possibilita extrair do arquivo de *log* todos os cenários percorridos pelo usuário ao selecionar objetos da interface que tenham sido monitorados. Selecionando-se um par de objetos que correspondam ao início e fim de um trecho da interação, a FAIUnix apresenta todos os trechos do arquivo de *log* nos quais estes objetos foram registrados. Os trechos de interação podem corresponder à realização de uma tarefa ou mesmo de subtarefas. Para cada trecho de interação são apresentados quais objetos foram ativados, o registro do tempo de ativação, o tempo de início e fim da interação no trecho, assim como a sua duração.

Embora não seja explorado no caso do navegador, a ferramenta FAIUnix pode monitorar teclas específicas, onde é possível analisar o registro de situações tais como solicitação de ajuda, *undo* (já monitorados a partir dos botões e menus).

---

[9] Os resultados serão discutidos mais adiante pois contêm os dados da coleta relativa ao estudo de caso.

6.2.6 Análise dos Dados

A partir destes dados, o projetista é capaz de analisar quantas e quais foram as tarefas concluídas, a ordem de execução das tarefas, a seqüência utilizada na execução da tarefa, quantos erros foram cometidos, em quais situações e quantas vezes foi necessária a ajuda, como o usuário utilizou o tempo durante a realização da tarefa, o padrão de utilização de recursos.

Número de tarefas completadas com êxito

Se este número for elevado pode-se interpretar que a interface está bem projetada ou apresenta tarefas fáceis de realizar. Por outro lado, se este número for baixo pode significar que as tarefas são complexas ou que os mecanismos de ajuda são inadequados.

Freqüência de realização das tarefas

Se a freqüência for alta, a realização da tarefa deve ser apoiada adequadamente no projeto da interface, por exemplo disponibilizando atalhos e recursos para edição de macros.

Utilização do tempo durante a realização da tarefa

O tempo utilizado na realização de uma tarefa pode ser utilizado para medir o grau de dificuldade na sua realização. É possível também analisar o tempo gasto de forma improdutiva na tentativa de resolver problemas ou consultando a ajuda.

Padrão de utilização de recursos

O registro da freqüência de utilização dos recursos durante a interação pode ser usado para avaliar o conhecimento que o usuário tem dos recursos disponíveis. Por exemplo, o conhecimento da existência de caminhos alternativos para realização de uma mesma tarefa.

Além de identificar os recursos mais freqüentemente utilizados é possível também identificar os recursos jamais utilizados, ou por desconhecimento do usuário ou por serem desnecessários à realização da tarefa.

Ordem de execução da tarefa

O registro de *log* pode indicar uma ordem de realização das tarefas diferente da que foi projetada. O conhecimento do padrão de sequenciamento pode levar a interface a antecipar as próximas ações do usuário levando a ativações automáticas durante a interação.

Seqüência das ações nas tarefas

A partir da análise da seqüência de ações é possível avaliar a eficiência dos caminhos escolhidos pelo usuário, comparando o caminho seguido com os caminhos projetados.

Erros cometidos durante a interação

Analisando o número de erros, o tipo de erro e as situações nas quais ocorrem durante a interação, é possível ao projetista buscar soluções para melhorar a interface. Alguns erros, dada sua gravidade, podem forçar o usuário a reiniciar a tarefa, fato que também pode ser contabilizado na análise.

Padrão de solicitação de Ajuda

O número de vezes que foi solicitada a ajuda e o tempo utilizado na consulta podem medir a facilidade de uso da interface e a eficácia dos mecanismos de ajuda. O registro do tempo e do contexto (em quais situações) de solicitação da ajuda levam a localização dos problemas da interface.

A partir desta análise é possível localizar problemas de usabilidade e antecipar quais as mudanças necessárias para resolvê-los. Por exemplo, é possível analisar a evolução do aprendizado do usuário a partir da redução no número de erros, redução no tempo de execução das tarefas, redução na solicitação de ajuda, ao longo de uma mesma sessão de teste ou ainda comparando o *log* de sessões realizadas ao longo do tempo.

A seção a seguir mostra como a ferramenta FAIUnix foi utilizada na avaliação do protótipo do *Browser-B*.

6.3 VALIDAÇÃO DO PROTÓTIPO DO *BROWSER*-B

Esta seção apresenta os resultados da coleta de dados relativa ao protótipo do *Browser-B*.

6.3.1 Resultados da coleta Estática

Após a construção do protótipo da interface, foi utilizado o módulo de coleta de dados estáticos para obtenção de resultados referentes à apresentação da informação nas janelas e a estrutura de menus e botões.

O coletor de dados estáticos gera um arquivo de resultados com as informações descritas nas Tabelas 6-1 e 6-2, a partir do arquivo de protótipo. Nas Tabelas 6-4, 6-5 e 6-6 são apresentados o arquivo de protótipo para a janela principal do navegador ("winBW").

Tabela 6-4: Arquivo de Protótipo da Janela winBW (elemento menu mArq)

```
001  ;GIL-2
002  (
003  (
004       :type                    :menu
005       :name                    mArq
006       :help                    ""
007       :columns                 1
008       :label                   ""
009       :label-type              :string
010       :menu-type               :command
011       :menu-handler            nil
012       :menu-title              ""
013       :menu-item-labels        ("Abrir Pagina" "Salvar Pagina" "Imprimir Pagina" "Sair" )
014       :menu-item-label-types   (:string :string :string :string )
015       :menu-item-defaults      (nil nil nil nil )
016       :menu-item-handlers      (Ho_abrpag Ho_savpag Ho_imppag Ho_sair )
017       :menu-item-menus         (nil nil nil nil )
018       :menu-item-colors        ("" "" "" "Red" )
019       :pinnable                t
020       :user-data               ()
021  )
```

Na Tabela 6-4, a janela winBW está definida de acordo com a sintaxe GIL. Um arquivo GIL consiste em uma lista de elementos que contém a descrição de uma interface. Cada elemento na lista inclui uma descrição da forma como o mesmo está definido no *Devguide* incluindo: tamanho, posição, relacionamento com outros elementos e características individuais. A descrição de um elemento está envolvida por um par de parênteses, e a lista completa dos elementos também está envolvida por um outro par de parênteses. Um arquivo

GIL pode conter comentários, que devem começar necessariamente com um ponto e virgula (;). Todos os comentários devem estar posicionados no começo do arquivo e antes da abertura do primeiro par de parênteses que inicia um arquivo GIL.

Tabela 6-5: Arquivo de Protótipo da Janela winBW (elemento janela principal)

```
136  (
137      :type                  :base-window
138      :name                  winBW
139      :owner                 nil
140      :width                 700
141      :height                450
142      :background-color      ""
143      :foreground-color      ""
144      :label                 "Browser B"
145      :label-type            :string
146      :mapped                t
147      :show-footer           t
148      :resizable             t
149      :icon-file             ""
150      :icon-mask-file        ""
151      :event-handler         nil
152      :events                ()
153      :user-data             ()
154  )
```

Tabela 6-6: Arquivo de Protótipo da Janela winBW (elemento botão bArqBW)

```
172  (
173      :type                  :button
174      :name                  bArqBW
175      :owner                 ctrlBW
176      :help                  ""
177      :x                     30
178      :y                     10
179      :constant-width        nil
180      :button-type           :normal
181      :width                 78
182      :height                19
183      :foreground-color      ""
184      :label                 "Arquivo"
185      :label-type            :string
186      :menu                  mArq
187      :notify-handler        nil
188      :event-handler         nil
189      :events                ()
190      :user-data             ()
191  )
```

Quando o *Devguide* gera um arquivo GIL, ele inclui como primeiro passo uma linha de comentário que mostra a versão utilizada da linguagem GIL. A segunda linha é o parêntese esquerdo que abre a lista de elementos e a terceira linha é o parêntese esquerdo que abre a definição do primeiro elemento na lista. Segue então a definição de cada elemento individualmente, cada definição separada por um fechamento e uma conseqüente abertura de

parênteses, entre as definições. Para finalizar o arquivo existe um parêntese direito que fecha a lista de elementos.

O arquivo de resultado da coleta de dados relativa à interface está representado nas Tabelas 6-7 à 6-10. Nestas Tabelas convencionou-se que cada linha representa uma janela do projeto da interface e cada coluna representa uma métrica utilizada. Tipicamente o avaliador analisaria cada coluna se detendo nos valores extremos.

Tabela 6-7: Arquivo de resultados da coleta estática

```
FAIUnix 1.01 (c) 1995-1999 Marckson Sousa


Interface           Aspect Ratio  NonWidget  Widget Density     Balance          Distinct Backgound            Distinct Foreground
Name                   ( H/W )    Area (%)   (Widget/Area)     Area Ratios       Colors                        Colors
                                                              ( L/R )   ( T/B )

BrowserB.G              0.643      18.01          3           1.037     0.640   14 25 27 Df                    6 Df
Ajuda.G                 0.590      24.57          3           1.027     0.509   25 Df                          3  6 30 Df
AbrirPagina.G           0.372      88.06          5           1.084     1.833   25 Df                          6 17 49 Df
SalvarPagina.G          0.372      80.58          6           0.838     2.406   25 Df                          3  6 17 49 Df
ImprimirPagina.G        0.394      86.01          7           1.328     1.053   25 Df                          6 17 49 Df
OrganizarFavoritos.G    1.459      22.29          2           1.209     1.121   25 Df                          6 44 54 Df
RenomearFavoritos.G     0.372      81.22          5           1.068     3.094   25 Df                         17 38 49 Df
ExcluirFavoritos.G      0.221      77.40         10           1.319     0.973   25 Df                          6 17 32 44 49 Df
AdicionarFavoritos.G    0.372      85.45          5           0.764     2.452   25 Df                          6 17 49 Df

************************************************************ Quadrant Colors ************************************************************

Interface           Qtr - Distinct                Qtr - Distinct                Qtl - Distinct                Qtl - Distinct
Name                      Foreground Colors             Background Colors             Foreground Colors             Background Colors

BrowserB.G          14 25 27 Df                   Df                            14 25 27 Df                    6 Df
Ajuda.G             25 Df                          6 30 Df                      25 Df                          3  6 Df
AbrirPagina.G       25 Df                         Df                            25 Df                         Df
SalvarPagina.G      25 Df                         Df                            25 Df                         Df
ImprimirPagina.G    25 Df                         Df                            25 Df                         Df
OrganizarFavoritos.G 25 Df                        Df                            25 Df                         Df
RenomearFavoritos.G 25 Df                         Df                            25 Df                         Df
ExcluirFavoritos.G  25 Df                         Df                            25 Df                         44 Df
AdicionarFavoritos.G 25 Df                        Df                            25 Df                         Df

                    Qbl - Distinct                Qbl - Distinct                Qbr - Distinct                Qbr - Distinct
                          Foreground Colors             Background Colors             Foreground Colors             Background Colors

BrowserB.G          Df                            Df                            Df                            Df
Ajuda.G             Df                            Df                            Df                            Df
AbrirPagina.G       25 Df                         17 Df                         25 Df                          6 17 49 Df
SalvarPagina.G      25 Df                          3 17 Df                      25 Df                          6 17 49 Df
ImprimirPagina.G    25 Df                         17 Df                         25 Df                          6 17 49 Df
OrganizarFavoritos.G 25 Df                        44 54 Df                      25 Df                          6 44 Df
RenomearFavoritos.G 25 Df                         17 38 Df                      25 Df                         17 49 Df
ExcluirFavoritos.G  25 Df                         17 32 44 Df                   25 Df                          6 17 49 Df
AdicionarFavoritos.G 25 Df                        17 Df                         25 Df                          6 17 49 Df
```

Tabela 6-8: Arquivo de resultados da coleta estática (Continuação)

```
********************************************************** Menus Definition ***************************************************************

Interface Name        Menu Name             Columns Number      Options Number          Menu-item-menu

BrowserB.G            mArq                          1                   4                   FALSE
BrowserB.G            mEdt                          1                   3                   FALSE
BrowserB.G            mExb                          1                   3                   FALSE
BrowserB.G            mIrP                          1                   2                   FALSE
BrowserB.G            mFav                          1                   2                   FALSE
BrowserB.G            mAjd                          1                   3                   TRUE
BrowserB.G            smCnt                         1                   7                   FALSE

                                            Option ID           Foreground Color        Call Menu

BrowserB.G            mArq                  Abrir Pagina              Df
BrowserB.G            mArq                  Salvar Pagina             Df
BrowserB.G            mArq                  Imprimir Pagina           Df
BrowserB.G            mArq                  Sair                      49
BrowserB.G            mEdt                  Recortar                  Df
BrowserB.G            mEdt                  Copiar                    Df
BrowserB.G            mEdt                  Colar                     Df
BrowserB.G            mExb                  Parar                     Df
BrowserB.G            mExb                  Atualizar                 Df
BrowserB.G            mExb                  Ver Codigo Fonte          Df
BrowserB.G            mIrP                  Voltar                    Df
BrowserB.G            mIrP                  Avancar                   Df
BrowserB.G            mFav                  Adicionar                 Df
BrowserB.G            mFav                  Organizar                 Df
BrowserB.G            mAjd                  Conteudo                  Df              smCnt
BrowserB.G            mAjd                  Indice                    Df
BrowserB.G            mAjd                  Pesquisar                 Df
BrowserB.G            smCnt                 Geral                     Df
BrowserB.G            smCnt                 Abrir Pagina              Df
BrowserB.G            smCnt                 Salvar Pagina             Df
BrowserB.G            smCnt                 Imprimir Pagina           Df
BrowserB.G            smCnt                 Adicionar a Favoritos     Df
BrowserB.G            smCnt                 Organizar Favoritos       Df
BrowserB.G            smCnt                 Excluir Favoritos         Df
```

Tabela 6-9: Arquivo de resultados da coleta estática (Continuação)

```
************************************************************ Buttons Definition ************************************************************

Interface Name        Button Name         Foreground Color      Menu

BrowserB.G            bArqBW                    Df            mArq
BrowserB.G            bEdtBW                    Df            mEdt
BrowserB.G            bExbBW                    Df            mExb
BrowserB.G            bIrPBW                    Df            mIrP
BrowserB.G            bFavBW                    Df            mFav
BrowserB.G            bAjdBW                    Df            mAjd
BrowserB.G            aVltBW                    Df
BrowserB.G            aAvnBW                    Df
BrowserB.G            aStpBW                     6
BrowserB.G            aRefBW                    Df
Ajuda.G               cntAJ                      3
Ajuda.G               indAJ                      3
Ajuda.G               psqAJ                      3
Ajuda.G               exbAJ                      6
Ajuda.G               fchAJ                     30
AbrirPagina.G         cnfAP                     17
AbrirPagina.G         canAP                     49
AbrirPagina.G         hlpAP                      6
SalvarPagina.G        prnSP                      3
SalvarPagina.G        cnfSP                     17
SalvarPagina.G        canSP                     49
SalvarPagina.G        hlpSP                      6
ImprimirPagina.G      cnfIP                     17
ImprimirPagina.G      canIP                     49
ImprimirPagina.G      hlpIP                      6
OrganizarFavoritos.G  renOF                     54
OrganizarFavoritos.G  delOF                     44
OrganizarFavoritos.G  hlpOF                      6
RenomearFavoritos.G   desRF                     38
RenomearFavoritos.G   cnfRF                     17
RenomearFavoritos.G   canRF                     49
ExcluirFavoritos.G    gtmEF                     32
ExcluirFavoritos.G    cnfEF                     17
ExcluirFavoritos.G    canEF                     49
ExcluirFavoritos.G    hlpEF                      6
AdicionarFavoritos.G  cnfAF                     17
AdicionarFavoritos.G  canAF                     49
AdicionarFavoritos.G  hlpAF                      6
```

Tabela 6-10: Arquivo de resultados da coleta estática (Continuação)

```
************************************************************ Color Table ************************************************************

  0  Aquamarine              17  Forest Green            34  Medium Sea Green        51  Sea Green
  1  Black                   18  Gold                    35  Medium Slate Blue       52  Sienna
  2  Blue                    19  Goldenrod               36  Medium Spring Green     53  Sky Blue
  3  Blue Violet             20  Gray                    37  Medium Turquoise        54  Slate Blue
  4  Brown                   21  Green                   38  Medium Violet Red       55  Spring Green
  5  Cadet Blue              22  Green Yellow            39  Midnight Blue           56  Steel Blue
  6  Coral                   23  Indian Red              40  Navy                    57  Tan
  7  Cornflower Blue         24  Khaki                   41  Navy Blue               58  Thistle
  8  Cyan                    25  Light Blue              42  Olive Drab              59  Turquoise
  9  Dark Green              26  Light Gray              43  Orange                  60  Violet
 10  Dark Olive Green        27  Light Steel Blue        44  Orange Red              61  Violet Red
 11  Dark Orchid             28  Lime Green              45  Orchid                  62  Wheat
 12  Dark Slate Blue         29  Magenta                 46  Pale Green              63  White
 13  Dark Slate Gray         30  Maroon                  47  Pink                    64  Yellow
 14  Dark Turquoise          31  Medium Aquamarine       48  Plum                    65  Yellow Green
 15  Dim Gray                32  Medium Blue             49  Red                     Df  Default Color
 16  Firebrick               33  Medium Orchid           50  Salmon
```

Com o objetivo de esclarecer a apresentação destes resultados, está descrito abaixo no texto alguns trechos deste arquivo.

1 - Distribuição de objetos na tela visando averiguar a densidade, ou concentração, de objetos nos quadrantes da janela:

Interface Name	Aspect Ratio (H/W)	NonWidget Area (%)	Widget Density (Widget/Area)	Balance Area Ratios (L/R)	(T/B)
OrganizarFavoritos.G	1.459	22.90	1	1.216	1.140
ExcluirFavoritos.G	0.221	77.40	10	1.319	0.973

Pela análise dos resultados acima, observa-se que OrganizarFavoritos apresenta um "*Aspect Ratio*" de 1.459 enquanto ExcluirFavoritos apresenta um valor de 0.221, concluindo-se que OrganizarFavoritos possui uma altura bastante superior ao seu comprimento, enquanto ExcluirFavoritos é justamente o oposto, um comprimento bastante superior a sua altura, conforme pode ser observado nas Figuras 6-7 e 6-9 respectivamente. De acordo com este diagnóstico, ambas as janelas necessitam ser reprojetadas para ficar com este parâmetro na faixa deseja (0.5 até 0.8).

2 - Distribuição de cores nas janelas :

Uma outra informação fornecida é a relação e o número de cores presentes em cada protótipo da interface. O relatório do módulo de coleta fornece a identificação das cores de *Background* e *Foreground* existentes na janela como um todo e em cada um de seus quadrantes individuais.

Interface Name	Distinct Backgound Colors	Distinct Foreground Colors
OrganizarFavoritos.G	25 Df	44 54 Df
ExcluirFavoritos.G	25 Df	6 17 32 44 49 Df

3 – Estrutura dos menus:

É possível averiguar o número de opções disponíveis para os menus, como também se cada opção possui um submenu associado. Para cada menu pode-se verificar quais as opções existentes, além da cor de cada item e do submenu que será ativado.

```
************************************ Menus Definition ************************************

Interface Name      Menu Name     Columns Number      Options Number     Menu-item-menu

BrowserB.G          mArq                1                   4              FALSE

                                    Option ID          Foreground Color      Call Menu

BrowserB.G            mArq        Abrir Pagina              Df
BrowserB.G            mArq        Salvar Pagina             Df
BrowserB.G            mArq        Imprimir Pagina           Df
BrowserB.G            mArq        Sair                      49
```

4 – Estrutura dos botões:

É possível averiguar a cor associada ao texto do botão, além de identificar se existe um menu associado.

```
********************************** Buttons Definition **********************************

Interface Name        Button Name          Foreground Color      Menu

BrowserB.G            bArqBW                     Df              mArq
BrowserB.G            bEdtBW                     Df              mEdt
BrowserB.G            aVltBW                     Df
BrowserB.G            aAvnBW                     Df
```

6.3.2 Resultados da coleta Dinâmica

Esta seção apresenta os resultados da coleta de dados dinâmicos relativos ao protótipo do *Browser-B*. Para realização da coleta e verificação do modelo, foi seguido o cenário apresentado na Tabela 5-9, no Capítulo 5.

Na Tabela 6-11, é apresentada a definição do objeto **mArq,** do tipo menu. Para possibilitar o processo de coleta, o módulo coletor dinâmico altera o "*menu-handler*" o qual vai ativar uma função de coleta sempre que ações associadas ao menu forem ativadas. As modificações introduzidas pelo módulo de coleta, acrescentando chamadas às rotinas de coleta de dados é mostrada na linha 009.

Tabela 6-11: Arquivo de Protótipo com chamadas às Rotinas de Coleta Dinâmica

```
001  (
002       :type                    :menu
003       :name                    mArq
004       :help                    ""
005       :columns                 1
006       :label                   ""
007       :label-type              :string
008       :menu-type               :command
009       :menu-handler            mh_mArq
010       :menu-title              ""
011       :menu-item-labels        ("Abrir Pagina" "Salvar Pagina" "Imprimir Pagina" "Sair" )
012       :menu-item-label-types   (:string :string :string :string )
013       :menu-item-defaults      (nil nil nil nil )
014       :menu-item-handlers      (Ho_abrpag Ho_savpag Ho_imppag Ho_sair )
015       :menu-item-menus         (nil nil nil nil )
016       :menu-item-colors        ("" "" "" "Red" )
017       :pinnable                t
018       :user-data               ()
019  )
```

Tabela 6-12: Arquivo com Rotinas que identificam os Eventos de Mouse e Teclado

```
001  /*
002   * /marckson/Tese/BrowserDinamico/BrowserB_dyn.c - FAIUnix User interface functions.
003   * This file was generated by `FAIUnix' from `BrowserB.G'.
004   * DO NOT EDIT BY HAND.
005   */
006
007  #include <stdio.h>
008  #include <time.h>
009  #include <sys/param.h>
010  #include <sys/types.h>
011  #include <xview/xview.h>
012  #include <xview/panel.h>
013  #include <xview/textsw.h>
014  #include <xview/xv_xrect.h>
015  #include <gdd.h>
016  #include "ev_say.h"
017  #include "ev_say.c"
018  #include "user_id_ui.c"
019
020  /*
021   * Dynamic Menu: Menu handler for `mArq'.
022   */
023  Menu
024  mh_mArq(menu, op)
025    Menu            menu;
026    Menu_generate   op;
027  {
028    if (collectON == 0) return menu;
029
030    if (op == MENU_NOTIFY) {
031      t = time(NULL);
032      tgm = localtime(&t);
033
034      fprintf(stddyn, "[BrowserB.G      ][:menu      ] %02d:%02d:%02d %-20s mArq      \n",
035      tgm->tm_hour, tgm->tm_min, tgm->tm_sec,
036      xv_get(xv_get(menu, MENU_SELECTED_ITEM), MENU_STRING));
037    }
038
039    return menu;
040  }
```

Na Tabela 6-12, é mostrado um segmento do código fonte (em linguagem C), que realiza a monitoração das atividades do objeto **mArq**. Quando uma das opções deste menu for selecionada, a identificação da opção será gravada assim como o instante de tempo em que este evento ocorreu. O registro da interação do usuário com a interface pode ser efetuado para qualquer um dos seus objetos. O registro consiste na identificação do objeto e no instante de tempo em que ocorreu um evento a ele associado. Este tipo de registro pode ser observado na Tabela 6-13, onde estão presentes os dados registrados pelo coletor dinâmico sobre o comportamento do usuário durante uma sessão de interação.

Na Tabela 6-13 é apresentado o arquivo de coleta dinâmica. Nele consta a identificação do protótipo em análise, o nome do usuário e a sessão de utilização, bem como a data e hora do início da sessão de teste. Cada operação do usuário é registrada através da identificação do elemento ativado e do instante de tempo em que isto correu.

Tabela 6-13: Resultados da Coleta Dinâmica – Sequenciamento de ações

```
001  Directory.....: /home/nichols/marckson/Tese/BrowserDinamico
002  Collect File..: x3.txt
003  Project Name..: Project BrowserB
004  User Name.....: Marckson
005  Section ID....: Sec Cenario
006  Section Date..: 27-08-99
007  Log Limit Time: 600 secs
008
009  ------------------------------------------------------------------------------
010   Prototype Name          Object type        Time    Object ID            Menu Name
011  ------------------------------------------------------------------------------
012  [BrowserB.G           ][:button         ] 11:34:11 bArqBW
013  [BrowserB.G           ][:menu           ] 11:34:12 Abrir Pagina          mArq
014  [AbrirPagina.G        ][:button         ] 11:34:14 NcnfAP
015  [BrowserB.G           ][:button         ] 11:34:20 bArqBW
016  [BrowserB.G           ][:menu           ] 11:34:22 Salvar Pagina         mArq
017  [SalvarPagina.G       ][:button         ] 11:34:27 NprnSP
018  [ImprimirPagina.G     ][:button         ] 11:34:29 NcnfIP
019  [SalvarPagina.G       ][:button         ] 11:34:36 NcanSP
020  [BrowserB.G           ][:button         ] 11:34:42 bArqBW
021  [BrowserB.G           ][:menu           ] 11:34:43 Abrir Pagina          mArq
022  [AbrirPagina.G        ][:button         ] 11:34:46 NcnfAP
023  [BrowserB.G           ][:button         ] 11:34:51 NaVltBW
024  [BrowserB.G           ][:button         ] 11:34:53 NaAvnBW
025  [BrowserB.G           ][:button         ] 11:34:55 bArqBW
026  [BrowserB.G           ][:menu           ] 11:34:57 Imprimir Pagina       mArq
027  [ImprimirPagina.G     ][:button         ] 11:34:59 NcnfIP
028  [BrowserB.G           ][:button         ] 11:35:05 NaStpBW
029  [BrowserB.G           ][:button         ] 11:35:11 bAjdBW
030  [BrowserB.G           ][:menu           ] 11:35:13 Indice                mAjd
031  [Ajuda.G              ][:button         ] 11:35:14 NfchAJ
032  [BrowserB.G           ][:button         ] 11:35:17 bArqBW
033  [BrowserB.G           ][:menu           ] 11:35:20 Sair                  mArq
```

Para exemplificar os resultados apresentados na Tabela 6-13, é analisado abaixo no texto um trecho do arquivo de resultados.

```
012: [BrowserB.G          ][:button        ] 11:34:11 bArqBW
       O botão bArqBW da janela BrowserB foi pressionado em 11:34:11

013:    [BrowserB.G          ][:menu          ] 11:34:12 Abrir Pagina    mArq
       A opção de menu Abrir Pagina foi selecionada em 11:34:12
```

Após o término da sessão de utilização, o módulo de coleta dinâmica é utilizado para gerar estatísticas sobre o número de acessos aos elementos da interface, conforme pode ser observado na Tabela 6-14.

Tabela 6-14: Resultados da Coleta Dinâmica - Incidência dos objetos

```
001  FAIUnix 1.01 (c) 1995-1999 Marckson Sousa
002
003  Directory.....: /home/nichols/marckson/Tese/BrowserDinamico
004  Collect File..: x3.txt
005  Project Name..: Project BrowserB
006  User Name.....: Marckson
007  Section ID....: Sec Cenario
008  Section Date..: 27-08-99
009  Log Limit Time: 600 secs
010
011
012
013  Object type -   :button        ************************************************
014
015  Interface Name        Object ID           Menu Name                   Ocurrence
016
017  BrowserB.G            bArqBW                                                5
018  AbrirPagina.G         NcnfAP                                                2
019  SalvarPagina.G        NprnSP                                                1
020  ImprimirPagina.G      NcnfIP                                                2
021  SalvarPagina.G        NcanSP                                                1
022  BrowserB.G            NaVltBW                                               1
023  BrowserB.G            NaAvnBW                                               1
024  BrowserB.G            NaStpBW                                               1
025  BrowserB.G            bAjdBW                                                1
026  Ajuda.G               NfchAJ                                                1
027
028  Object type -   :menu          ************************************************
029
030  Interface Name        Object ID           Menu Name                   Ocurrence
031
032  BrowserB.G            Abrir Pagina        mArq                              2
033  BrowserB.G            Salvar Pagina       mArq                              1
034  BrowserB.G            Imprimir Pagina     mArq                              1
035  BrowserB.G            Indice              mAjd                              1
036  BrowserB.G            Sair                mArq                              1
```

Os resultados apresentados demonstram o registro da interação do usuário com um protótipo de interface para uma sessão de utilização. A apresentação destes resultados ainda pode ser visualizada sob a forma de quais cenários foram utilizados para a realização de determinada tarefa. Por exemplo, considerando a tarefa de imprimir página, obtém-se o resultado apresentado na Tabela 6-15.

Tabela 6-15: Resultados da Coleta Dinâmica (Cenários da tarefa 'Imprimir Página')

```
001  FAIUnix 1.01 (c) 1995-1999 Marckson Sousa
002
003
004  From: (bArqBW                                  )
005  To:   (NcnfIP                                  )
006
007  Event #                P a t h                   tFrom     tTo      Duration
008  ------- ---------------------------------------- -------- -------- -----------
009      1   (bArqBW                                  )
010          (Abrir Pagina          mArq              )
011          (NcnfAP                                  )
012          (bArqBW                                  )
013          (Salvar Pagina         mArq              )
014          (NprnSP                                  )
015          (NcnfIP                                  ) 11:34:11 11:34:29     18 secs
016
017      2   (bArqBW                                  )
018          (Abrir Pagina          mArq              )
019          (NcnfAP                                  )
020          (NaVltBW                                 )
021          (NaAvnBW                                 )
022          (bArqBW                                  )
023          (Imprimir Pagina       mArq              )
024          (NcnfIP                                  ) 11:34:42 11:34:59     17 secs
```

Após a realização de sessões de interação com o usuário, foram detectados dois novos problemas. O primeiro refere-se à existência de um botão ("Desfazer") sem significado na janela Renomear Favoritos (winRF) e a falta do botão ("Desfazer") na janela Organizar Favoritos (winOF), de onde se pode deduzir que durante a construção do protótipo houve uma troca de janelas no posicionamento do botão; e o segundo problema refere-se a falta de um botão que conduza a ajuda na janela Organizar Favoritos (winOF).

O primeiro problema pode ser resolvido através da retirada do botão da janela Renomear Favoritos e a sua inclusão na janela Organizar Favoritos, enquanto que para a resolução do segundo, é necessário retornar às etapas anteriores do processo avaliatório, conforme será discutido no Capítulo 7.

6.4 CONCLUSÕES

Este capítulo apresentou a ferramenta para coleta de dados FAIUnix e mostrou como a partir da análise dos dados coletados sobre a estrutura do projeto de interação, e dados obtidos da realização das tarefas, tais como os erros cometidos pelo usuário, é possível localizar problemas de usabilidade e melhorar o projeto da interface.

7 APLICAÇÃO ITERATIVA DO MÉTODO

Este capítulo apresenta a aplicação iterativa do método de concepção no projeto do *Browser-B*. Trata-se de um processo iterativo de especificação com o propósito de refinar os requisitos de projeto, aumentando a compreensão do contexto de uso do produto e das expectativas do cliente.

7.1 PROCEDIMENTO ITERATIVO

Embora as etapas do método sejam interdependentes, as atividades de avaliação e ajustes na especificação podem iniciar em qualquer ponto do método e mover-se para qualquer outro ponto, como é ilustrado na Figura 7.1.

Na primeira iteração, são construídos e analisados os modelos, na seqüência de passos apresentada no Capítulo 3. Uma vez construídos os modelos é possível realizar várias iterações até que requisitos de projeto sejam alcançados. O procedimento iterativo de aplicação do método é descrito a seguir:

- A concepção inicia com a análise da tarefa e prossegue com a análise de consistência deste modelo, como ilustra o Capítulo 4. Caso tenham sido encontradas inconsistências neste modelo, elas deverão ser corrigidas e, dependendo da natureza dos ajustes realizados (alteração de tarefas, restrições, etc), estes ajustes deverão ser propagados para as demais fases.

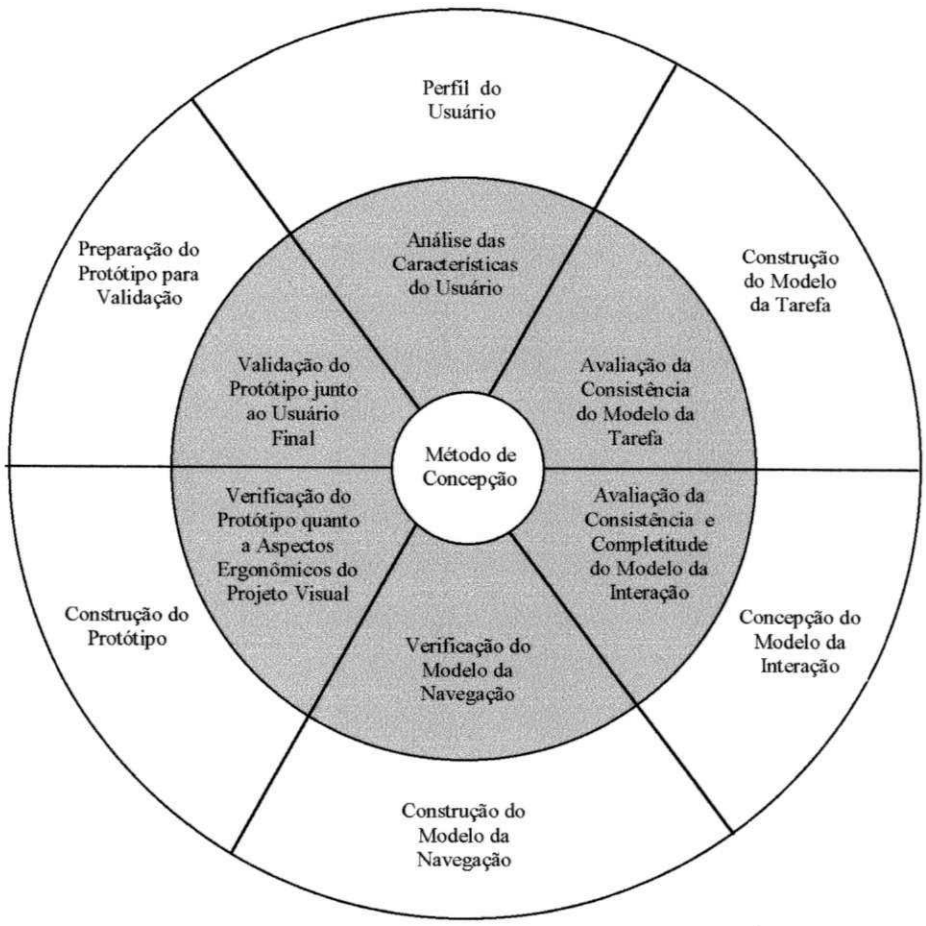

Figura 7-1: Aplicação Iterativa do Método

- A fase seguinte compreende a avaliação do modelo da interação, envolvendo dois passos: análise de consistência do modelo da interação e verificação do modelo da navegação.
  - Caso tenham sido encontradas inconsistências no Modelo da Interação, estas deverão ser corrigidas e, dependendo da natureza dos ajustes realizados, estes ajustes também deverão ser propagados para o modelo da navegação, ou podem também implicar ajustes no modelo da tarefa.
  - O Modelo da Navegação, por ser formal, passa por um processo de verificação podendo ou não necessitar de ajustes. Caso venha a ser modificado, dependendo da natureza das modificações, ajustes deverão ser feitos no modelo da interação, ou refletir ainda no modelo da tarefa.
- A fase seguinte consiste na validação do protótipo. Caso tenham sido detectados problemas, e dependendo da natureza das correções, ajustes deverão ser feitos apenas no protótipo ou podem refletir ainda sobre as fases anteriores (modelo da interação e modelo da tarefa).

7.2 Refinamento da Especificação do Navegador

De posse dos modelos e do protótipo obtidos na primeira iteração é possível iniciar o processo de refinamento dos requisitos, como passamos a descrever.

7.2.1 Segunda Iteração

A partir da análise dos resultados da verificação do modelo da navegação do *Browser-B*, apresentada no Capítulo 5, foi constatado que não existiam estados recorrentes (Tabela 5-11 – "*Home Markings: None*"), contrariando uma das propriedades do modelo da navegação.

Para solucionar este problema, é necessário retornar ao modelo da interação e analisar o projeto dos objetos disponíveis para a navegação. Após a análise, chegou-se à conclusão de que a tarefa "Finalizar Organizar Favoritos" não havia sido contemplada no modelo da interação e que estava faltando definir um objeto para ativação do retorno da janela winOF (Organizar Favoritos). Não havia um meio de deslocar-se para a janela anterior, e consequentemente não havia acesso à saída do sistema.

A versão modificada do modelo da interação é apresentada na Tabela 7-1. O item que foi acrescentado é destacado em negrito/sombreado.

Tabela 7-1: Modelo da Interação- Descrição dos Botões (2ª iteração)

Descrição de Botões				
Id da Janela	*Id do Botão*	*Descrição*	*Ativa*	*Comportamento*
WinOF	desOF	Desfazer	desOF	*No_Navigation* *
	renOF	Renomear	winRF	*Navigation*
	delOF	Excluir	winEF	*Navigation*
	hlpOF	Ajuda	winAJ	*Navigation*
	fchOF	**Fechar**	-	***Undo & Close***

Dada a relação entre os modelos, esta alteração no modelo da interação, torna necessária a modificação do modelo da navegação, que por sua vez demanda a atualização de algumas declarações no modelo em redes de Petri.

A versão modificada do modelo da navegação é apresentada nas Tabelas 7-2 a 7-5 apresentadas a seguir. Os itens modificados são: na Tabela 7-2 a inclusão de um elemento da cor *BUTTON* (fchOF) nas declarações da rede; na Tabela 7-3 a inclusão de um objeto *BUTTON* (1`b(fchOF)) no lugar *Undo & Close*; na Tabela 7-4 a inclusão de um objeto *BUTTON* (b(fchOF)) no lugar *No_Navigation* e na Tabela 7-5 a inclusão de um objeto *BUTTON* (b(fchOF)) no lugar *Interface_Definition* na janela winOF. Os elementos incluídos são destacados em negrito/sombreado.

Tabela 7-2: Declarações Globais (2ª iteração)

```
color WINDOW         = with winBW | winAJ | winAP | winSP | winIP | winOF | winRF | winEF |
                            winAF;
color BUTTON         = with bArqBW | bEdtBW | bExbBW | bIrPBW | bFavBW | bAjdBW | aVltBW |
                            aAvnBW | aStpBW | aRefBW | cntAJ | indAJ | psqAJ | exbAJ | fchAJ |
                            cnfAP | canAP | hlpAP | prnSP | cnfSP | canSP | hlpSP | cnfIP |
                            canIP | hlpIP | desOF | renOF | delOF | hlpOF | fchOF | desRF |
                            cnfRF | canRF | cnfEF | canEF | hlpEF | cnfAF | canAF | hlpAF;
color MENU           = with mArq | mEdt | mExb | mIrP | mFav | mAjd | smCnt;
color OP_MENU        = with o_abrpag | o_savpag | o_imppag | o_sair | o_recort | o_copiar |
                            o_colar | o_parar | o_atualz | o_codfnt | o_voltar | o_avanc |
                            o_adcfav | o_orgfav | o_hcont | o_hindex | o_pesq | o_hgeral |
                            o_habrpag | o_hsavpag | o_himppag | o_hadcfav | o_horgfav |
                            o_hexcfav;
color ACTION         = union b:BUTTON + om:OP_MENU;
color ACTWIN         = product ACTION * WINDOW;
color ACTION_LIST    = list ACTION;
color AA_LIST        = product ACTION * ACTION_LIST;
color INTERFACE      = product WINDOW * ACTION_LIST;
color INTERFACE_LIST = list INTERFACE;
color UNDO_LIMIT     = int;

fun is_in(elem,[]) = false |
       is_in(elem,x::y) = if elem=x then true else is_in(elem,y);

fun take (_, 0)    = []
  | take([], _)    = raise Subscript
  | take(h::t, n) = h::take(t, n-1) handle Overflow => raise Subscript;

fun limit_list(ilist:INTERFACE_LIST, u_lim:int):INTERFACE_LIST =
  if length(ilist) > u_lim then take(ilist,u_lim) else ilist;

var w: WINDOW;
var wa: WINDOW;
var bt: BUTTON;
var om1: OP_MENU;
var act: ACTION;
var al: ACTION_LIST;
var ala: ACTION_LIST;
var il: INTERFACE_LIST;
var ul: UNDO_LIMIT;
```

Tabela 7-3: Declarações - Lugar "*Undo & Close*" (UC_SET) – (2ª iteração)

```
1`b(fchAJ) + 1`b(cnfAP) + 1`b(canAP) + 1`b(cnfSP) + 1`b(canSP) + 1`b(cnfIP) +
1`b(canIP) + 1`b(fchOF) + 1`b(cnfRF) + 1`b(canRF) + 1`b(cnfEF) + 1`b(canEF) +
1`b(cnfAF) + 1`b(canAF)
```

Tabela 7-4: Declarações - Lugar "*No_Navigation*" (NNV_SET) – (2ª iteração)

```
1`(b(bArqBW),[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),
   b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bEdtBW),[om(o_recort),om(o_copiar),om(o_colar),b(bArqBW),b(bExbBW),b(bIrPBW),b(bFavBW),
   b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bExbBW),[om(o_parar),om(o_atualz),om(o_codfnt),b(bArqBW),b(bEdtBW),b(bIrPBW),b(bFavBW),
   b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bIrPBW),[om(o_voltar),om(o_avanc),b(bArqBW),b(bEdtBW),b(bExbBW),b(bFavBW),b(bAjdBW),
   b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bFavBW),[om(o_adcfav),om(o_orgfav),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bAjdBW),
   b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(bAjdBW),[om(o_hcont),om(o_hindex),om(o_pesq),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),
   b(bFavBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(b(aVltBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(aAvnBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(aStpBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(aRefBW),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) +
1`(b(cntAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(indAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(psqAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(exbAJ),[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(b(desOF),[b(desOF),b(renOF),b(delOF),b(hlpOF),b(fchOF)]) +
1`(b(desRF),[b(desRF),b(cnfRF),b(canRF)]) +
1`(om(o_recort),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_copiar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_colar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_parar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_atualz),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_codfnt),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_voltar),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_avanc),[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)]) +
1`(om(o_hcont),[om(o_hgeral),om(o_habrpag),om(o_hsavpag),om(o_himppag),om(o_hadcfav),
   om(o_horgfav),om(o_hexcfav),b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(aVltBW),
   b(aAvnBW),b(aStpBW),b(aRefBW)])
```

Tabela 7-5: Declarações - Lugar "*Interface_Definition*" (INT_DEF) (2ª iteração)

```
1`(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) + 1`(winAJ,[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(winAP,[b(cnfAP),b(canAP),b(hlpAP)]) + 1`(winSP,[b(prnSP),b(cnfSP),b(canSP),b(hlpSP)]) +
1`(winIP,[b(cnfIP),b(canIP),b(hlpIP)]) +
1`(winOF,[b(desOF),b(renOF),b(delOF),b(hlpOF),b(fchOF)]) +
1`(winRF,[b(desRF),b(cnfRF),b(canRF)]) + 1`(winEF,[b(cnfEF),b(canEF),b(hlpEF)]) +
1`(winAF,[b(cnfAF),b(canAF),b(hlpAF)])
```

Feitas as alterações, foi realizada uma nova verificação do modelo da navegação, obtendo-se os resultados que serão analisados a seguir.

Tabela 7-6: Resultados da Ferramenta *Design/CPN* (2ª iteração)

```
Statistics
------------------------------------------------------------------------
  Occurrence Graph
    Nodes:  26
    Arcs:   169
    Secs:   1
    Status: Full

Boundedness Properties
------------------------------------------------------------------------
  Best Integers Bounds    Upper      Lower
  New'E_SET 1             1          1
  New'INT_DEF 1           9          9
  New'Interface 1         1          1
  New'NNV_SET 1           25         25
  New'NV_SET 1            23         23
  New'UC_SET 1            14         14
  New'U_LIM 1             1          1


Home Properties
------------------------------------------------------------------------
  Home Markings:  [12]


Liveness Properties
------------------------------------------------------------------------
  Dead Markings:  [12]
```

Análise dos resultados após as alterações

Como pode ser observado na Tabela 7-6, os resultados da ferramenta *Design/CPN* apontam para apenas uma marcação morta (marcação 12). Considerando que tipicamente, em um sistema de interfaces, é desejável que apenas os estados de "*exit*" (saída do sistema) correspondam a marcações mortas, é necessário analisar esta marcação a partir de seus descritores. Da análise dos descritores observa-se que a marcação '12' corresponde à saída do sistema. Esta constatação pode ser confirmada pela verificação do disparo da transição "*Exit*", que conduz a uma marcação morta, como ilustra a Figura 7-2.

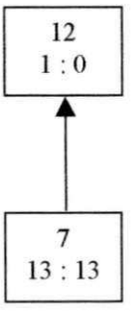

Figura 7-2: Marcação "morta" (12) e seu antecessor (7) (2ª iteração)

Os descritores relativos à Figura 7-2 são apresentados a seguir:

```
12
New'Interface 1: 1`[]
```

```
69:7->12
New'EXIT 1:
{w=winBW,il=[],al=[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),
b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)],act=om(o_sair)}
```

```
7
New'Interface 1:
1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),
b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]
```

Esta segunda iteração ilustrou a situação de retorno ao modelo da interação em decorrência da verificação do modelo da navegação. A próxima seção, apresenta uma terceira iteração que ilustra uma nova iteração com ajustes na especificação de caráter mais abrangente.

7.2.2 Terceira Iteração

A partir da observação da interação de um usuário com o protótipo, apresentada no Capítulo 6, foram constatados os seguintes problemas:

- *A existência do botão "Desfazer" na janela Renomear Favoritos ao invés da janela Organizar Favoritos.*
- *Não havia acesso à ajuda, na janela Renomear Favoritos.*

Para a solução do primeiro problema basta apenas retirar o botão "Desfazer" da janela Renomear Favoritos e colocá-lo na janela Organizar Favoritos. Por outro lado, para a solução do segundo problema, é necessário realizar uma análise de consistência do modelo da interação, investigando-se o conjunto de objetos ativadores. Neste modelo não foi constatado qualquer problema de consistência, levando à análise do modelo da tarefa.

No modelo da tarefa o objetivo foi verificar se havia uma relação entre a tarefa "solicitar ajuda" e a tarefa "renomear favoritos". A partir da análise verificou-se que a solicitação de ajuda não havia sido prevista para a tarefa "renomear favoritos" por ocasião da construção do modelo da tarefa. Após a inclusão desta facilidade no modelo da tarefa foi

necessária a propagação das correções pelas demais fases do processo de especificação conforme é apresentado a seguir.

A versão original do modelo da tarefa apresentada na Figura A-11 foi modificada para a versão descrita na Figura 7-3.

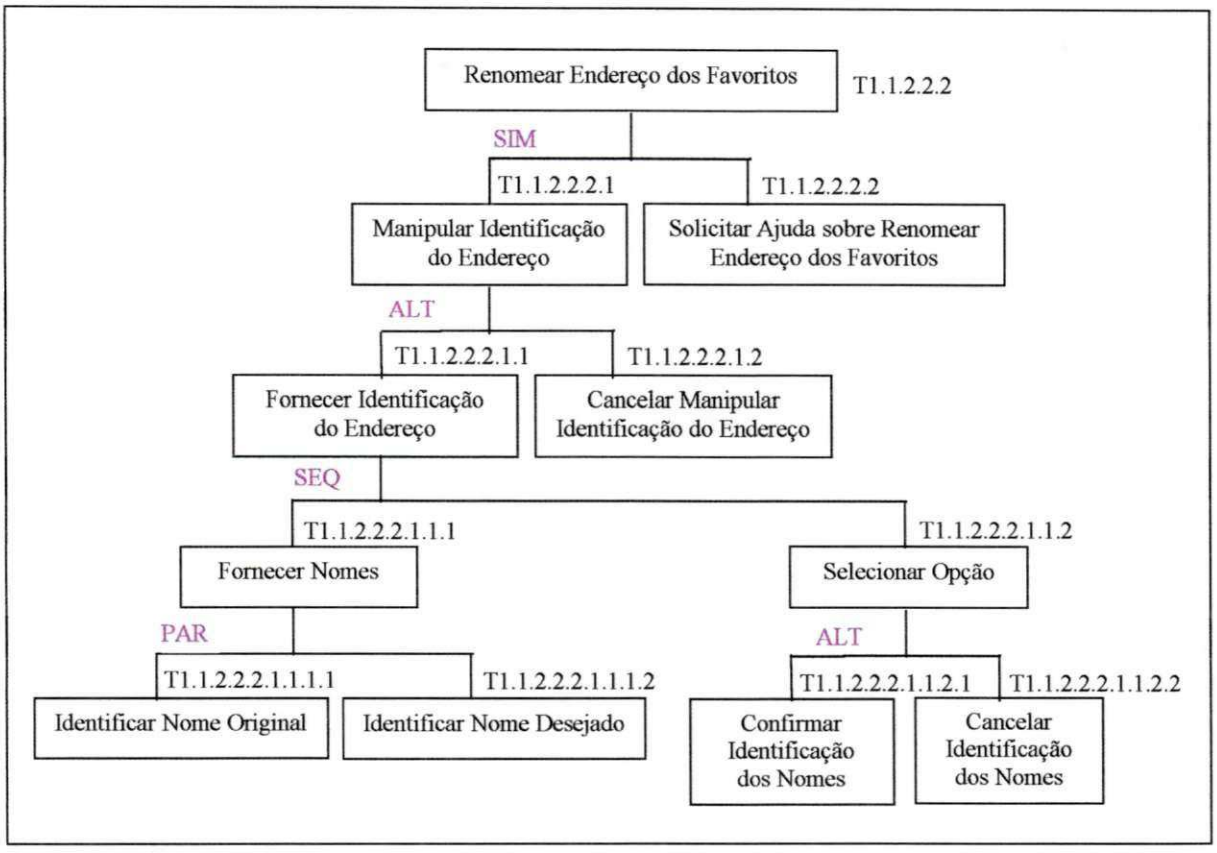

Figura 7-3: Tarefa Renomear Favoritos (3ª iteração)

A modificação do modelo da tarefa levou à modificação do modelo da interação de modo a incluir o elemento da ativação da solicitação de ajuda, na janela winRF. Esta modificação é mostrada nas Tabelas 7-7 e 7-8. O item acrescentado é destacado em negrito/sombreado.

Tabela 7-7: Modelo da Interação (3ª iteração)

Id. Descritor	Sub-Tarefa		Técnica	Tipo Objeto Ativador	Associação dos objetos (janela)
	Ação	Objeto da Tarefa			
T1.1.2.2.2.1.1.1.1	Identificar	Nome Original	Manipulação Texto	Campo de Texto	Renomear Favoritos
T1.1.2.2.2.1.1.1.2	Identificar	Nome Desejado	Manipulação Texto	Campo de Texto	Renomear Favoritos
T1.1.2.2.2.1.1.2.1	Confirmar Identificação	Dos Nomes	Ativação por Botão	Botão	Renomear Favoritos
T1.1.2.2.2.1.1.2.2	Cancelar Identificação	Dos Nomes	Ativação por Botão	Botão	Renomear Favoritos
T1.1.2.2.2.2	**Solicitar Ajuda**	**Sobre Organizar Favoritos**	**Ativação por Botão**	**Botão**	**Organizar Favoritos**

Tabela 7-8: Modelo da Interação – Descrição dos Botões (3ª iteração)

Descrição de Botões				
Id da Janela	*Id do Botão*	*Descrição*	*Ativa*	*Comportamento*
WinRF	cnfRF	Confirmar	-	*Undo & Close*
	canRF	Cancelar	-	*Undo & Close*
	hlpRF	**Ajuda**	**WinAJ**	***Navigation***

Em decorrência da alteração no modelo da interação, foi necessário modificar e verificar o modelo da navegação. As alterações no modelo de navegação são apresentadas a seguir em negrito/sombreado nas Tabelas 7-9 a 7-11. As modificações consistiram na inclusão de um objeto *BUTTON* (hlpRF) nas declarações da rede na Tabela 7-9; na inclusão do elemento (1`(b(hlpRF),winAJ)) no lugar *Navigation* na Tabela 7-10 e na inclusão de um objeto *BUTTON* (b(hlpRF)) no lugar *Interface_Definition* na Tabela 7-11.

Tabela 7-9: Declarações Globais (3ª iteração)

```
color WINDOW         = with winBW | winAJ | winAP | winSP | winIP | winOF | winRF | winEF |
                            winAF;
color BUTTON         = with bArqBW | bEdtBW | bExbBW | bIrPBW | bFavBW | bAjdBW | aVltBW |
                            aAvnBW | aStpBW | aRefBW | cntAJ | indAJ | psqAJ | exbAJ | fchAJ |
                            cnfAP | canAP | hlpAP | prnSP | cnfSP | canSP | hlpSP | cnfIP |
                            canIP | hlpIP | desOF | renOF | delOF | hlpOF | fchOF | desRF |
                            cnfRF | canRF | hlpRF | cnfEF | canEF | hlpEF | cnfAF | canAF |
                            hlpAF;
color MENU           = with mArq | mEdt | mExb | mIrP | mFav | mAjd | smCnt;
color OP_MENU        = with o_abrpag | o_savpag | o_imppag | o_sair | o_recort | o_copiar |
                            o_colar | o_parar | o_atualz | o_codfnt | o_voltar | o_avanc |
                            o_adcfav | o_orgfav | o_hcont | o_hindex | o_pesq | o_hgeral |
                            o_habrpag | o_hsavpag | o_himppag | o_hadcfav | o_horgfav |
                            o_hexcfav;
color ACTION         = union b:BUTTON + om:OP_MENU;
color ACTWIN         = product ACTION * WINDOW;
color ACTION_LIST    = list ACTION;
color AA_LIST        = product ACTION * ACTION_LIST;
color INTERFACE      = product WINDOW * ACTION_LIST;
color INTERFACE_LIST = list INTERFACE;
color UNDO_LIMIT     = int;

fun is_in(elem,[]) = false |
       is_in(elem,x::y) = if elem=x then true else is_in(elem,y);

fun take (_, 0)   = []
  | take([], _)   = raise Subscript
  | take(h::t, n) = h::take(t, n-1) handle Overflow => raise Subscript;

fun limit_list(ilist:INTERFACE_LIST, u_lim:int):INTERFACE_LIST =
  if length(ilist) > u_lim then take(ilist,u_lim) else ilist;

var w: WINDOW;
var wa: WINDOW;
var bt: BUTTON;
var om1: OP_MENU;
var act: ACTION;
var al: ACTION_LIST;
var ala: ACTION_LIST;
var il: INTERFACE_LIST;
var ul: UNDO_LIMIT;
```

Tabela 7-10: Declarações - Lugar "*Navigation*" (NV_SET) – (3ª iteração)

```
1`(b(hlpAP),winAJ) + 1`(b(prnSP),winIP) + 1`(b(hlpSP),winAJ) +
1`(b(hlpIP),winAJ) + 1`(b(renOF),winRF) + 1`(b(delOF),winEF) +
1`(b(hlpOF),winAJ) + 1`(b(hlpEF),winAJ) + 1`(b(hlpAF),winAJ) +
1`(b(hlpRF),winAJ) +
1`(om(o_abrpag),winAP) +1`(om(o_savpag),winSP) +
1`(om(o_imppag),winIP) + 1`(om(o_adcfav),winAF) +
1`(om(o_orgfav),winOF) + 1`(om(o_hindex),winAJ) +
1`(om(o_pesq),winAJ) + 1`(om(o_hgeral),winAJ) + 1`(om(o_habrpag),winAJ) +
1`(om(o_hsavpag),winAJ) + 1`(om(o_himppag),winAJ) + 1`(om(o_hadcfav),winAJ) +
1`(om(o_horgfav),winAJ) + 1`(om(o_hexcfav),winAJ)
```

Tabela 7-11: Declarações - Lugar "*Interface_Definition*" (INT_DEF) – (3ª iteração)

```
1`(winBW,[b(bArqBW),b(bEdtBW),b(bExbBW),b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),
   b(aStpBW),b(aRefBW)]) + 1`(winAJ,[b(cntAJ),b(indAJ),b(psqAJ),b(exbAJ),b(fchAJ)]) +
1`(winAP,[b(cnfAP),b(canAP),b(hlpAP)]) + 1`(winSP,[b(prnSP),b(cnfSP),b(canSP),b(hlpSP)]) +
1`(winIP,[b(cnfIP),b(canIP),b(hlpIP)]) +
1`(winOF,[b(desOF),b(renOF),b(delOF),b(hlpOF),b(fchOF)]) +
1`(winRF,[b(desRF),b(cnfRF),b(canRF),b(hlpRF)]) + 1`(winEF,[b(cnfEF),b(canEF),b(hlpEF)]) +
1`(winAF,[b(cnfAF),b(canAF),b(hlpAF)])
```

Realizadas as alterações, foi realizada uma nova verificação do modelo da navegação, obtendo-se os resultados apresentados na Tabela 7-12.

Tabela 7-12: Resultados da Ferramenta *Design/CPN* (3ª iteração)

```
 Statistics
------------------------------------------------------------------------
  Occurrence Graph
    Nodes:  28
    Arcs:   178
    Secs:   1
    Status: Full

 Boundedness Properties
------------------------------------------------------------------------

  Best Integers Bounds    Upper       Lower
  New'E_SET 1             1           1
  New'INT_DEF 1           9           9
  New'Interface 1         1           1
  New'NNV_SET 1           25          25
  New'NV_SET 1            24          24
  New'UC_SET 1            14          14
  New'U_LIM 1             1           1


Home Properties
------------------------------------------------------------------------
  Home Markings:  [12]


 Liveness Properties
------------------------------------------------------------------------
  Dead Markings:  [12]
```

Análise dos resultados da verificação do modelo da navegação

De acordo com o relatório da ferramenta *Design/CPN* para o modelo da navegação modificado na terceira iteração, existe apenas uma marcação morta (marcação 12). Esta marcação corresponde à saída do sistema (*Exit*), conforme é mostrado pelos descritores a seguir.

```
12
New'Interface 1: 1`[]
```

```
69:7->12
New'EXIT 1:
{w=winBW,il=[],al=[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),
b(bIrPBW),b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)],act=om(o_sair)}
```

```
7
New'Interface 1:
1`[(winBW,[om(o_abrpag),om(o_savpag),om(o_imppag),om(o_sair),b(bEdtBW),b(bExbBW),b(bIrPBW),
b(bFavBW),b(bAjdBW),b(aVltBW),b(aAvnBW),b(aStpBW),b(aRefBW)])]
```

Análise dos caminhos de acesso na navegação

Como já havia sido discutido no Capítulo 5, se todas as marcações da rede forem recorrentes pode-se interpretar como verdadeira a existência de caminhos de acesso entre todos os pontos da interface. Para que esta característica fosse analisada foram eliminadas todas as saídas do sistema, retirando-se todas as fichas do lugar *E_SET*. Obtidos os resultados, foram analisadas as novas marcações recorrentes. Os resultados obtidos demonstram que, no caso do Navegador, existem caminhos de acesso entre todos os pontos da interface. Este resultado é mostrado a seguir, na Tabela 7-13.

Tabela 7-13: Resultados da Ferramenta *Design/CPN* (3ª iteração)

```
Statistics
------------------------------------------------------------------------
  Occurrence Graph
    Nodes:  27
    Arcs:   177
    Secs:   1
    Status: Full

Boundedness Properties
------------------------------------------------------------------------
  Best Integers Bounds    Upper      Lower
  New'E_SET 1             0          0
  New'INT_DEF 1           9          9
  New'Interface 1         1          1
  New'NNV_SET 1           25         25
  New'NV_SET 1            24         24
  New'UC_SET 1            14         14
  New'U_LIM 1             1          1


Home Properties
------------------------------------------------------------------------
  Home Markings:  All

Liveness Properties
------------------------------------------------------------------------
  Dead Markings:  None
```

Tendo modificado e verificado o modelo da navegação, o próximo passo da concepção consiste na modificação do protótipo do navegador de modo a incluir a ajuda na janela winRF (renomear favoritos) e a consertar o posicionamento do botão "Desfazer". As janelas modificadas são apresentadas nas Figuras 7-4 e 7-5:

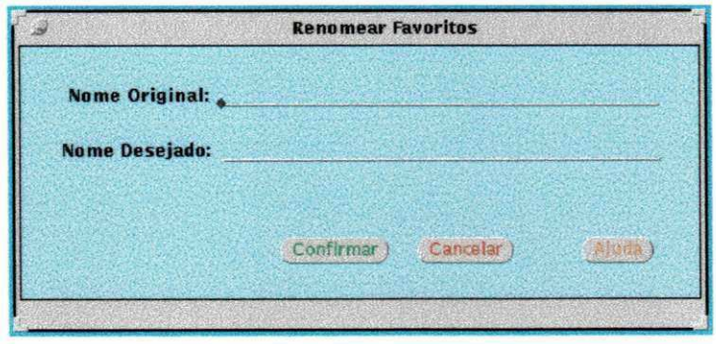

Figura 7-4: Janela 'Renomear Favoritos' (3ª iteração)

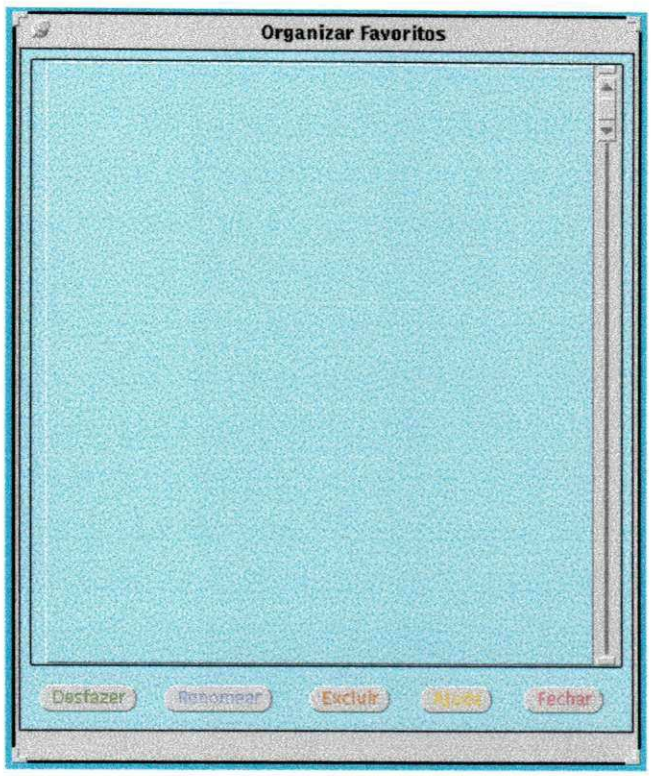

Figura 7-5: Janela Organizar Favoritos (3ª iteração)

Uma vez modificado o protótipo, este pode ser novamente submetido ao usuário para novas sessões de avaliação. Se outros problemas forem encontrados, novas iterações se fazem necessárias e o ciclo se reinicia até que soluções satisfatórias sejam encontradas.

7.3 TESTES DOS PROTÓTIPOS COM O USUÁRIO FINAL

Foi definido um roteiro de tarefas para ser realizado pelo usuário final com o protótipo inicial e aquele resultante das modificações da aplicação do método proposto neste trabalho. Durante a interação do usuário com os protótipos, a ferramenta FAIUnix foi utilizada para registrar informações sobre a interação.

Roteiro de Tarefas

1. Abrir Página, Ver Código Fonte, Salvar Página, Imprimir, Adicionar Favoritos
2. Abrir Página, Adicionar Favoritos, Organizar Favoritos, Renomear Favoritos (1° favorito)
3. Abrir Página, Adicionar Favoritos, Organizar Favoritos, Excluir Favoritos (2° favorito), Finalizar Browser

Resultados Obtidos

A tabela 7-14 compara os resultados da avaliação realizada sobre um protótipo inicial e um protótipo resultante das modificações da aplicação do método. A partir dela é possível observar que no protótipo revisado a duração da tarefa foi reduzida de 6'56'' para 2'56'' (que representa uma redução de 57,69%). Observa-se ainda que o número de erros e solicitações de ajuda foram eliminados. O número de erros foi reduzido de 3 para 0, e o número de solicitações de ajuda de 3 para 0.

Tabela 7-14: Dados comparativos das sessões de coleta

	Sessão 1 (Iteração 1)	**Sessão 2 (Iteração 3)**
Duração da sessão	14:59:43 a 15:06:38 = 6' 56''	15:07:59 a 15:10:55 = 2' 56''
Duração da tarefa 1	14:59:43 a 15:01:44 = 2' 1''	15:07:59 a 15:09:24 = 1' 25''
Duração da tarefa 2	15:01:44 a 15:04:59 = 3' 15''	15:09:24 a 15:10:11 = 47''
Duração da tarefa 3	15:04:59 a 15:06:38 = 1' 40''	15:10:11 a 15:10:55 = 44''
Solicitação de ajuda	3	0
Contexto da solicitação de ajuda	Organizar Favoritos (Tarefa 2 – 15:02:55) Organizar Favoritos (Tarefa 2 – 15:03:29) Organizar Favoritos (Tarefa 3 – 15:06:15)	-
Incidência de erros (botão cancelar ação)	3	0
Contexto de cancelar ação	Adicionar Favoritos (Tarefa 1 – 15:01:20) Renomear Favoritos (Tarefa 2 – 15:04:20) Renomear Favoritos (Tarefa 3 – 15:06:13)	-

A partir dos resultados da ferramenta FAIUnix, é possível observar o dimensionamento dos recursos utilizados pelo usuário, onde pode-se verificar quais os elementos mais utilizados. Nas Tabelas 7-15 e 7-16 são apresentados os recursos utilizados para os dois protótipos analisados.

Tabela 7-15: Resultados do Dimensionamento de Recursos – Sessão 1

Dimensionamento de Recursos – Sessão 1 (Iteração 1)

Object type - :button **********

Interface Name	Object ID	Menu Name	Ocurrence
BrowserB.G	bArqBW		6
AbrirPagina.G	NcnfAP		3
BrowserB.G	bExbBW		1
SalvarPagina.G	NcnfSP		1
ImprimirPagina.G	NcnfIP		1
BrowserB.G	bFavBW		6
AdicionarFavoritos.G	NcanAF		1
AdicionarFavoritos.G	NcnfAF		3
OrganizarFavoritos.G	NrenOF		3
RenomearFavoritos.G	NcnfRF		1
OrganizarFavoritos.G	NhlpOF		3
Ajuda.G	NexbAJ		3
Ajuda.G	NindAJ		3
Ajuda.G	NpsqAJ		2
Ajuda.G	NfchAJ		3
Ajuda.G	NcntAJ		2
RenomearFavoritos.G	NcanRF		2
OrganizarFavoritos.G	NdelOF		3
ExcluirFavoritos.G	NgtmEF		2
ExcluirFavoritos.G	NcnfEF		1

Object type - :menu **********

Interface Name	Object ID	Menu Name	Ocurrence
BrowserB.G	Abrir Pagina	mArq	3
BrowserB.G	Ver Codigo Fonte	mExb	1
BrowserB.G	Salvar Pagina	mArq	1
BrowserB.G	Imprimir Pagina	mArq	1
BrowserB.G	Adicionar	mFav	4
BrowserB.G	Organizar	mFav	2
BrowserB.G	Sair	mArq	1

Tabela 7-16: Resultados do Dimensionamento de Recursos – Sessão 2

Dimensionamento de Recursos – Sessão 2 (Iteração 3)

Object type - :button **********

Interface Name	Object ID	Menu Name	Ocurrence
BrowserB.G	bArqBW		6
AbrirPagina.G	NcnfAP		3
BrowserB.G	bExbBW		1
SalvarPagina.G	NcnfSP		1
ImprimirPagina.G	NcnfIP		1
BrowserB.G	bFavBW		5
AdicionarFavoritos.G	NcnfAF		3
OrganizarFavoritos.G	NrenOF		1
RenomearFavoritos.G	NcnfRF		1
OrganizarFavoritos.G	NfchOF		2
OrganizarFavoritos.G	NdelOF		1
ExcluirFavoritos.G	NcnfEF		1

Object type - :menu **********

Interface Name	Object ID	Menu Name	Ocurrence
BrowserB.G	Abrir Pagina	mArq	3
BrowserB.G	Ver Codigo Fonte	mExb	1
BrowserB.G	Salvar Pagina	mArq	1
BrowserB.G	Imprimir Pagina	mArq	1
BrowserB.G	Adicionar	mFav	3
BrowserB.G	Organizar	mFav	2
BrowserB.G	Sair	mArq	1

Da análise dos cenários, apresentados nas Tabelas 7-17 a 7-19, fica evidente a extensão do cenário percorrido pelo usuário na sessão realizada com o protótipo inicial. Este fato tem origem na inexistência do botão "Fechar" na janela "Organizar Favoritos".

Uma outra constatação resultante da análise destes dados diz respeito ao processo cognitivo do usuário o qual reutiliza a solução entrada para realizar a tarefa 2 (sair da página "Renomear Favoritos" a partir da navegação para a janela "Excluir Favoritos") quando da realização da tarefa 3.

Tabela 7-17: Análise dos Cenários para a Tarefa 1

Plano de Teste	Abrir Página, Ver Código Fonte, Salvar Página, Imprimir, Adicionar Favoritos
Sessão 1	[BrowserB.G][:button] 14:59:43 bArqBW [BrowserB.G][:menu] 14:59:44 Abrir Pagina mArq [AbrirPagina.G][:button] 14:59:53 NcnfAP [BrowserB.G][:button] 15:00:03 bExbBW [BrowserB.G][:menu] 15:00:08 Ver Codigo Fonte mExb [BrowserB.G][:button] 15:00:14 bArqBW [BrowserB.G][:menu] 15:00:16 Salvar Pagina mArq [SalvarPagina.G][:button] 15:00:39 NcnfSP [BrowserB.G][:button] 15:00:50 bArqBW [BrowserB.G][:menu] 15:00:52 Imprimir Pagina mArq [ImprimirPagina.G][:button] 15:01:02 NcnfIP [BrowserB.G][:button] 15:01:11 bFavBW [BrowserB.G][:menu] 15:01:12 Adicionar mFav [AdicionarFavoritos.G][:button] 15:01:20 NcanAF [BrowserB.G][:button] 15:01:22 bFavBW [BrowserB.G][:menu] 15:01:23 Adicionar mFav [AdicionarFavoritos.G][:button] 15:01:34 NcnfAF
Sessão 2	[BrowserB.G][:button] 15:07:59 bArqBW [BrowserB.G][:menu] 15:08:01 Abrir Pagina mArq [AbrirPagina.G][:button] 15:08:09 NcnfAP [BrowserB.G][:button] 15:08:16 bExbBW [BrowserB.G][:menu] 15:08:22 Ver Codigo Fonte mExb [BrowserB.G][:button] 15:08:30 bArqBW [BrowserB.G][:menu] 15:08:32 Salvar Pagina mArq [SalvarPagina.G][:button] 15:08:46 NcnfSP [BrowserB.G][:button] 15:08:50 bArqBW [BrowserB.G][:menu] 15:08:54 Imprimir Pagina mArq [ImprimirPagina.G][:button] 15:09:01 NcnfIP [BrowserB.G][:button] 15:09:08 bFavBW [BrowserB.G][:menu] 15:09:09 Adicionar mFav [AdicionarFavoritos.G][:button] 15:09:18 NcnfAF

Tabela 7-18: Análise dos Cenários para a Tarefa 2

Plano de Teste	Abrir Página, Adicionar Favoritos, Organizar Favoritos, Renomear Favoritos (1° favorito)
Sessão 1	[BrowserB.G][:button] 15:01:45 bArqBW [BrowserB.G][:menu] 15:01:46 Abrir Pagina mArq [AbrirPagina.G][:button] 15:01:56 NcnfAP [BrowserB.G][:button] 15:02:02 bFavBW [BrowserB.G][:menu] 15:02:04 Adicionar mFav [AdicionarFavoritos.G][:button] 15:02:10 NcnfAF [BrowserB.G][:button] 15:02:14 bFavBW [BrowserB.G][:menu] 15:02:16 Organizar mFav [OrganizarFavoritos.G][:button] 15:02:26 NrenOF [RenomearFavoritos.G][:button] 15:02:35 NcnfRF [OrganizarFavoritos.G][:button] 15:02:55 NhlpOF [Ajuda.G][:button] 15:03:00 NexbAJ [Ajuda.G][:button] 15:03:09 NindAJ [Ajuda.G][:button] 15:03:15 NpsqAJ [Ajuda.G][:button] 15:03:20 NfchAJ [OrganizarFavoritos.G][:button] 15:03:29 NhlpOF [Ajuda.G][:button] 15:03:34 NindAJ [Ajuda.G][:button] 15:03:39 NexbAJ [Ajuda.G][:button] 15:03:43 NcntAJ [Ajuda.G][:button] 15:03:48 NpsqAJ [Ajuda.G][:button] 15:03:54 NexbAJ [Ajuda.G][:button] 15:03:58 NcntAJ [Ajuda.G][:button] 15:04:02 NfchAJ [OrganizarFavoritos.G][:button] 15:04:17 NrenOF [RenomearFavoritos.G][:button] 15:04:20 NcanRF [OrganizarFavoritos.G][:button] 15:04:22 NdelOF [ExcluirFavoritos.G][:button] 15:04:39 NgtmEF
Sessão 2	[BrowserB.G][:button] 15:09:24 bArqBW [BrowserB.G][:menu] 15:09:25 Abrir Pagina mArq [AbrirPagina.G][:button] 15:09:32 NcnfAP [BrowserB.G][:button] 15:09:37 bFavBW [BrowserB.G][:menu] 15:09:38 Adicionar mFav [AdicionarFavoritos.G][:button] 15:09:43 NcnfAF [BrowserB.G][:button] 15:09:51 bFavBW [BrowserB.G][:menu] 15:09:52 Organizar mFav [OrganizarFavoritos.G][:button] 15:09:55 NrenOF [RenomearFavoritos.G][:button] 15:10:02 NcnfRF [OrganizarFavoritos.G][:button] 15:10:09 NfchOF

Tabela 7-19: Análise dos Cenários para a Tarefa 3

Plano de Teste	Abrir Página, Adicionar Favoritos, Organizar Favoritos, Excluir Favoritos (2° favorito), Finalizar Browser
Sessão 1	[BrowserB.G][:button] 15:05:00 bArqBW [BrowserB.G][:menu] 15:05:04 Abrir Pagina mArq [AbrirPagina.G][:button] 15:05:15 NcnfAP [BrowserB.G][:button] 15:05:22 bFavBW [BrowserB.G][:menu] 15:05:23 Adicionar mFav [AdicionarFavoritos.G][:button] 15:05:27 NcnfAF [BrowserB.G][:button] 15:05:35 bFavBW [BrowserB.G][:menu] 15:05:36 Organizar mFav [OrganizarFavoritos.G][:button] 15:05:39 NdelOF [ExcluirFavoritos.G][:button] 15:05:44 NcnfEF [OrganizarFavoritos.G][:button] 15:06:10 NrenOF [RenomearFavoritos.G][:button] 15:06:13 NcanRF [OrganizarFavoritos.G][:button] 15:06:15 NhlpOF [Ajuda.G][:button] 15:06:19 NindAJ [Ajuda.G][:button] 15:06:25 NfchAJ [OrganizarFavoritos.G][:button] 15:06:28 NdelOF [ExcluirFavoritos.G][:button] 15:06:32 NgtmEF [BrowserB.G][:button] 15:06:35 bArqBW [BrowserB.G][:menu] 15:06:38 Sair mArq
Sessão 2	[BrowserB.G][:button] 15:10:11 bArqBW [BrowserB.G][:menu] 15:10:13 Abrir Pagina mArq [AbrirPagina.G][:button] 15:10:21 NcnfAP [BrowserB.G][:button] 15:10:26 bFavBW [BrowserB.G][:menu] 15:10:27 Adicionar mFav [AdicionarFavoritos.G][:button] 15:10:31 NcnfAF [BrowserB.G][:button] 15:10:35 bFavBW [BrowserB.G][:menu] 15:10:37 Organizar mFav [OrganizarFavoritos.G][:button] 15:10:39 NdelOF [ExcluirFavoritos.G][:button] 15:10:44 NcnfEF [OrganizarFavoritos.G][:button] 15:10:49 NfchOF [BrowserB.G][:button] 15:10:52 bArqBW [BrowserB.G][:menu] 15:10:55 Sair mArq

7.4 Conclusões

Como resultado da aplicação iterativa do método, o projeto da interface do estudo de caso foi modificado em três iterações de modo a se adequar aos critérios de usabilidade propostos neste trabalho. Este procedimento poderia ser repetido tantas vezes quantas fossem necessárias até que a proposta de interação concebida atendesse a estes propósitos e à expectativa do usuário. Dado o tempo disponível para conclusão do trabalho, não foi realizada uma avaliação da satisfação do usuário, mas este é sem dúvida um aspecto de suma importância e que deve ser incluído no método durante sua aplicação.

Como foi exposto, após cada iteração, um protótipo é entregue ao cliente para avaliação e realimentação, possibilitando o retorno à fase de especificação de requisitos tantas vezes quanto necessárias. Cada modificação realizada em qualquer uma das etapas conduz a novas iterações de avaliação.

8 CONCLUSÕES

Este trabalho teve como objetivo principal a investigação de um método que associasse a avaliação baseada na verificação de modelos com a avaliação baseada na validação de protótipos, com o propósito de refinar o procedimento de análise da especificação da interface de produtos. O refinamento pretendido objetivou melhorar a capacidade de avaliar os mecanismos de interação projetados e aspectos do projeto visual, com os quais estão associados um número considerável de problemas de usabilidade.

Com o objetivo de apoiar o método proposto, foi desenvolvida uma ferramenta para apoiar a aplicação de um conjunto de métricas ao projeto da interação e em especial ao projeto visual. Foi também desenvolvida uma ferramenta para a coleta de dados da interação com protótipos que se mostrou útil ao tornar a coleta de dados mais rápida, menos dependente do avaliador, e capaz de reduzir custos da avaliação da usabilidade.

O procedimento de especificação/avaliação é realizado de forma iterativa até que a qualidade de uso, desejada para a interface do produto, seja alcançada, como foi apresentado no Capítulo 7.

A seguir é feita uma análise comparativa entre o método apresentado neste trabalho e outros métodos encontrados na literatura. Também é feita uma comparação entre as ferramentas de suporte ao método e outras ferramentas similares.

8.1 Discussão dos Resultados

Embora tanto a verificação de modelos quanto da validação de protótipos, venham sendo utilizadas na avaliação da especificação, não foi encontrada referência à utilização destes métodos em um mesmo contexto de avaliação.

Como foi apresentado no Capítulo 2, o MUSIC consiste de um conjunto de métodos para medir a usabilidade de produtos em um contexto. Estes métodos se fundamentam na comparação entre o percentual completado da tarefa e o percentual de qualidade dos objetivos alcançados. Assim como o método apresentado neste trabalho, ele também busca avaliar a eficiência no uso dos recursos de interação e a eficácia do projeto, porém acrescenta a avaliação da satisfação do usuário. Reconhecendo a importância da avaliação da satisfação do usuário, propomos que seja incorporada ao método, complementando a avaliação da eficácia e da eficiência da interação. No entanto, a ausência deste componente da avaliação não compromete o trabalho uma vez que a satisfação do usuário independe dos atributos específicos do produto. Observe-se ainda que no MUSIC a avaliação da usabilidade se apoia no registro em vídeo.

Lacerot e Paternó [LP 98] propuseram um método de avaliação baseado na análise do modelo de tarefa e em testes de usabilidade. O método se baseia na hipótese de que a análise de tarefas e análise de erros cometidos durante a interação com um produto podem sugerir como melhorar a interface. Comparado ao nosso método, do ponto de vista do registro da interação, este método dispõe de melhores recursos para avaliação uma vez que os dados coletados pela ferramenta QC/Replay são armazenados na forma de *scripts* que podem ser posteriormente "executados" para reproduzir a interação. Uma outra vantagem é que esta ferramenta não necessita de modificações no código da interface. Apesar de utilizarem o modelo da tarefa e a avaliação de usabilidade, o método proposto em [LP 98] e aquele aqui apresentado diferem fundamentalmente no modo como são detectados os problemas da interação. Além do mais, foram propostos para diferentes fases do ciclo de vida tornando difícil uma comparação entre eles.

Um outro aspecto que é tratado a partir de modelos é o mecanismo de navegação. O caráter genérico do modelo de navegação objetivou incorporar as principais características encontradas em diferentes situações de navegação em sistemas interativos. Para tanto se fundamentou nas relações temporais entre tarefas e ações do usuário, como descrito em UAN

[HH 93]. As situações modeladas foram exploradas no estudo de caso. A adequação do modelo a outros casos de avaliação da navegação poderá ser feita de forma simples, dispensando um conhecimento aprofundado da teoria de redes de Petri. A aplicação do modelo a situações de maior complexidade, onde haja um número maior de objetos de interação, causará apenas um aumento no número de fichas nos lugares da definição da navegação.

Os modelos de interfaces construídos com o formalismo de redes de Petri normalmente são dedicados a situações específicas [RPC 96][RPC 98][SE 96][PBSBD 93][PB 94][PBS 95] [PB 96], isto é, demandam mudanças na estrutura de suas redes para que possam representar outras situações. Observa-se também que estes modelos não se concentram nos aspectos de navegação tratados neste trabalho. Para que pudesse ser aplicado a um universo maior de casos, o modelo da navegação deveria representar a ativação e desativação de objetos da interação uma vez que esta característica dos objetos é necessária para analisar situações de bloqueio.

A exemplo da ferramenta *Sherlock* [MS 97], a FAIUnix, apresentada no Capítulo 6, também suporta a avaliação da distribuição espacial de objetos na tela, embora não ofereça recursos para análise de consistência. Por outro lado, os dados coletados atendem ao propósito da avaliação pretendida neste trabalho, do ponto de vista do projeto visual dos objetos da interface (janelas, botões, menus) e da estrutura da interação, a exemplo do projeto dos níveis de menus. O coletor de dados estáticos da ferramenta FAIUnix facilitou a tarefa de validação do protótipo para o projetista, ao apoiar a análise de questões estruturais da especificação da interface. Com os dados obtidos através desta ferramenta, sobre a distribuição espacial dos objetos na tela, a utilização de cores no projeto e o projeto dos objetos da interação, foi possível antecipar questões de usabilidade que só seriam detectadas por ocasião da realização de testes envolvendo o usuário.

O registro de dados obtido pela monitoração das ações do usuário através de *software logging*, torna os valores mais precisos do que valores obtidos por outras técnicas de coleta tais como vídeo ou áudio. Por exemplo o registro realizado pela ferramenta DRUM (*Diagnostic Recorder for Usability Measurement*), utilizada no projeto MUSiC [Bev 97], que consiste de estimativas com base no registro de eventos de subtarefas.

O foco das ferramentas de *software logging* tem sido a análise de desempenho, enfatizando o tempo gasto na execução de tarefa. Embora a ferramenta FAIUnix registre a

duração de tarefas e o tempo de ocorrência de eventos, seu foco é voltado para o estudo da navegação na interação, ou seja, quais os caminhos percorridos pelo usuário que indicam a estratégia de interação adotada e quais os problemas de usabilidade relacionados.

Apesar da funcionalidade da ferramenta FAIUnix ter sido inspirada na funcionalidade da ferramenta de suporte ao método *Playback* [NS 84], há uma diferença marcante no processo de coleta de dados. No método *Playback*, a idéia central era registrar a atividade do usuário durante uma interação com um sistema, em um segundo computador, em um ambiente de testes. A ferramenta FAIUnix é executada no mesmo computador no qual o protótipo está sendo executado, gravando nesta máquina os dados coletados da interação. Esta estratégia, permite que uma sessão de avaliação também seja realizada fora do ambiente controlado de um laboratório, sendo realizada no próprio ambiente de trabalho do usuário e dispensando a presença do avaliador, tornando os testes mais realistas e menos intrusivos. O avaliador necessita apenas preparar o arquivo de coleta antes da sessão iniciar. É também importante ressaltar, que embora o propósito da ferramenta tenha sido apoiar a avaliação de protótipos, é possível utilizá-la no contexto da avaliação de produtos que tenham sido desenvolvidos para o ambiente *Openlook*. Assim sendo, sua utilização pode ser estendida a outras fases do ciclo de vida do produto, incluindo testes de validação e a monitoração do uso do produto para efeito de manutenção.

Comparada ao *Observer* [Nol 99], a ferramenta FAIUnix apresenta muitas semelhanças quanto aos dados coletados (arquivo de *log*) e aos relatórios apresentados. O *Observer*, no entanto, é mais flexível quanto à configuração de coleta pois para cada sessão de observação pode ser definida uma configuração diferente, enquanto que na FAIUnix a configuração é fixa para todas as sessões. Uma outra funcionalidade encontrada no *Observer*, e inexistente na FAIUnix, é o registro da interação em vídeo.

A ferramenta para coleta de dados FAIUnix facilitou a análise de dados sobre a estrutura do projeto de interação, sobre a realização das tarefas e sobre os erros cometidos pelo usuário, permitindo a localização de problemas de usabilidade e consequentemente levando a uma melhora do projeto da interface.

A associação da verificação de modelos com a validação de protótipos em um mesmo método possibilitou investigar aspectos que podem ser corretos do ponto de vista da verificação dos modelos mas que podem não ser adequados do ponto de vista da usabilidade do produto. Por exemplo, do ponto de vista do modelo da interação, se para uma tarefa foi

projetado apenas um mecanismo de interação (um caminho de navegação), o modelo pode ser considerado correto, mas não apresenta flexibilidade para o usuário.

Por conta da tarefa planejada, alguns recursos do protótipo não foram utilizados, a exemplo de "Editar o conteúdo" de uma página. No entanto todos os demais recursos incorporados ao protótipo foram contemplados.

8.2 PROPOSTAS DE CONTINUIDADE

Com base na avaliação dos resultados, na qual são apresentadas limitações das soluções adotadas, esta seção apresenta propostas de continuidade para o trabalho.

A representação do modelo da tarefa, tanto na árvore quanto nos descritores, é repetitiva e sujeita a erros. A construção do modelo da interação, a exemplo do modelo da tarefa, também está sujeita a erros, particularmente no que diz respeito ao mapeamento entre os dois modelos. É sugerido o desenvolvimento de uma ferramenta computacional para apoiar tanto a construção do modelo da tarefa quanto o mapeamento para o modelo da interação.

A análise dos mecanismos de navegação presentes no modelo da interação também poderia ser facilitada com o uso de uma ferramenta de modo a localizar todos os caminhos projetados para realização de uma tarefa destacando os caminhos mais eficientes (mais curtos ou mais simples).

Apesar dos resultados obtidos com a utilização de redes de Petri na construção e análise do modelo de navegação terem sido positivos, uma das dificuldades encontradas foi a associação entre os estados da rede e os estados da interação. Apesar de ser possível fazer esta associação a partir da análise dos descritores das marcações, esta é uma tarefa extremamente trabalhosa dado o grande número de estados do modelo. É então sugerida a análise automática a partir de funções de análise desenvolvidas para este propósito. Esta informação é importante para associar a execução do modelo da navegação aos cenários correspondentes no modelo da interação.

O modelo de navegação não representa a ativação e desativação de objetos da interação. Sugere-se a extensão do modelo para incluir este aspecto de modo a viabilizar a

análise de bloqueios decorrentes de restrições operacionais. A extensão poderia consistir na inclusão de atributos nos objetos e nas expressões do modelo.

Um outro aspecto do método que poderia se beneficiar do uso de uma ferramenta seria a construção do protótipo a partir do modelo da interação.

A análise dos dados coletados pela ferramenta FAIUnix, poderia ser apoiada por diretrizes de projeto disponíveis para consulta *on-line*. É então sugerido um suporte para consulta a diretrizes de projeto, de uma forma sensível ao contexto da avaliação.

Sugere-se a incorporação ao método, da avaliação da satisfação do usuário de modo a complementar a avaliação da usabilidade.

No método apresentado o tempo é considerado apenas durante a validação do protótipo, no entanto para aumentar a abrangência dos resultados, o tempo deve ser considerado também no modelo da navegação. Desta forma seria possível estimar o tempo de realização da tarefa, já nesta fase do projeto.

A abrangência dos resultados deste trabalho foi restrita dado que testes de usabilidade com a comunidade de usuários deveriam ter sido realizados. Os testes realizados apesar de terem sidos baseados em um plano de testes careceram da aplicação a um número significativo de usuários [Jef 94]. No entanto, dado o tempo disponível para conclusão deste trabalho, não foi possível realizar esta etapa, sendo deixado como sugestão para os próximos trabalhos.

Referências Bibliográficas

[Bev 97] N. Bevan, *Quality and usability: A new framework*. vanVeenendaal, E, and McMullan, J (eds), Achieving software product quality, Tutein Nolthenius, Netherlands, 1997.

[Boe 88] B.W. Boehm, *A Spiral Model of Software Development and Enhancement*, IEEE Computer, 21(5), 61-72, 1988.

[BSF 99] Boulder Software Foundry, Inc.
homepage: http://www.bouldersoftware.com

[CSM 83] S. Card, T. Moran, A. Newell, *The Psychology of Human Computer Interaction*, Laurence Erlbaum Assoc., Hillsdale, NJ, 1983.

[Din 96] E. da S. A. Diniz. "*FAIWin - Ferramenta para Avaliação de Interfaces em Ambiente Windows, a partir da monitoração de dados*", Dissertação de Mestrado, DSC – UFPB, Agosto 1996, 111 p.

[Dru 97] Doron Drusinsky. *BetterState Pro Tutorial: An Itnroduction to Designing with StateCharts*. Integrated Systems, Inc. Printed in the United States of America, April 1997.

[Hel 91] D. Heller. *Xview Programming Manual*, Volume Seven, O'Reilly & Associates, Inc. Printed in the United States of America, 1991.

[HH 93] D. Hix and H. R. Hartson. *Developing User Interfaces*, Ensuring Usability through Product & Process, John Willey & Sons, Inc., 1993.

[HR 98] J. Hackos and J. Redish . *User and Task Analysis for Interface Design.* John Wiley & Sons, Inc., ISBN 0-471-17831-4, 1998

[Jef 94] Jeffrey, Rubin. Handbook of usability testing: how to plan, design, and conduct effective tests., John Wiley & Sons, Inc., ISBN 0-471-59403-2, 1994.

[JJ 91] H. Johnson and P. Johnson *Task Knowledge Structures: Psychological basis and integration into system design.* Department of Computer Science, University of London, http:// www.dcs.qmw.ac.uk

[Jen 92] K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*, Volume 1, Basic Concepts. Monographs in Theoretical Computer Science, Springer-Verlag, 1992.

[KC 93] J. Kirakowiski and M. Cobett, *SUMI: the software Usability Measurement Inventory*, B J Ed Tech, 24.3 210-214

[LP 98] A. Leceerof and F. Paternò, *Automatic Support for Usability Evaluation*, In IEEE Transactions on Software engineering, pp. 863-888, Vol. 24, No. 10, October 1998.

[MC 99] I. MacColl and D. Carrington, *User Interface Correctness.* Computer Science Department, University of Quensland, Australia. On-line version: http://www.it.uk.edu.au/~ianm/correct.html

[MR 93] M. Macleod and R. Rengger, *The development of DRUM: a software tool for video-assisted usability evaluation.* In JL Alty et al (eds.) People and Computers VIII (Proc. Of HCI'93 Conf. Loughborough UK, September 1993). Cambridg : CUP, 293-309.

[MS 97] R. Mahajan and B. Shneiderman, *Visual and Textual Consistency Checking Tools for Graphical User Interfaces.* In IEEE Transactions on Software Engineering, pp 722- 735, Vol. 23, No. 11, November 1997.

[MSC 93] Meta Software Corporation. *Design/CPN Reference Manual for X-Windows*, Version 2.0, Cambridge, MA, U.S.A., 1993.

[Mur 89] T. Murata. *Petri Nets: Properties, Analysis and Applications*. Proceedings of the IEEE., Vol. 77, No 4, April 1989.

[Nak 95] L. Y. Nakayama. "*FAI – Ferramenta de Avaliação de Interface com o usuário no Ambiente AGILE*", Dissertação de Mestrado, DEE-UFPB, Maio 1995, 108 p.

[Nol 99] Noldus Information Technology
homelage: http://www.noldus.com/products/observer/observer.html

[NS 84] A. S. Neal e R. M. Simons. Playback: *A method for evaluating the usability of software and its documentation*, IBM Systems Journal, vol. 23, No. 1, 1984.

[Nie 93] J. Nielsen. *Usability Engineering*, Academic Press/AP Professional, Cambridge, MA., 1993.

[Nie 96] J. Nielsen. *Usability Metrics: Tracking Interface Improvements*, In IEEE Software, pp. 12-13, November 1996.

[PB 94] P. Palanque & R. Bastide. *Theoretical Foundations of recent formal approaches in HCI design*. Research Symposium CHI'94. Boston, 23-30 April 1994.

[PB 96] P. Palanque, R. Bastide. *A design life-cycle for the formal design of interactive systems*. BCS-FACS Workshop on the Formal Aspects of the Human-Computer Interface, FAHCI'96, Sheffield, U.K., September 1996.

[PBS 95] P. Palanque, R. Bastide, V. Senges. *Validating interactive system design through the verification of formal task and system models*. 6th IFIP Working Conference on Engineering for Human-Computer Interaction (EHCI'95) Grand Targhee Resort, Wyoming, U.S.A. 14-18 August 1995.

[PBSBD 93] P. Palanque, R. Bastide, C. Sibertin-Blanc, and L. Dourte. *"Design of User-Driven Interfaces using Petri Nets and Objects"*. In F. Bodard, C. Rolland, and C. Cauvet, editors, Conference on Advanced Information System Engineering CAISE'93, Springer Verlag, LNCS n.685, Paris (France), June 1993.

[Pre 95] R. Pressman, *Engenharia de Software*; Tradução José C. B. dos Santos; revisão José Carlos Maldonado, Paulo C. Masiero, Rosely Sanches – São Paulo; Makron Books do Brasil, 1995.

[Pro 92] C. D. Procópio. "*Desenvolvimento do Gerenciador de Diálogos e de Ferramentas de Prototipagem do Sistema AGILE*", Dissertação de Mestrado, DSC – UFPB, Abril 1992, 123 p.

[QT 97] J. E. R. Queiroz e M. F. Q. V. Turnell. *Proposta de Abordagem para Seleção de Estratégias Avaliatórias de Processos Interativos*. Anais da XXIII Conferência Latino Americana de Informática – CLEI'97, Valparaíso, Chile, pp. 673-682, Nov. 1997.

[Rei 81] P. Reisner, *Formal Grammar and Design of an Interactive System*, IEEE Transactions on Software Engineering, SE-5, 229-240, 1981.

[RPC 96] F. de Rosis, S. Pizzutilo, B. De Carolis *A Tool to Support Specification and Evaluation of Context-Customized Interfaces,* SIGCHI Bulletin, Vol. 28, No. 3, July 1996.

[RPC 98] F. de Rosis, S. Pizzutilo, B. De Carolis. *Formal description and evaluation of user-adapted interfaces*, Int. J. Human-computer Studies (1998) 49, 95-120.

[San 92] R. Santos. "*Módulos Gerenciador de Telas e Gerenciador de Periféricos de um sistema de Prototipagem rápida usuário-computador*", Dissertação de Mestrado, DEE – UFPB, Abril 1992, 131 p.

[SB 92] D. Scapin and J. M. C. Bastien. *A validation of Ergonomic Criteria for the evaluation of Human-Computer Interfaces*. International Journal of Human-Computer Interaction, 1992, pp. 183-196.

[Sch 95] E. Schlungbaum and T. Elwert, Tadeus – *a model-based approach to the Development of Interactive Software Systems*, In. Rostocker Informatik-Berichte, pp. 93-104, (1995) 17.

[SE 96] E. Schlungbaum and T. Elwert: *Dialogue graphs - a formal and visual specification technique for dialogue modelling*. In: Proceedings of BCS-Workshop: Formal Aspects of the Human Computer Interface, 1996.

Figura A-2: Tarefa Consultar Página

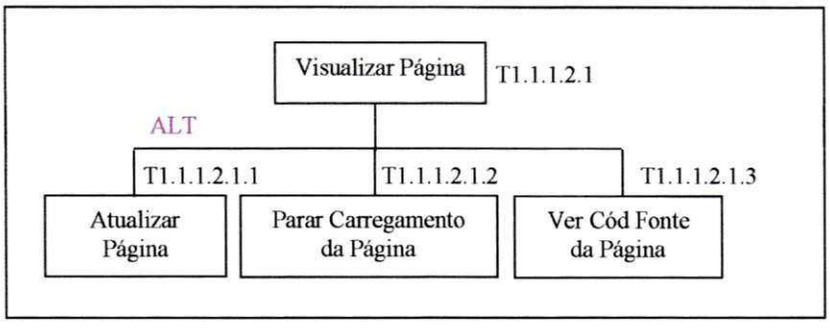

Figura A-3: Tarefa Visualizar Página

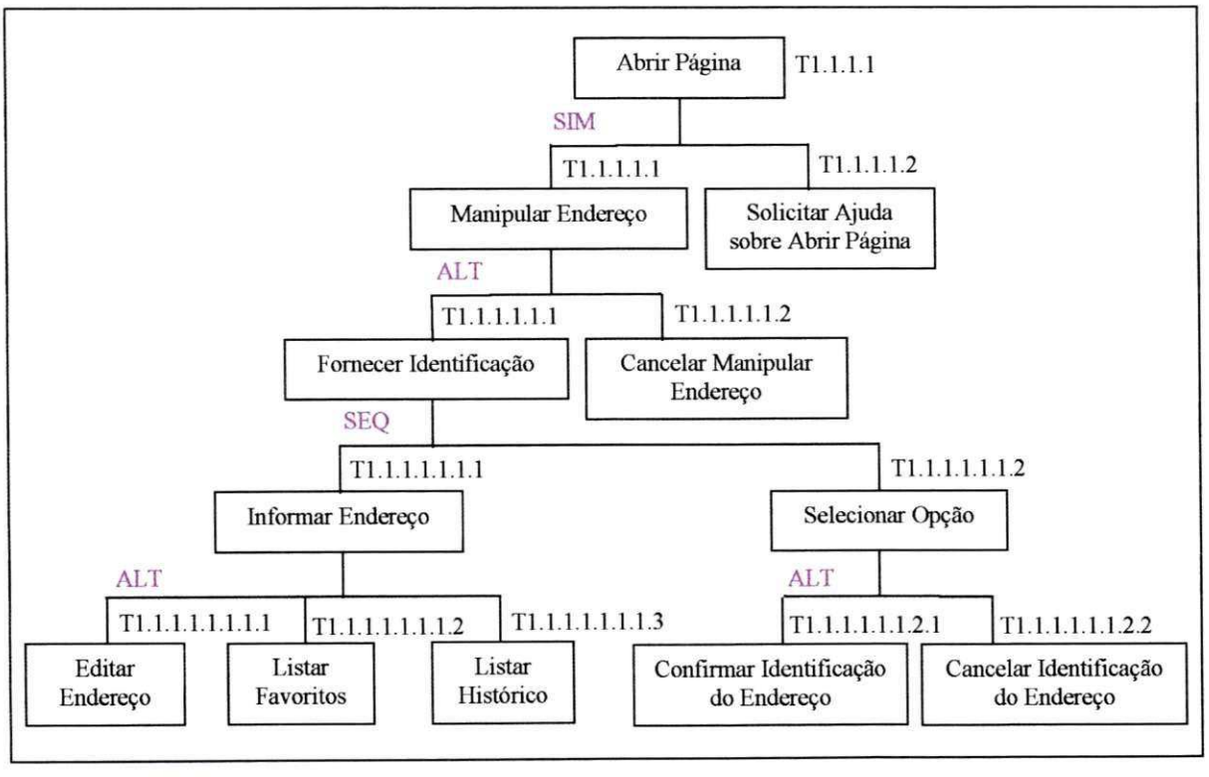

Figura A-4: Tarefa Abrir Página

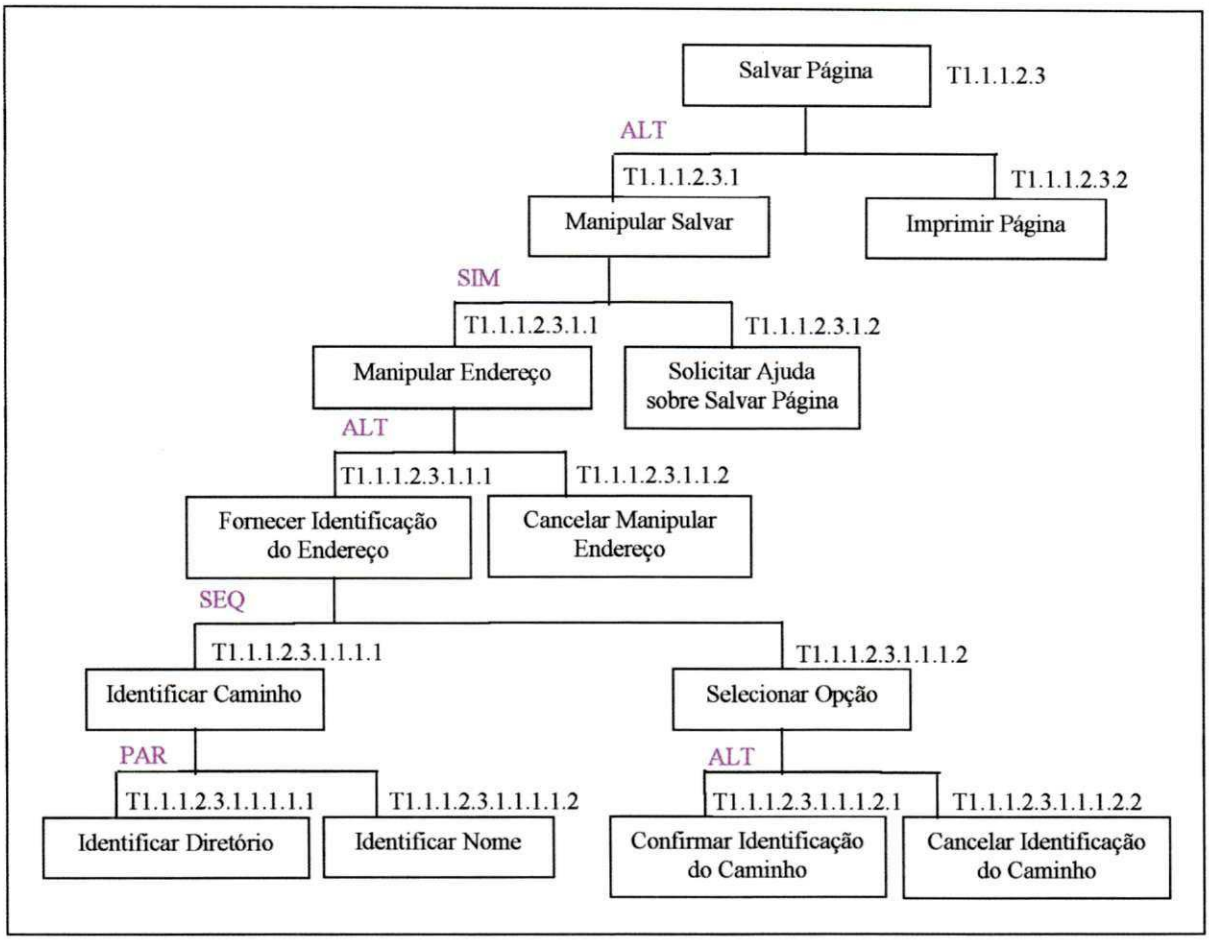

Figura A-5: Tarefa Salvar Página

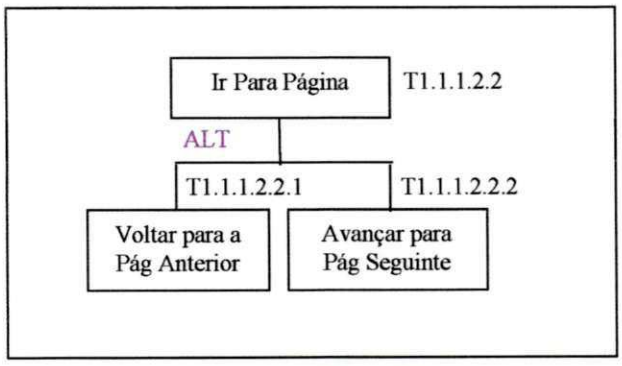

Figura A-6: Tarefa Ir Para Página

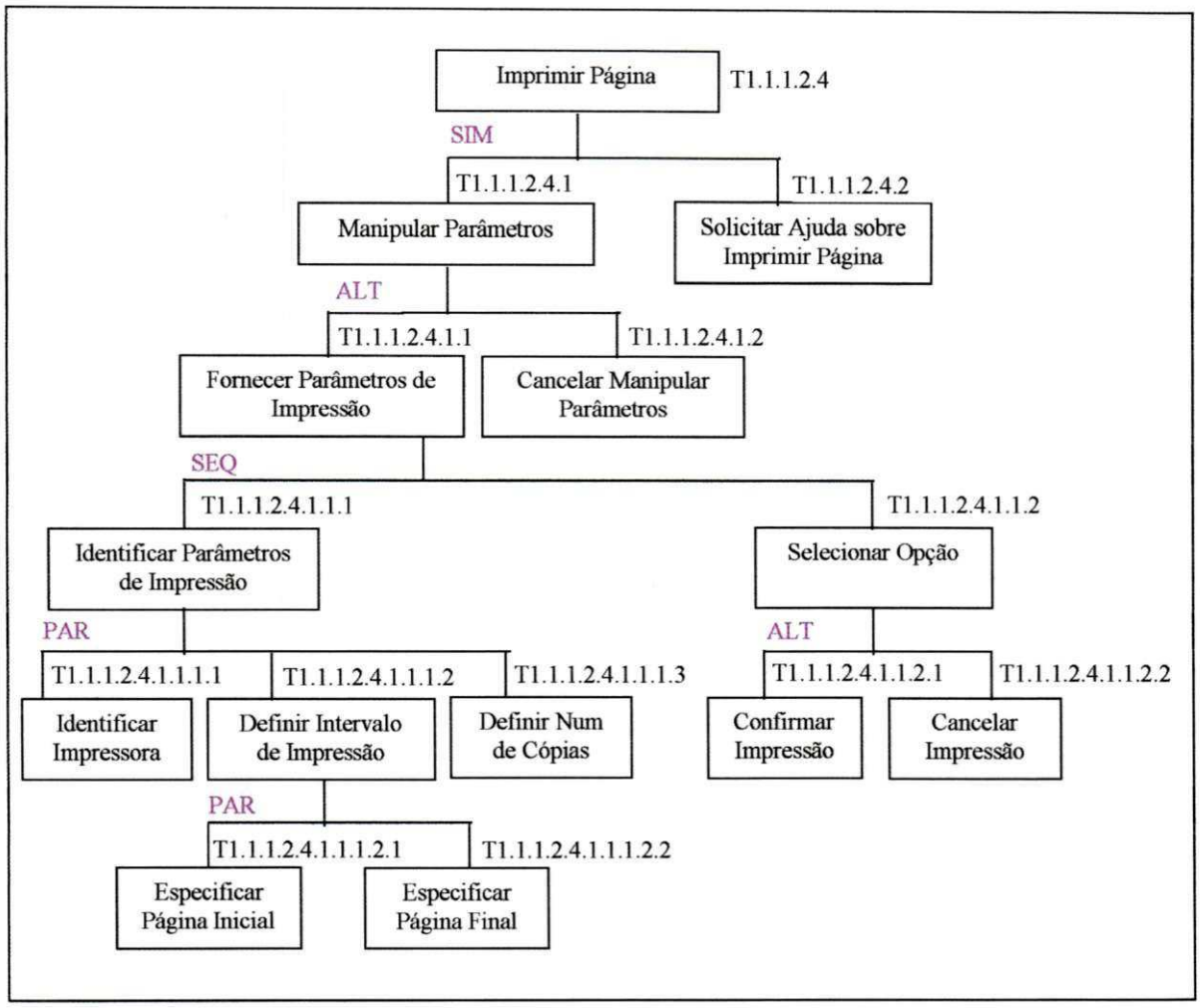

Figura A-7: Tarefa Imprimir Página

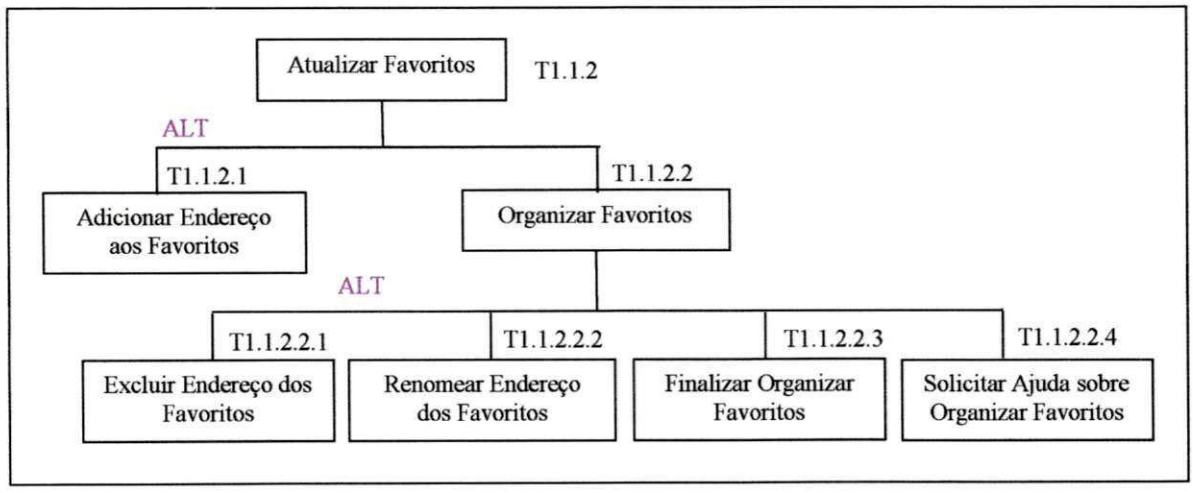

Figura A-8: Tarefa Atualizar Favoritos

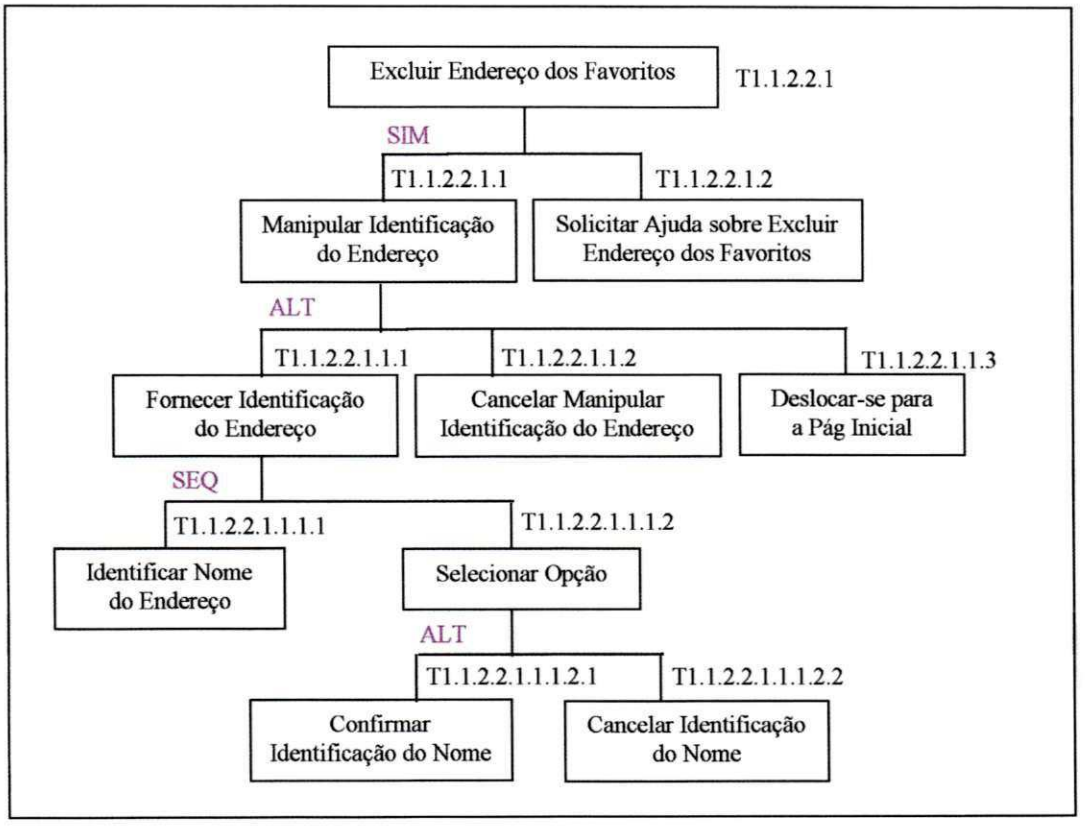

Figura A-9: Tarefa Excluir Endereço dos Favoritos

Coleta estática. Coleta de dados relativos aos aspectos estruturais da interface, a exemplo de relação de cores utilizadas nas janelas, o número total de cores em cada janela, o total de objetos em cada janela.

CPN (*Coloured Petri Nets*). É uma classe de rede de Petri de alto nível adotada na construção do modelo que representa as características das interfaces.

Densidade dos objetos na janela (*Widget Density*). Razão entre o número de objetos presentes na janela, no nível mais alto da interação, e a área total da janela. Para efeito de normalização esta razão é multiplicada por 100.000.

Descritor da tarefa. Estrutura em *frame* que representa uma tarefa a partir dos elementos que a compõem.

Design/CPN. Ferramenta de suporte a construção e análise de redes de Petri coloridas que possibilita criar, modificar, organizar, executar, depurar, examinar e validar as CPNs.

DevGuide. Ferramenta para prototipação de interfaces para o ambiente Unix.

Distribuição dos objetos na janela (*Balance Area Ratios*). É a medida do equilíbrio da distribuição de objetos na janela.

DRUM. Uma ferramenta de suporte ao projeto MUSiC, que oferece recursos para coleta e análise de dados em vídeo.

Estado da interação. Representa um contexto da interação com informações sobre o estado atual da interação e um histórico da navegação contendo informações sobre os estados anteriores.

Estado da interface. Ver estado da interação.

FAIDOS. Ferramenta de suporte a avaliação de interfaces desenvolvidas para o ambiente MS-DOS.

FAIWin. Ferramenta de suporte a avaliação de interfaces desenvolvidas para o ambiente *Windows*.

FAIUnix. Ferramenta de suporte a avaliação de interfaces desenvolvidas para o ambiente Unix.

Ferramentas FAI. Conjunto de ferramentas para suporte a avaliação de interfaces, desenvolvidas no âmbito da Universidade Federal da Paraíba.

GOMS. (*Goals, Operators, Methods and Selection Rules).* Caracterização da interação do usuário com um sistema, como uma atividade orientada por objetivos, na qual o menor caminho para atingir um objetivo é considerado o melhor.

Gxv. Ferramenta para geração de código fonte em linguagem C, a partir de um arquivo de protótipo gerado pelo *DevGuide*.

ISO (*International Standards Organization*). As normas ISO estão disponíveis para comercialização no '*ISO Catalogue*'.

Janela. Espaço na tela utilizado para entrada e saída de informações.

MAD (Método Analítico de Descrição - *Méthode Analytique de Desctiption*). Método utilizado na descrição de tarefas. Este método se baseia em objetos aos quais estão associados estados, objetivos, ações e condições.

Método de avaliação baseado no desempenho do usuário enquanto utiliza a interface. Gera grandes volumes de informação tanto qualitativa quanto quantitativa que vão desde a opinião do usuário sobre o produto até o registro automático, ou não, de dados da interação.

Modelo da interação. Modelo que representa a interação construído a partir do conhecimento sobre o perfil do usuário e do modelo da tarefa. Este modelo resulta do mapeamento das funções necessárias levantadas no modelo da tarefa para uma estratégia de navegação, mecanismos de interação, e os objetos com os quais o usuário deverá interagir.

Modelo da navegação. Modelo que representa os objetos de interação e o seu comportamento do ponto de vista de navegação entre janelas da interface.

Modelo da tarefa. Modelo que representa o conhecimento sobre a tarefa.

MUSiC (*Metrics for Usability Standards in Computing*). Conjunto de métodos propostos para medir a usabilidade de sistemas. Estes métodos se destinam à especificação e avaliação da qualidade no uso e se fundamentam na realimentação destas informações sobre o projeto do sistema.

Navegação. Corresponde a passagem do controle de uma janela para outra na interface.

Objetos da interface. Conjunto de elementos que possibilitam a interação do usuário com a interface, a exemplo de botões e menus.

Observer. Sistema desenvolvido para coletar, analisar, apresentar e gerenciar dados oriundos de procedimentos de observação do uso de um produto. Pode ser utilizado para registrar atividades, atitudes, movimentos, expressões faciais, interações sociais e qualquer outro aspecto do comportamento humano.

OpenLook. Padrão de interface do *OpenWindows*.

OpenWindows. Ambiente de janelas que utiliza interface gráfica no sistema operacional Unix.

Percentual de Ocupação da janela (*Nonwidget Area*). Razão entre a área não ocupada e a área total de uma janela, expressa em porcentagem.

Perfil do usuário. Conjunto de características sobre a população de usuários, relevante para a interação. Este perfil pode ser obtido a partir da observação do usuário em seu ambiente de trabalho, se possível realizando tarefas similares, ou ainda a partir de questionários ou entrevistas com representantes do grupo de usuários.

Processo iterativo. Processo que se repete até que soluções adequadas sejam alcançadas. No contexto do trabalho, o processo iterativo termina quando os objetivos de usabilidade propostos são alcançados no projeto.

Projeto conceitual. Projeto no domínio dos objetos e tarefas

Proporções da janela (*Aspect Ratio*). Razão entre a altura e a largura de uma janela.

PUM (*Programable User Model*). Modelo executável que simula o comportamento humano durante a interação com uma máquina ou sistema.

QC/Replay. *Software* comercial que registra a interação do usuário com objetos da interface (botões, menus, etc.) e suas ações elementares (pressionamento do mouse, de teclas, etc.).

Qualidade da interação. Conjunto de propriedades que devem ser satisfeitas para que possamos considerar que uma interface satisfaz as necessidades de seus usuários.

Sessão de avaliação. Tempo decorrido entre o início e o fim de um período durante o qual o usuário realiza uma interação com a interface com o objetivo de avaliação.

Sherlock. Ferramenta utilizada na análise de consistência, apoiando a avaliação da distribuição espacial de objetos na tela e da terminologia utilizada em diálogos entre o usuário e uma interface.

SUMI. Questionário para avaliação da satisfação do usuário utilizado no projeto MUSiC.

TAG (*Task Action Grammars*). Gramática que se propõe a reduzir a complexidade da semântica da execução de uma tarefa.

Tarefa. Atividade do usuário em um domínio específico de aplicação

TKS (*Task Knowledge Structures*). Método utilizado para identificar, analisar e modelar a estrutura de uma tarefa, baseado na representação do conhecimentos necessário para o usuário realizar uma tarefa.

UAN (*User Action Notation*). Notação orientada à tarefa e ao usuário, que descreve o comportamento do usuário e do sistema durante a realização de uma tarefa.

USINE (*USer INterface Evaluator*). Ferramenta de coleta de dados de uma sessão de avaliação, que possibilita a associação entre ações do usuário no modelo da tarefa, com ações do usuário em um arquivo de *log*.

Usuário final. Usuário para o qual um sistema é desenvolvido.

Usuário projetista. Usuário de um sistema de suporte a concepção e desenvolvimento de sistemas.

Usabilidade. Expressa pela facilidade de aprendizado, compreensão e operação de um sistema pelo usuário.

XVT. Sistema voltado para o desenvolvimento e avaliação de interfaces, composto por ferramentas multi-plataformas (*Microsoft Windows*, *Macinhtosh*, *OS/2*, *OSF/Motif*).