UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
COORDENAÇÃO DOS CURSOS DE
PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Dissertação de Mestrado

Aplicação de um Modelo de Navegação de IHM ao Contexto de Sistemas Industriais

Alexandre Scaico

Campina Grande – PB
Agosto - 2001

Aplicação de um Modelo de Navegação de IHM ao Contexto de Sistemas Industriais

Alexandre Scaico

Dissertação de mestrado submetida à Coordenação dos Cursos de Pós-Graduação em Engenharia Elétrica da Universidade Federal da Paraíba – Campus II como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências no domínio da Engenharia Elétrica.

Área de Concentração
Processamento da Informação

Orientadora
Maria de Fátima Queiroz Vieira Turnell, PhD

Campina Grande, Paraíba
Agosto de 2001

S278a Scaico, Alexandre
Aplicacao de um modelo de navegacao de IHM ao contexto de sistemas industriais / Alexandre Scaico. - Campina Grande, 2001.
88 f.

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

1. Interfaces com o Usuario 2. Redes de Petri 3. Sistemas Industriais 4. Dissertacao - Engenharia Eletrica I. Turnell, Maria de Fatima Queiroz Vieira II. Universidade Federal da Paraiba - Campina Grande (PB) III. Título

CDU 681.3.02(043)

APLICAÇÃO DE UM MODELO GENÉRICO DE NAVEGAÇÃO DE IHM AO CONTEXTO DE SISTEMAS INDUSTRIAIS

ALEXANDRE SCAICO

Dissertação Aprovada em 31.08.2001

Maria de Fatima Q. Vieira Turnell

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

PROF. ÀNGELO PERKUSICH, D.Sc., UFPB
Componente da Banca

Jorge Cesar Abrantes de

PROF. JORGE CESAR ABRANTES DE FIGUEIREDO, D.Sc., UFPB
Componente da Banca

CAMPINA GRANDE - PB
Agosto - 2001

Resumo

Este trabalho apresenta e discute a aplicação de um modelo de navegação em interfaces com usuário. Este modelo, construído em redes de Petri Coloridas (CPN), apresenta uma estrutura genérica concebida para apoiar a análise da usabilidade do componente de navegação no projeto de interface homem-máquina. O contexto de aplicação do modelo é o de sistemas de controle supervisório para plantas industriais.

O estudo se desenvolveu em duas etapas. Na primeira foi utilizado um modelo originalmente proposto em [40,41], adequando-o ao contexto das interfaces industriais. Na segunda etapa o modelo foi modificado de modo a representar o comportamento temporal da navegação nestes sistemas tais como as restrições temporais no tratamento de alarmes.

Do trabalho conclui-se que o modelo de navegação adotado se adequou aos propósitos do estudo e ao contexto da interface de sistemas industriais. Conclui-se também que a utilização deste modelo é valiosa na análise da usabilidade de sistemas, do ponto de vista da navegação.

Palavras chave: interfaces com o usuário, redes de Petri, sistemas industriais.

Abstract

This work presents and discusses the application of a navigation model to support the analysis of computer-user interfaces. This model, built in Colored Petri Nets (CPN), presents a generic structure conceived in order to support the usability evaluation of the user interfaces navigation component. The application context is that of supervisory control systems for industrial plants.

The study was developed in two phases. In the first phase, a model originally proposed in [40, 41], was used to represent an industrial plant, having been adjusted to this new context. In the second phase, the model was extended to represent the temporal behavior of the navigation component in order to represent actions with temporal restrictions such as alarm treatment.

Do trabalho conclui-se que o modelo de navegação adotado se adequou aos propósitos do estudo e ao contexto da interface de sistemas industriais. Conclui-se também que a utilização deste modelo é valiosa na análise da usabilidade de sistemas, do ponto de vista da navegação.

From this work it was concluded that the navigation model, is adequate for the navigation analysis as well as to the industrial context, and that the proposed extension made feasible the representation of the industrial system's interface behavior. It was also concluded that the use of the model approach is valuable in the analysis of the usability of these systems, from the viewpoint of navigation.

Key words: user interface, Petri nets, industrial systems.

Agradecimentos

Agradeço a profª Maria de Fátima Q. V. Turnell pela amizade, por ter acreditado e incentivado este trabalho, e por ter me mostrado o verdadeiro significado da palavra orientação.

Sou grato a meus pais, Bárbara Simonetti e Ademar Maculan, pelo amor, apoio, carinho e compreensão que eles sempre me deram. As minhas irmãs, Babi e Bebel, pelos momentos de alegria que elas proporcionam sempre.

Agradeço a minha namorada, Leniedva de Lima Amorim, pelo amor que ela me dá e pela sua paciência e compreensão nos momentos difíceis deste trabalho.

Aos amigos e colegas do mestrado, sempre presentes para incentivar e ajudar. Aos professores, que estiveram sempre presentes para ajudar a transpor as dificuldades que surgiram neste trabalho.

Aos meus vizinhos, pelo apoio e pelos grandes momentos de descontração.

Sumário

Lista de Figuras

Capítulo 1 – Introdução

A percepção da importância da qualidade das interfaces com o usuário nos processos interativos homem-máquina vêm aumentando consideravelmente nos últimos anos, e com isso muitos projetistas de sistemas interativos têm buscado melhorar o projeto deste componente.

A interação entre o usuário e o sistema computacional é facilitada pela interface, a qual tem como principal objetivo proporcionar um acesso "transparente" aos recursos que o sistema disponibiliza. Essa interação deve ser agradável, eficiente e de fácil aprendizado, de forma a reduzir o tempo de realização das tarefas, bem como o número de erros cometidos pelo usuário.

No projeto de uma interface homem-máquina deve-se levar em conta as necessidades do usuário que utilizará o sistema e observar os fatores ergonômicos, tais como a clareza na apresentação das informações e a facilidade de navegação, dentre outros.

Outro aspecto a ser considerado é a carga cognitiva (memorização, reconhecimento e decisão) exigida para operação do sistema, a qual deve ser reduzida ao máximo. Também são fatores importantes na interação, o tempo de resposta do sistema para o usuário, assim como do usuário para o sistema.

1.1 Ambiente Industrial

No ambiente industrial, particularmente na automação do controle de processos, os programas supervisórios aumentam a segurança do trabalho, os níveis de produtividade e o controle da qualidade no processo de produção; além de permitir um gerenciamento mais eficaz do sistema.

Nestes sistemas o componente interface com o usuário é um aspecto crítico para seu bom funcionamento. Os programas supervisórios pertencem à classe dos sistemas em

tempo real. Assim, o tempo gasto na realização das ações deve ser bem dimensionado. Tipicamente, muitas tarefas possuem prazos estritos os quais devem ser cumpridos, de modo a não comprometer o processo sob controle nem as condições de segurança do sistema.

Portanto, as propriedades temporais da interação podem determinar a usabilidade do sistema. Uma grande latência na resposta do sistema ao usuário pode aumentar a taxa de falhas. Lentidão na resposta do usuário ao sistema também aumenta a taxa de falhas uma vez que os prazos estritos de muitas tarefas podem ser perdidos. As falhas nestes sistemas são críticas dado o tipo de atividade que está sendo executada, a exemplo do controle de uma planta industrial. Atrasos na resposta do usuário a certos eventos (como a resposta a ocorrência de um alarme) podem resultar em situações catastróficas (perda de produção, quebra de máquinas e até risco de vida das pessoas envolvidas no processo).

O usuário de um programa supervisório (operador da planta) necessita de acesso em tempo hábil à informação devendo também agir de modo inequívoco e em tempo hábil sobre o sistema. Uma interface confusa pode levar a uma interpretação errada das informações por parte do operador ocasionando erros na tomada de decisões cujas conseqüências podem ser graves.

Deve-se também observar que a produtividade do sistema sob controle está diretamente ligada à qualidade da interação. Portanto, as interfaces devem ser projetadas de modo a prover ao operador um caminho (ou cenário) otimizado para a realização das tarefas. Existindo esse tipo de preocupação no projeto, muitos dos erros ocasionados por dificuldade na interpretação das informações ou por perda de prazos na execução de ações podem ser eliminados.

A partir da literatura consultada [9,10,45,46] observa-se que para os sistemas industriais muitos estudos são realizados sob o processo, tentando melhorá-lo do ponto de vista funcional; mas pouco se investiga a interface. Os projetistas não se aprofundam na escolha do estilo de interação mais adequado ou da melhor forma de apresentar as informações, etc.

Devido a essa falta de preocupação com o projeto da interface os programas supervisórios levam a uma baixa qualidade da interação. E, embora o componente de controle possa ter sido bem estruturado, se a interação for deficiente o usuário não conseguirá aproveitar os recursos oferecidos pelo sistema. Por exemplo, a escolha de

ícones inadequados para representar objetos do processo, demanda mais tempo do usuário para reconhece-lo e agir sobre ele.

1.2 Foco do Trabalho

Este trabalho tem como foco a avaliação da qualidade das interfaces homem-máquina de sistemas industriais visando a sua otimização do ponto de vista da facilidade de uso e eficácia da navegação.

A avaliação da usabilidade de uma interface pode ser realizada de diferentes pontos de vista. O estudo pode estar voltado para a documentação, o projeto visual (utilização de cores, tipos de fontes utilizados), a disposição dos mecanismos de interação, os mecanismos de ajuda, etc.

Neste trabalho o objeto de estudo para a avaliação da qualidade das interfaces é o componente de navegação. Através da análise e adequação ergonômica dos mecanismos de navegação pretende-se aumentar a produtividade de um sistema controlado por um programa supervisório de um ambiente industrial.

1.3 Motivação

Em um sistema computacional interativo o componente de navegação exerce uma grande influência sobre a qualidade da interação. No ambiente industrial, dadas as questões de produtividade e segurança, o componente de navegação torna-se ainda mais importante.

Apesar da vasta literatura sobre interfaces, não foram encontrados trabalhos que focalizassem o estudo das interfaces de ambientes industriais automatizados. Muitos dos estudos localizados utilizam a modelagem da interface utilizando métodos formais [4,22,28,36], porém os modelos apresentados não abordam o componente de navegação da interface de forma explícita, além de serem específicos para um contexto particular de aplicação. Também não foi encontrada a relação entre o estudo dos ambientes industriais automatizados e das interfaces dos programas que controlam estes ambientes.

Uma vez que muitos trabalhos são baseados em métodos formais, os modelos resultantes são difíceis de serem aplicados por projetistas e desenvolvedores de interfaces na indústria que não possuem formação em métodos formais. Para que passem a ser

utilizados, é necessário que os modelos sejam de simples aplicação e os métodos de análise propostos sejam facilmente compreendidos.

Finalmente, um problema que surge na modelagem de interface é a necessidade de representar um elevado número de objetos e ações, o que pode tornar um modelo muito extenso, dificultando sua análise.

1.4 Metodologia Utilizada

O uso de métodos formais é uma estratégia que permite verificar matematicamente propriedades importantes acerca do sistema modelado.

Dentre os métodos formais foi escolhido redes de Petri para a realização deste trabalho. A escolha de redes de Petri se deu por dois motivos. Por possibilitar descrever paralelismo, concorrência, assincronismo e não-determinismo, que são características típicas de interfaces gráficas; além de permitir especificar a evolução dos estados do sistema em resposta as ações do usuário. Existem na literatura vários trabalhos de modelagem de interfaces baseados no uso de redes de Petri os quais comprovam a sua adequação à modelagem de interfaces [4,22,25,30,36].

O segundo motivo é que este trabalho tem como base um modelo de navegação proposto em [40,41], o qual foi construído utilizando em redes de Petri coloridas. O modelo proposto em [40,41] é uma representação genérica da navegação através da interface em sistemas interativos, com o objetivo de sintetizar, independentemente dos dispositivos de interação, as situações de transição mais comuns em uma interface com o usuário.

1.5 Objetivos

O objetivo principal deste trabalho é aumentar a usabilidade das interfaces industriais do ponto de vista da otimização do componente de navegação, através de sua modelagem e análise.

Como objetivo específico é proposta a aplicação do modelo de navegação para representar as interfaces de programas supervisórios industriais, de modo que possa ser mais facilmente utilizado por desenvolvedores de interface que não tenham formação em métodos formais.

1.6 Organização do Documento

Este documento está organizado em oito capítulos, sendo este o primeiro.

No Capítulo 2 são apresentados os conceitos básicos do formalismo das redes de Petri, adotado no processo de modelagem e análise ao longo do desenvolvimento deste trabalho.

No Capítulo 3 são abordadas as interfaces homem-máquina, focalizando no contexto de aplicação de programas supervisórios industriais.

No Capítulo 4 são descritos os métodos de avaliação de interfaces destacando a avaliação por meio de métodos formais. Neste capítulo é feita uma revisão da literatura sobre trabalhos de avaliação de interfaces baseados em métodos formais, e que utilizam o formalismo das redes de Petri.

No Capítulo 5 é descrito o modelo de navegação adotado para o processo de modelagem deste trabalho. Neste capítulo é também caracterizada a interface do ambiente industrial modelado.

O Capítulo 6 contém a descrição do modelo da interface industrial construído ainda sem levar em conta restrições temporais, e as análises realizadas para verificar sua corretude.

No Capítulo 7 é descrito o modelo da interface industrial construído levando em conta as restrições temporais e o tratamento dos alarmes. Nele, são também apresentadas as análises realizadas sobre o modelo.

Finalizando, no Capítulo 8, são discutidos os resultados obtidos neste trabalho, tecidas considerações sobre o alcance dos resultados e, apresentadas propostas de continuidade.

Capítulo 2 - Redes de Petri

2.1 Introdução

Neste capítulo são apresentados os conceitos relativos ao formalismo das redes de Petri, utilizados na modelagem da interface homem-máquina que serviu como base para este trabalho, bem como nos modelos que foram desenvolvidos no decorrer deste trabalho.

Serão apresentados os conceitos de modelagem utilizando métodos formais, redes de Petri Lugar-Transição, redes de Petri Coloridas, redes de Petri Coloridas hierárquicas, redes de Petri Coloridas temporizadas e os métodos de análise de redes de Petri.

2.2 Métodos Formais

Em muitas situações, um fenômeno ou sistema não é estudado diretamente, mas de uma forma indireta através de sua modelagem. Um modelo é uma representação, usualmente matemática, das características consideradas importantes no sistema sob estudo. A modelagem é muito útil por permitir um estudo do sistema sem o custo ou inconveniente de ter que trabalhar diretamente com o sistema [6].

O uso de métodos formais na modelagem de sistemas é muito útil por se basear no uso de notações com semântica precisa que possibilita a discussão de propriedades e a predição do desempenho do sistema através de verificação matemática [25,31]. Caso o modelo não se comporte como o desejado ou não atenda às propriedades desejadas, deve-se retrabalhar o modelo até que todas as falhas sejam resolvidas.

2.3 Redes de Petri

O conceito das redes de Petri teve origem na tese de Doutorado "*Kommunikation mit Automaten*" de Carl Adam Petri, na Alemanha em 1962 [6]. As redes de Petri são

ferramentas matemáticas e gráficas de modelagem aplicáveis a diversos tipos de sistemas [23]. Através das redes de Petri podemos descrever sistemas concorrentes, assíncronos, distribuídos, paralelos, não-determinísticos e/ou estocásticos.

Uma rede de Petri é composta de um grafo e de um estado inicial (chamado de marcação inicial, M_o). Este grafo é bipartido, direcionado e ponderado. Os nós deste grafo são denominados *lugares* e *transições*. Os nós são ligados através de arcos, observando que um arco deve ligar uma transição a um lugar ou um lugar a uma transição (não sendo permitida a ligação entre nós do mesmo tipo). Aos arcos se associam pesos (números inteiros positivos) como inscrições, onde um arco com peso K pode ser interpretado como um conjunto de K arcos paralelos entre dois nós [23]. Normalmente omite-se a inscrição de arcos de peso unitário.

O grafo define a estrutura da rede de Petri (sua natureza sintática). É necessário ainda expressar a dinâmica do sistema modelado, através de seus estados e evoluções entre estados. Para isso são acrescidas marcas a rede, chamadas de fichas, para definir o estado do sistema [35]. Apenas os lugares da rede podem conter fichas, que pode assumir qualquer número inteiro não-negativo. O número total de fichas em um lugar é denominado marcação do lugar, *M(p)*. A marcação da rede é um vetor *M*, contendo *m* elementos, onde *m* indica o número total de lugares na rede. Os elementos de *M* são as marcações dos *m* lugares da rede, onde *M(i)* indica a marcação do lugar *i*. M_o denota a marcação inicial da rede. A evolução das fichas (marcação) nos lugares representa os vários estados do sistema [6,23].

Formalmente, as redes de Petri são definidas como uma 5-upla PN = (P,T,F,W,M_o), onde [23]:

1. $P = \{p_1, p_2, ..., p_m\}$ é um conjunto finito de lugares
2. $T = \{t_1, t_2, ..., t_n\}$ é um conjunto finito de transições
3. $F \subseteq (P \times T) \cup (T \times P)$ é um conjunto de transições
4. $W: F \rightarrow N^*$ é uma função peso
5. $M_0: P \rightarrow N$ é a marcação inicial
6. $P \cap T = \phi$ e $P \cup T \neq \phi$

Na modelagem, os lugares representam as condições para a ocorrência de um evento ou o estado do sistema, e as transições os eventos do sistema. Para cada transição da rede existe um número de lugares com arcos direcionados dos lugares para a transição (lugares de entrada) e um número de lugares com arcos direcionados da transição para os

lugares (lugares de saída). Com isso, para um evento ocorrer (indicando a evolução do sistema através da ocorrência de uma transição) mudando a marcação da rede, a seguinte regra de transição (também chamada de regra de disparo) deve ser seguida [23]:

1. (Pré-condição) Uma transição *t* está habilitada (passível de execução) se cada lugar de entrada *p* de *t* contiver pelo menos *w(p,t)* fichas, onde *w(p,t)* é o peso do arco de *p* para *t*.
2. Uma transição habilitada pode ou não disparar (ocorrer).
3. (Pós-Condição) Ocorrendo o disparo de *t*, *w(p,t)* fichas são retiradas de cada lugar de entrada *p* de *t*, e são adicionadas *w(t,p)* fichas a cada lugar de saída *p* de *t*, onde *w(t,p)* é o peso do arco de *t* para *p*.

Nem sempre uma transição necessita ter lugares de entrada. Neste caso estas transições são denominadas de transições fonte, pois não necessitam de nenhuma pré-condição para a sua ocorrência, estando sempre habilitadas. O mesmo pode ser afirmado em relação aos lugares de saída, e neste caso temos transições sorvedouro, que em sua ocorrência apenas consomem fichas.

Graficamente os lugares são representados por círculos, as transições por retângulos e a fichas por pontos pretos. É comum, de uma maneira informal, referir-se a representação gráfica de uma rede de Petri como se fosse a própria rede de Petri [6]. A seguir temos um exemplo gráfico de uma Rede de Petri e sua evolução:

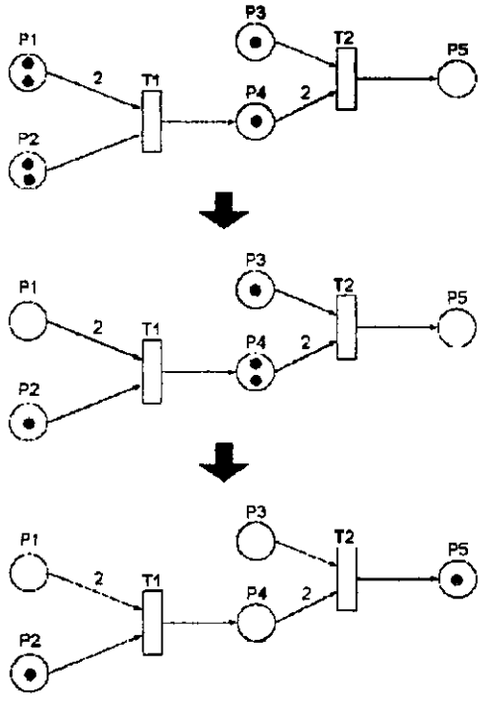


Figura 1 - Exemplo de rede de Petri e sua evolução

Na rede da Figura 2 inicialmente apenas T1 está habilitada, pois seus lugares de entrada possuem pelo menos o número de fichas necessárias a sua habilitação. T2 não está habilitada, pois é necessário pelo menos duas fichas em P4 para habilitá-la. Com o disparo de T1 duas fichas são removidas de P1 e uma de P2, e uma ficha é depositada em P4 (como pode ser observado na segunda rede da figura). Com isso T2 se torna habilitada, podendo disparar retirando uma ficha de P3 e duas de P4, e depositando uma ficha em P5, como pode ser observado na figura.

As redes de Petri aqui apresentadas são denominadas redes de Petri Lugar-Transição.

2.4 Extensões das Redes de Petri

As redes de Petri definidas na seção anterior não se mostram adequadas para a modelagem de muitos sistemas, devido à complexidade destes sistemas, o que implicaria em redes de Petri lugar-transição complexas e muito extensas (com um número muito grande de nós e arcos); além de não serem capazes de modelar aspectos temporais relacionados ao sistema, impossibilitando uma análise quantitativa [6]. Para contornar estes problemas foram propostas extensões ao modelo original, das quais as mais relevantes são as redes de Petri de alto nível e as extensões temporais das redes de Petri lugar-transição.

Como este trabalho foi realizado utilizando as redes de Petri Coloridas (CPN) [19,20], vamos detalhar apenas esta extensão das redes de Petri.

2.5 Redes de Petri Coloridas (CPN)

As redes de Petri coloridas resultam em uma representação mais compacta em relação as redes lugar-transição devido as fichas carregarem valores de dados, que podem ser complexos (ex: uma lista onde o primeiro elemento é um inteiro, o segundo é uma *string*, o terceiro é um par de inteiros), denominados *cores das fichas*. Cada lugar da rede deve possuir fichas cujas cores pertençam a um tipo específico, sendo este tipo denominado *conjunto de cores* do lugar. É necessário portanto (similarmente ao caso das linguagens de programação) declarar todos os *conjuntos de cores* (tipos) que existam na rede [19]. É

permitido definir cores a partir de operações entre cores já existentes (ex: união, produto,etc.).

Aos arcos desta rede, ao invés de números inteiros, são associadas inscrições que correspondem a expressões. Essas expressões determinam, na ocorrência de uma transição, quantas e de que cores são as fichas que devem ser removidas dos lugares de entrada e adicionadas aos lugares de saída [19,35]. Inscrições também podem ser associadas às transições, sendo então denominadas guardas. Uma guarda é uma expressão que tem como resultado um valor booleano, cuja finalidade é restringir a ocorrência de uma dada transição [19].

As declarações de uma CPN (também denominadas nó de declaração) servem para especificar o tipo dos elementos que são suportados pelos lugares da rede, bem como as variáveis e funções utilizadas nas inscrições de arcos e guardas.

As inscrições dos arcos são construídas em função de *variáveis de transição* definidas nas declarações, cuja cor coincide com a cor do lugar de onde as fichas devem ser retiradas/adicionadas. O mesmo é válido para as guardas das transições. Uma *ligação* (*binding*) é a associação das variáveis de transição de um dado arco a um valor possível de cores. Um *elemento de ligação* é um par (t,b) onde t é uma *transição* e b é uma *ligação* para t [19].

Uma transição em uma CPN é dita estar habilitada se possuir em seus lugares um número de fichas suficiente para satisfazer as inscrições dos arcos (existir um *elemento de ligação*) e se sua guarda for satisfeita. Com isso são retiradas fichas dos lugares de entrada e são adicionadas fichas aos lugares de saída de acordo com as expressões (avaliação das inscrições) dos arcos que ligam a transição aos seus lugares de entrada e de saída.

Uma CPN pode então ser definida como sendo composta de uma estrutura (um grafo bipartido e direcionado), de um conjunto de inscrições e de um conjunto de declarações [19,35]. Em [19] é apresentada a definição formal da CPN.

Na Figura 2 é apresentado um exemplo de uma CPN. O sistema modelado nesta figura é de um sistema ferroviário composto por um trilho de trem circular, com dois trens (trem *a* e trem *b*), que se movem na mesma direção. O trilho é dividido em 7 setores. Para um trem se mover entre setores é necessário que o setor seguinte esteja livre (para que ele possa ocupá-lo) e que o setor subseqüente ao seguinte também esteja livre (para

garantir que não existirá a possibilidade de choque entre os trens). Inicialmente o trem *a* está no setor 7 e o trem *b* no setor 2.

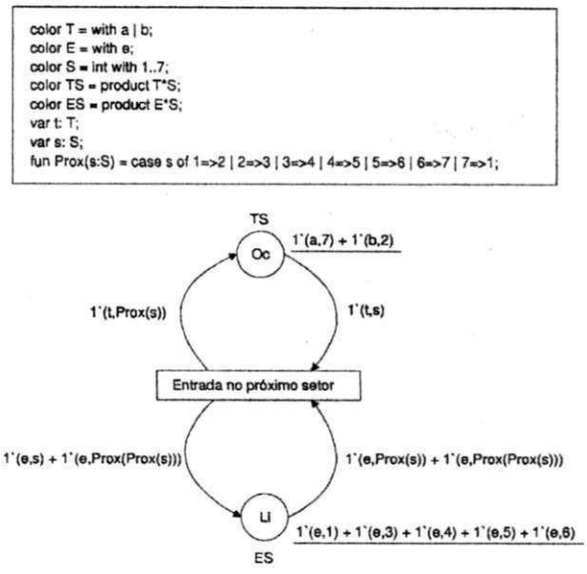


Figura 2 – Modelo do sistema ferroviário em CPN

Como exemplo da compactação propiciada pelas CPNs, é apresentada na Figura 3, o modelo construído utilizando redes de Petri lugar-transição.

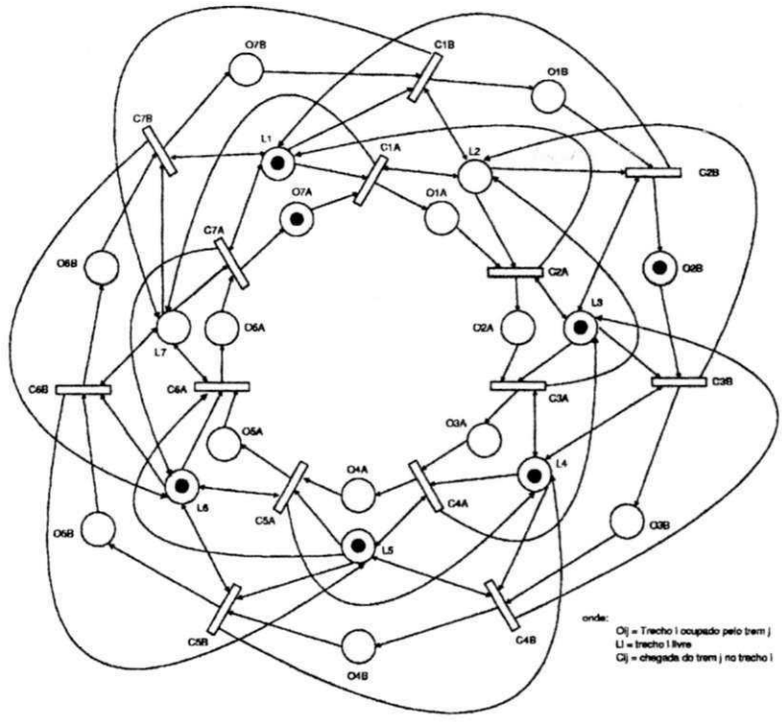


Figura 3 – Modelo do sistema ferroviário em redes de Petri lugar-transição

Com as CPNs também é possível realizar a modelagem de forma hierárquica (redes de Petri Coloridas hierárquicas) ou com a inserção de restrições temporais (redes de Petri Coloridas temporizadas). Pode-se modelar de forma hierárquica e temporizada, construindo assim um modelo hierárquico temporizado. Nas seções seguintes serão detalhadas as redes de Petri Coloridas hierárquicas e as redes de Petri Coloridas temporizadas.

2.6 Redes de Petri Coloridas Hierárquicas

O objetivo fundamental de uso de redes hierárquicas é possibilitar construir modelos a partir da combinação de outros modelos, similarmente à programação modular, que permite a construção de um programa mais complexo a partir de um conjunto de módulos e sub-rotinas [19].

Além disso, com o uso de hierarquia também é possível eliminar as redundâncias no modelo, uma vez que as partes redundantes podem ser instâncias de um único sub-modelo do modelo completo.

É importante atentar que com o uso de hierarquia não se obtém um maior poder de modelagem em relação às redes coloridas ou lugar-transição. Esses três tipos de redes possuem o mesmo poder de expressão. A diferença está na complexidade do modelo resultante. Podemos comparar estas três classes de redes em termos de linguagem de programação, onde as redes lugar-transição seriam equivalentes à programação em linguagem de máquina, as redes coloridas equivalentes à programação com o uso de dados estruturados, e as redes hierárquicas comparáveis à programação com o uso de módulos e sub-rotinas [19,35].

Com o uso de hierarquia é possível dividir o modelo em vários níveis de abstração, podendo assim lidar com uma visão mais abstrata de partes do modelo sem se preocupar com os detalhes da implementação dos itens que constituem os níveis mais baixos de abstração.

Para trabalhar com hierarquia, as redes de Petri coloridas hierárquicas (*Hierarchical Coloured Petri Nets – HCPN*) são definidas utilizando os conceitos de *lugar de fusão* e *transição de substituição*. *Lugares de fusão* são estruturas que permitem que vários lugares da rede se comportem como sendo um único lugar. Assim, todos os *lugares* que fazem parte de um *conjunto de fusão* (conjunto de *lugares* definidos como

um mesmo *lugar de fusão*) possuem as mesmas fichas e, quando uma ficha é adicionada ou removida de um lugar que faz parte de um *conjunto de fusão*, todos os lugares que fazem parte desse *conjunto de fusão* terão a mesma quantidade de fichas removidas/acrescidas [19].

Transições de substituição são estruturas que permitem relacionar uma transição em um nível mais alto de abstração a uma rede mais complexa que fornece maiores detalhes das atividades que a *transição de substituição* representa [35]. Dessa forma, existe a relação entre redes individuais e nós de outras redes, ou seja, existem relações entre redes não hierárquicas para formar uma rede hierárquica. Para isso é necessário introduzir o conceito de *páginas* [19]. As *páginas* se referem a cada uma das redes individuais que formam a rede hierárquica. Cada *página* deve possuir um nome único.

A rede mais detalhada que a *transição de substituição* representa encontra-se em uma *página* denominada *subpágina*, enquanto a rede que contém a *transição de substituição* é denominada *superpágina*. Cada *transição de substituição* é denominada *supernó* de sua *subpágina* correspondente.

A relação entre uma *transição de substituição* e sua *subpágina* correspondente se dá através de duas estruturas denominadas *portas* e *sockets*, que descrevem a interface entre a *transição de substituição* e a *subpágina* correspondente. Os *sockets* são atribuições aos lugares da rede contida na *superpágina* que estão conectados às *transições de substituição* enquanto as *portas* são atribuições a lugares da *subpágina*, de tal modo que um par *socket/porta* forma um *conjunto de fusão*. Com isso é definida uma conexão entre uma *superpágina* e uma *subpágina*, pois sempre que uma ficha é adicionada/removida de um *socket* ela também é adicionada/removida da *porta* associada a esse *socket* [19,35].

É interessante observar que uma HCPN pode ser transformada em uma rede não-hierárquica equivalente. Para isso é necessário substituir cada *transição de substituição* por sua respectiva *subpágina*, substituindo cada *socket* por sua respectiva *porta* [19].

Nas figuras a seguir é apresentado um exemplo simples de uma HCPN. Este modelo representa a troca de informações entre um dispositivo mestre e vários escravos, onde um mestre envia uma requisição para todos os escravos, os quais testam a requisição e enviam um cheque ao mestre, o qual define a ação a ser realizada e envia a ordem aos escravos. A página *Sistema* (Figura 4) contém duas transições de substituição, *Mestres* e *Escravos*. A transição de substituição *Mestres* tem como subpágina a rede *Mestres*

(Figura 5), e a transição de substituição *Escravos* tem como subpágina a rede *Escravos* (Figura 6). Uma definição formal das HCPNs pode ser encontrada em [19].

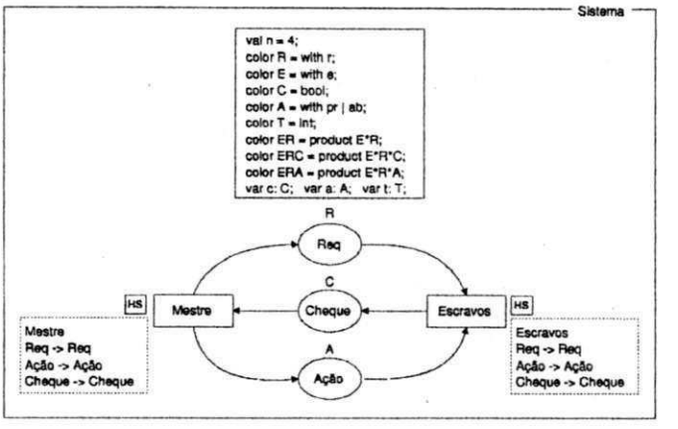


Figura 4 - Exemplo de rede hierárquica (superpágina)

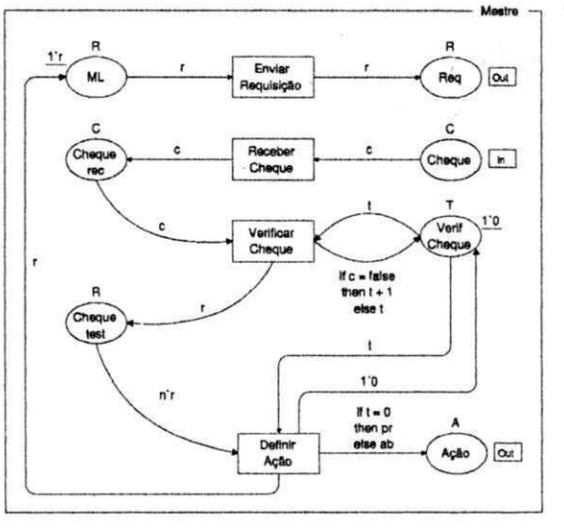


Figura 5 - Subpágina correspondente a transição de substituição Mestre

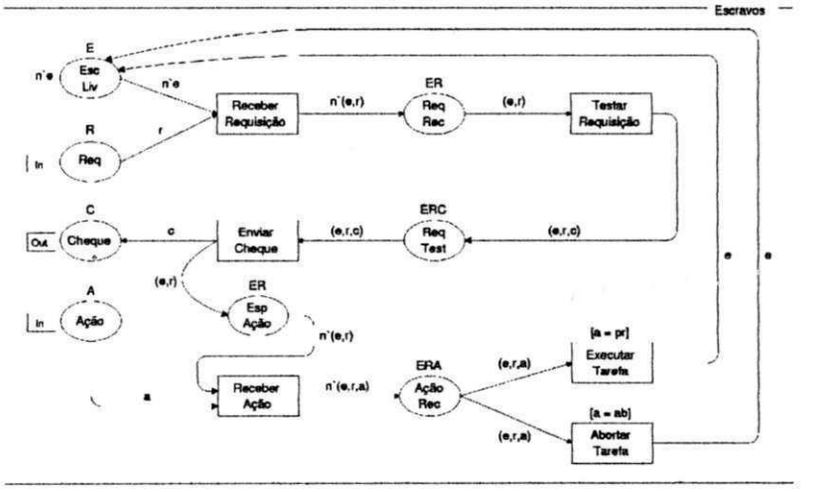


Figura 6 - Subpágina correspondente a transição de substituição Escravos

2.7 Redes de Petri Coloridas Temporizadas

As redes de Petri coloridas, hierárquicas ou não, não possibilitam a análise de desempenho do sistema modelado. Para contornar essa limitação foram propostas as redes de Petri coloridas temporizadas, nas quais foi incluído o conceito de tempo. Com o uso dessas redes é possível especificar como as diferentes atividades e estados "consomem" tempo.

A inserção do conceito de tempo em um modelo CPN se dá pela introdução de um *relógio global* (*clock*). O valor do relógio representa o *tempo do modelo*, que pode ser discreto ou contínuo. Com isso, cada ficha agora possui uma *marca de tempo* em adição a sua cor. A *marca de tempo* indica o tempo de espera mínimo a partir do qual a ficha pode ser usada (removida por um elemento de ligação) [20].

A habilitação de uma transição depende de dois fatores: da existência de um *elemento de ligação* e de sua *marca de tempo*. O conceito de habilitação das CPN em que bastava existir o *elemento de ligação* foi aqui extendido. A existência de um *elemento de ligação* indica que a transição contém as fichas necessárias para a sua habilitação sem levar em conta as restrições temporais (diz que a transição está *habilitada pela cor*). Mas, para a transição estar habilitada, o *elemento de ligação* deve estar também em um estado chamado *pronto*. Isso indica que todas as *marcas de tempo* das fichas removidas devem ser menores ou iguais ao tempo corrente do modelo (relógio do modelo) [20].

A modelagem de atividade que requer um certo tempo x para sua execução é obtida fazendo com que a correspondente transição t ao ocorrer crie marcas de tempo nas fichas de saída que são x unidades de tempo maiores que o valor do relógio na ocorrência de t. Com isso tem-se que essas fichas ficam indisponíveis para a habilitação de uma transição por x unidades de tempo, modelando assim a execução de um evento que consome tempo.

Como exemplo de redes coloridas temporizadas apresenta-se a rede da figura 7. Na Figura 7 pode-se observar que nas declarações, a cor P é temporizada. O retângulo abaixo das declarações mostra o valor corrente do relógio, e as fichas nos lugares de cor P possuem uma marca de tempo (valor após o símbolo @). Como é mostrado na figura, uma rede temporizada pode conter elementos não temporizados.

Na Figura 7 apenas a transição *T4* está habilitada, pois ela dispõe da ficha não-temporizada necessária, e a ficha temporizada necessária está no estado *pronto* (marca de

tempo 641 alcançada). Com a ocorrência de *T4* essas fichas de entrada são retiradas e é criada uma ficha no lugar *E* com uma marca de tempo 12 unidades maior que o tempo no qual *T4* ocorreu (definido pela inscrição do arco de saída). Com isso *T5* não se torna habilitada pois a ficha no seu lugar de entrada não está no estado pronto (*T5* está apenas habilitada pela cor). *T5* só será habilitada quando o relógio global chegar na marca de tempo de ficha em *E* (653 unidades de tempo), ocasionando a ocorrência de *T5*.

Uma definição formal das CPN temporizadas pode ser encontrada em [20].

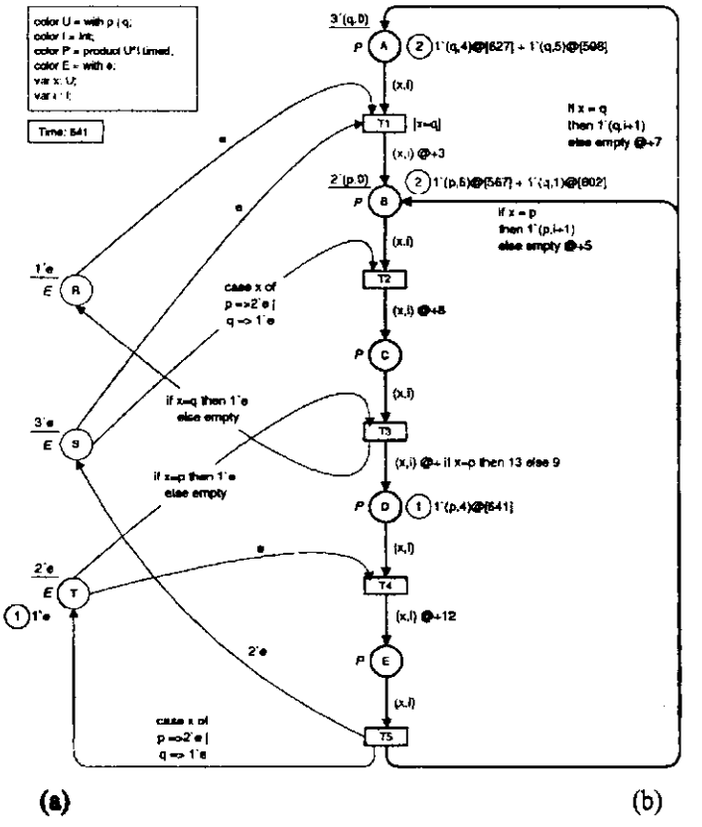


Figura 7 - CPN temporizada

2.8 Análise de Redes de Petri

Com o uso de redes de Petri é possível efetuar a verificação formal de propriedades dos modelos construídos. São dois os grupos de propriedades analisáveis nas redes de Petri: as *propriedades comportamentais* ou *dinâmicas*, que dependem da marcação inicial; e as *propriedades estruturais*, que dependem apenas da estrutura da rede [19,20].

Os métodos de análise se baseiam na verificação de propriedades comportamentais do sistema, através da construção de *grafos de ocorrência*; na verificação de propriedades estruturais do sistema, através de *invariantes de lugar e transição*; e, na redução do modelo tendo em vista a facilitação de sua análise, através de técnicas de *redução e decomposição de redes* [19,20].

O *grafo de ocorrência* (*Occ – Occurence Graph*) é um grafo que contém um nó para cada marcação alcançável na rede (a marcação é descrita em uma inscrição de texto no nó) e um arco para cada elemento de ligação ocorrido na rede, ou seja, é uma representação do espaço de estados da rede [19]. No caso das redes lugar-transição ao invés de elementos de ligação têm-se a indicação da transição que ocorreu [6,23]. Nas redes coloridas temporizadas também há uma diferença no *Occ*. Os nós contêm um estado ao invés de uma marcação. Cada nó contém um valor de tempo e uma marcação temporizada [20].

Para as CPN, HCPN e CPN temporizadas existe um pacote de ferramentas computacionais de apoio a modelagem e análise das redes, o *Design/CPN* [8]. Ele é constituído de quatro ferramentas computacionais integradas em um único pacote:

- Editor gráfico para construir, modificar e executar análise sintática de modelos CPN;
- Um simulador que pode operar simulação interativa ou automática, possibilitando ainda definir diferentes critérios para parada e observação da evolução da rede;
- Uma ferramenta para gerar e analisar grafos de ocorrência de modelos CPN;
- Uma ferramenta para análise de desempenho que possibilita a simulação e observação de dados de desempenho de modelos CPN.

O Design/CPN é um dos pacotes de ferramentas mais elaborados para construção, modificação, e análise de redes de Petri, e tem sido extensivamente utilizado com sucesso em diferentes domínios de aplicação. Maiores detalhes sobre a ferramenta podem ser encontrados em [8].

2.9 Considerações Finais

Neste capítulo foram apresentados, de forma introdutória, os conceitos relacionados com o formalismo adotado neste trabalho para a etapa de modelagem e avaliação da interface homem-máquina industrial.

No próximo capítulo serão descritas as interfaces homem-máquina, focalizando no contexto de aplicação de programas supervisórios industriais.

Capítulo 3 – Interfaces Homem-Máquina

3.1 Introdução

Neste capítulo serão tratadas as interfaces homem-máquina e será discutida a sua importância em um processo interativo. Será dada uma visão acerca dos processos industriais e da automação destes processos. Serão abordadas especificamente as interfaces industriais, apresentando seus elementos e características mais importantes.

A necessidade de uma maior discussão sobre as interfaces homem-máquina industriais, base central para o desenvolvimento deste trabalho, decorre da natureza crítica dos sistemas que elas controlam.

3.2 Interfaces Homem-Máquina (IHM)

A interface com o usuário de um sistema envolve considerações sobre o sistema propriamente dito, o usuário do sistema e a maneira na qual o usuário e o sistema interagem. Além dos módulos projetados para a interação, a interface abrange modelos e impressões que são construídos na mente do usuário em resposta a interação com esses módulos. Então o projeto da interface do usuário incorpora elementos que são parte do sistema, características do usuário e os métodos de comunicação de informação entre eles [3]. Na Figura 8 temos a relação entre o usuário e o sistema através da interface homem-máquina.

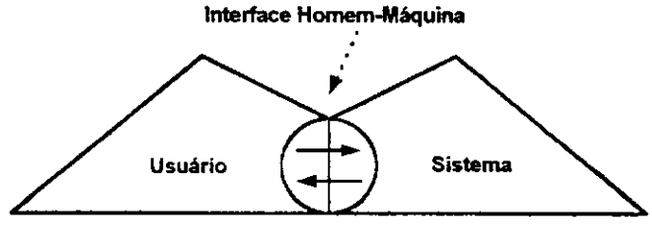


Figura 8 - Interação usuário/sistema por meio da IHM

No contexto de utilização de software, pode-se definir o termo interface homem-máquina como sendo o componente de um aplicativo que traduz uma ação do usuário em uma ou mais solicitações à aplicação propriamente dita e fornece ao usuário uma resposta em conseqüência de suas ações.

Mas, uma realidade fundamental no desenvolvimento de aplicativos é que para o usuário de um sistema computacional interativo a interface é o sistema. O que os usuários desejam é que os projetistas desenvolvam aplicações que vão ao encontro de suas necessidades e que sejam fáceis de usar.

3.3 A Importância da IHM

Um dos fatores mais significativos na qualidade de um sistema computacional interativo é a IHM. Nenhum segmento da Ciência da Computação sofreu mais alterações nos últimos anos que este: o desenvolvimento de novas técnicas, o aparecimento de novas tecnologias para implementar essas técnicas, novas comunidades de usuários com formações às mais diversas e novas tarefas propostas a cada dia. E, com o atual surgimento de dispositivos de interfaceamento mais ergonômicos, dotados de recursos cada vez mais abrangentes, aumenta a possibilidade da especificação e implementação de interfaces mais poderosas [33].

Com essa sofisticação de recursos, bem como técnicas de interação mais adequadas, o usuário assumiu uma postura mais ativa, podendo 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 antes raramente encontrado. Com isso o componente de interface de um sistema interativo tem características de concorrência e requisitos de tempo-real [40].

Nos dias atuais as interfaces possibilitam dois estilos de interação: a *interação seqüencial*, onde o fluxo da troca de informações entre o usuário e o sistema é previsível, com cada ação levando a um sucessor predefinido; e a *interação concorrente*, na qual o usuário realiza várias tarefas simultaneamente, e onde o seqüênciamento dentro de uma dada tarefa é independente do seqüênciamento das outras [40].

O propósito chave da interação homem-máquina é possibilitar ao usuário o uso dos recursos das aplicações disponíveis no sistema (para a solução de problemas específicos) de forma "transparente", isto é, sem preocupação em como eles são

executados. O usuário deve se sentir apto a explorar todos os recursos disponíveis no sistema, mantendo um elevado nível de satisfação durante todo o processo interativo.

A eficiência da interação homem-máquina se traduz na facilidade da comunicação entre o usuário e o conjunto de recursos disponíveis; na segurança que o sistema desperta no usuário ao executar as tarefas por ele solicitadas, permitindo ao usuário concentrar-se em seu trabalho, bem como prever o que ocorre após de cada uma de suas ações; no desempenho da execução das tarefas pelo sistema e na integridade das informações fornecidas ao usuário. [33].

Porém, mesmo que se defina a funcionalidade adequada, que se assegure à confiabilidade e se otimize o tempo/custo computacional das tarefas, o projeto final da interface poderá ser inadequado para o grupo de usuário ao qual ela se destina. Um dos requisitos essenciais para o sucesso do projeto de uma IHM é a familiarização do projetista com as características da comunidade de usuários-alvo e com seus objetivos. Outro requisito é a definição do conjunto de recursos que será oferecido para a execução de tarefas inerentes ao contexto. Assim, o requisito mais importante em todo processo de concepção do modo de interação com os usuários é a consideração de aspectos ergonômicos [33].

É importante, então, realizar um "Projeto centrado no usuário". Esse tipo de projeto se fundamenta no projeto da interação (e conseqüentemente a IHM) do ponto de vista do que é melhor para o usuário, ao invés de fundamentar as decisões de projeto no que é mais rápido e fácil de implementar [44]. Também deve levar em consideração, como já foi mencionado, o perfil do usuário-alvo do sistema, projetando a IHM baseada no perfil "médio" dos usuários desse sistemas para que as interações não sejam consideradas pelo usuário nem muito simples nem muito complexas, pois em ambos os casos o usuário se sentirá desmotivado para o uso do sistema. E, como também já foi mencionado, deve-se dar uma atenção especial a ergonomia da IHM, para que o usuário consiga localizar as informações de forma clara, evitando o cansaço do usuário além de reduzir a probabilidade de erros.

Deve-se observar que as diferenças individuais entre usuários e a variabilidade das tarefas por eles realizadas são dois fatores de grande impacto sobre a usabilidade do sistema (e, por conseqüência da IHM), com grande repercussão sobre a decisão dos projetos no que diz respeito ao tipo de estilo de interação a ser adotada e no conjunto de ações que deverão ser implementadas. Facilidade de aprendizado, possibilidade de

exploração máxima dos recursos disponíveis após a etapa de aprendizagem, freqüência e gravidade de erros possíveis, possibilidade de adaptação de usuários a outras ferramentas oferecidas no ambiente e, sobretudo, a satisfação subjetiva dos usuários são algumas das regras mais importantes da engenharia de usabilidade. Consistência é outro ponto chave na interação homem-máquina que recebe atenção especial ao se destacar como um dos problemas de usabilidade da atualidade [33].

Então podemos observar que a realização de um projeto de Interface Homem-Máquina é muito mais que apenas um bom projeto visual, deve-se também ter a preocupação com o conforto, a segurança e a eficiência dessa interface [3]:

- Conforto: O conforto para o usuário é mais do que apenas um ambiente físico confortável e ergonômico. Em relação a IHM, o conforto tem relação com a apresentação visual das informações. As IHM não podem se confusas, dispersas, ou possuírem vários tipos de códigos para as operações similares. As IHM devem ser coerentes, claras e concisas de modo ao usuário se senta confortável durante o uso.
- Segurança: A segurança do usuário na realização de uma ação é um fator muito importante. A IHM deve prover meios para que o usuário saiba exatamente o que está fazendo, além de prover uma realimentação sobre a completa realização de uma tarefa.
- Eficiência: A eficiência em uma IHM diz respeito ao projeto funcional, permitindo a realização de uma tarefa da forma mais rápida, melhor e com a menor taxa de erro.

3.4 A Manufatura Atual

A expansão da informática em nível mundial e a ampla comercialização de computadores após os anos 60 tiveram impacto transformador no processo industrial e no papel do trabalhador desse tipo de processo. Além da informática, as novas técnicas de gestão da produção [7] como o *Just-in-Time*, a *Manufatura Integrada por Computador*, o *MRP* (*Material Requirement Planning – Planejamento de Necessidades de Materiais*), o *MPRII* (*Manufacturing Resource Planning – Planejamento de Recursos de Manufatura*), a *TQM* (*Total Quality Management – Gestão pela Qualidade Total*), a *Gestão Holística*, a *Engenharia Simultânea*, a *Reengenharia de Processos de Negócios* entre outras, tiveram

um papel fundamental na configuração do sistema produtivo às vésperas do século XXI. Essas técnicas têm provocado sensíveis mudanças nas atividades que ocorrem em um sistema de manufatura [7].

Por outro lado, com o surgimento de novas tecnologias microeletrônicas, as formas de produção foram substancialmente alteradas. A utilização maciça de equipamentos microprocessados na linha de produção e nos diversos setores das empresas vêem modificando amplamente os vários conceitos produtivos, além de permitir vários ganhos, tais como a redução no tempo de lançamento de novos produtos, a reengenharia de cadeias de processo, o aumento na confiabilidade do produto, o aumento na qualidade e a redução de custos, entre outros.

Esses fatores estão relacionados com a maneira através da qual este suporte computacional é utilizado. De fato, a definição de uma arquitetura tecnológica adequada às necessidades da empresa pode se constituir em uma importante vantagem competitiva. Se, por um lado, a automação de tarefas executadas em cada setor resulta, por si só, em grandes benefícios, a sua integração em uma arquitetura tecnológica bem estruturada pode resultar em uma nova forma organizacional, permitindo novos ganhos. A definição da arquitetura tecnológica se dá em consonância com o conceito de integração que se pretende implantar. De fato, diferentes conceitos de integração da produção irão demandar diferentes tipos de equipamentos bem como níveis distintos de complexidade de implementação [7].

Deve-se portanto observar, que a indústria de manufatura é hoje um dos setores que mais tem exigido pesquisas e suporte tecnológico para acompanhar essas enormes mudanças que estão ocorrendo nas corporações produtivas.

No sentido de otimizar o processo de manufatura, surgiram diversas iniciativas tecnológicas que trouxeram substanciais melhoras no desempenho da produção industrial, diminuindo prazos de desenvolvimento de produtos e custos de produção, bem como aumentando a qualidade dos produtos e a produtividade da linha de produção. Porém, a implementação de muitas dessas iniciativas tecnológicas em um processo industrial traz consigo um custo de investimento freqüentemente muito alto para ser bancado por uma indústria que já tem uma sistemática de produção definida. Para tornar este investimento atraente para o capital gestor do processo, a inserção de novas tecnologias no meio produtivo precisa garantir alguns benefícios, como: incremento na produtividade, redução

no custo dos produtos e no tempo gasto no ciclo de produção, aumento na competitividade, etc [15,16].

Dentre as iniciativas tecnológicas existentes, a automação no controle de processo é aquela que, geralmente, traz a melhor relação custo/benefício, pois consiste da alteração apenas do método de gestão operacional do processo, não necessitando de mudanças significativas na sua estrutura, ao mesmo tempo em que geralmente causa um grande impacto na otimização operacional do produto [15].

3.5 Automação no Controle de Processos

O cenário típico onde se enquadra esta iniciativa é aquele de um processo industrial grande o suficiente para que a supervisão operacional seja centralizada, que já possui todo um conjunto de aspectos de produção definidos, tais como: operacionalização, custos, produtividade, segurança, ciclo de produto, etc. As variáveis físicas são monitoradas e controladas, através de variáveis elétricas, por controladores programáveis que localizados em uma sala de controle da planta, apresentam o estado das variáveis do processo ao operador, o qual pode modifica-las através do ajuste de set-point [14].

Esse tipo de controle da planta apresenta algumas limitações no tocante a produtividade e a segurança do trabalho, já que o operador fica sobrecarregado pelo grande número de informações provenientes do Controladores Programáveis, constituindo com isso uma grande fonte de erros de controle [15].

Para superar esta limitação surgem os sistemas de supervisão de processo ou sistemas supervisórios (que consistem basicamente de um computador e um programa aplicativo que processa as informações provenientes dos Controladores Programáveis).

3.6 Sistemas de Supervisão de Processos

A partir do momento em que a monitoração e o controle de um processo são feitos com a ajuda de um sistema supervisório, o processamento das variáveis de campo é mais rápido e eficiente. Qualquer evento imprevisto no processo é rapidamente detectado e mudanças nos *set-points* são imediatamente providenciadas pelo sistema supervisório, no sentido de normalizar a situação. Ao operador fica a incumbência de acompanhar o processo de

controle da planta, interferindo apenas em casos em que sejam necessárias tomadas de decisão de atribuição restrita ao operador [16].

Com o uso de sistemas supervisórios ocorre também a otimização das tarefas cotidianas, dentre as quais podemos citar: mudanças de ritmo de produção, modificações de variáveis do processo, acompanhamento da produtividade de uma planta industrial, etc.

Na Figura 9 é apresentada a estrutura simplificada de um sistema supervisório, constituído de uma estação supervisora (onde é executado o programa supervisório) e dos dispositivos sob supervisão [14].

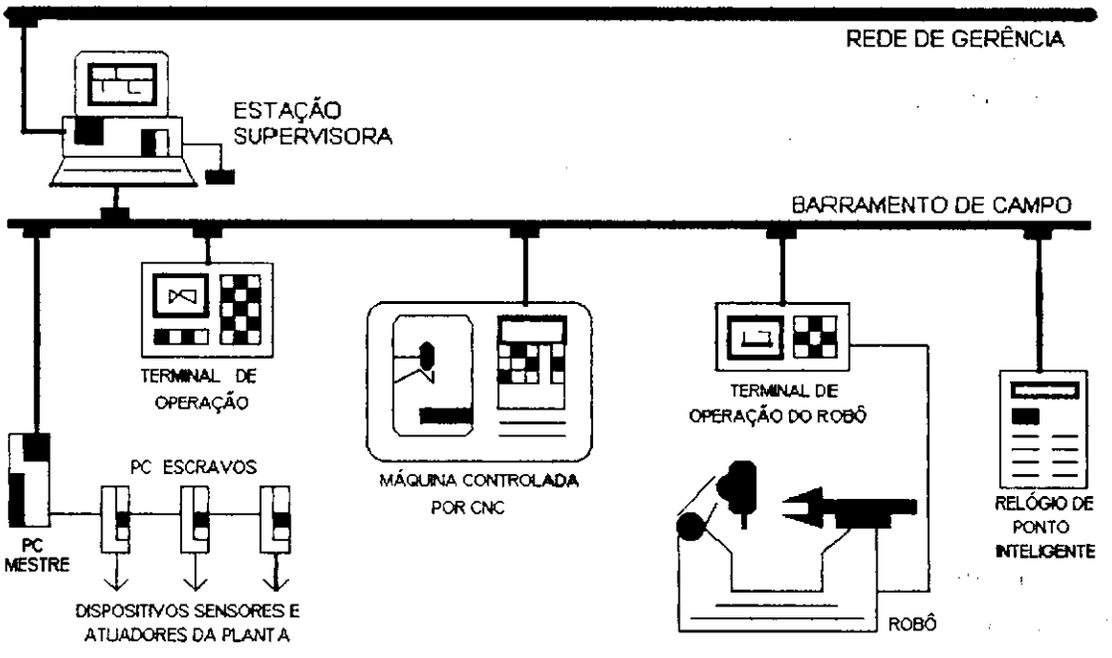


Figura 9 - Estrutura de Supervisão de um Processo Industrial

A tarefa do sistema supervisório é recolher informações sobre as variáveis do processo que são consideradas relevantes para o seu controle, processá-las, executar algum procedimento automático (se for o caso), e apresentar as informações processadas ao operador do sistema, em tempo real, através de uma IHM, de forma clara e inteligível. Através da IHM o operador pode receber alarmes sobre variáveis fora da faixa configurada como normal, e dispõe de meio para atuar no processo de forma a corrigir estes problemas [14].

Mais do que o processo industrial em si, paradigmas da produção foram transformados com a automatização e informatização da indústria. Pois o sistema

supervisório ainda serve de comporta de dados entre o chão de fábrica e a rede de gerência da produção. Através da estação supervisora, os níveis gerenciais superiores da industria podem obter dados em tempo real sobre o andamento da produção, comparar esses dados com informações referentes à demanda e ao estoque, e tomar decisões administrativas de forma mais rápida, precisa e eficiente. Vale salientar que o acesso em tempo real às informações de produção não se restringem apenas a gerência local da planta, mas também ao níveis gerenciais remotos a planta, através de acesso via rede pública ou privada.

3.7 Interface Homem-Máquina de Programas Supervisórios

Excetuando-se as diferenças de complexidade do controle dos processos a serem automatizados, a implantação do programa supervisório em um contexto de produção é bastante semelhante para os diversos tipos de produtos existentes no mercado. Isso faz com que um dos principais fatores diferenciadores do resultado de sua aplicação seja a Interface Homem-Máquina (IHM). A IHM define a qualidade do sistema supervisório quanto a ergonomia, segurança e produtividade do trabalho do operador.

Deve-se observar também que a qualidade final da IHM depende principalmente da qualificação do projetista para esse tipo de trabalho. Estudos do Grupo de Interface Homem-Máquina do Departamento de Engenharia Elétrica da UFPB [16], têm evidenciado que os projetistas de IHM das aplicações dos programas supervisórios não tem, em geral, uma formação que propicie projetos com características ergonômicas e funcionais otimizadas. Este problema é agravado nos casos em que falta ao projetista, além da formação em IHM, uma visão mais ampla da operacionalização do processo sob automação, dos pontos de vista de engenharia de processos e de segurança do trabalho.

Segundo Farias em [15], apesar da diversidade de processos que controlam, as interfaces industriais compartilham várias características em comum, listadas a seguir:

- Apresentação do sinótico (resumo da planta industrial) da planta monitorada, dando ao operador uma visão geral do sistema e de seu funcionamento;
- Apresentação de tabelas de valores de variáveis do processo, que podem ser modificadas (a exemplo de set-points);
- Geração de relatórios automáticos de produção;

- Geração de gráficos de histórico, que mostram em forma gráfica as variáveis do processo em relação ao tempo ou outras variáveis;
- Geração de gráficos de tendências;
- Sumário de alarmes, que são reconhecidos pelo operador para tomada de atitudes necessárias;
- Emulação de operação de equipamentos remotos, permitindo a operação de um equipamento remoto como se o mesmo estivesse fisicamente presente;
- Apresentação de informação multimídia para a supervisão do processo industrial.

Os elementos que compõem uma IHM industrial são semelhantes aqueles que compõem as interfaces de um modo geral, tais como: menus, botões e caixas de entrada de dados. A principal diferença consiste na natureza crítica da aplicação. A estrutura de navegação das interfaces industriais em geral é fortemente hierárquica, tipicamente oferecendo apenas uma alternativa de acesso a cada aspecto de funcionalidade do sistema supervisório. Isso se deve à demanda por respostas rápidas e inequívocas do operador do sistema; pois nestas circunstâncias a carga cognitiva sobre o operador deve ser reduzida de modo a minimizar as chances de interpretação errada da informação e o conseqüente equívoco na tomada de decisões [11].

Do estudo das interfaces industriais, podemos destacar as seguintes tarefas típicas realizadas pelo usuário durante a interação [11]:

- Comutação do estado de dispositivos: Consiste nas ações que levam a ativação ou desativação de objetos representados na interface, tal como um botão liga/desliga;
- Ajuste/entrada de valores: Consiste na digitação de dados alfanuméricos ou ainda na manipulação de um objeto gráfico na interface com o propósito de atribuir um valor a uma variável do processo dentro de uma faixa de valores;
- Seleção de opções: Seleção de uma ação para o processo, dentro de um conjunto de comandos disponíveis ao operador;
- Solicitação de informações: Consiste na solicitação da exibição de tabelas, gráficos e relatórios sobre o processo; ou de sua impressão para fins gerenciais;
- Monitoração de variáveis do sistema: Consiste na leitura e atualização de valores de variáveis do sistema industrial. Esses valores podem estar representados na forma gráfica ou alfanumérica;
- Navegação entre janelas: Consiste na seleção de uma nova janela da IHM com a qual se deseja interagir. Essa seleção pode ser através de botões, menus, listas, etc.

A Figura 10 exemplifica todas as tarefas descritas acima.

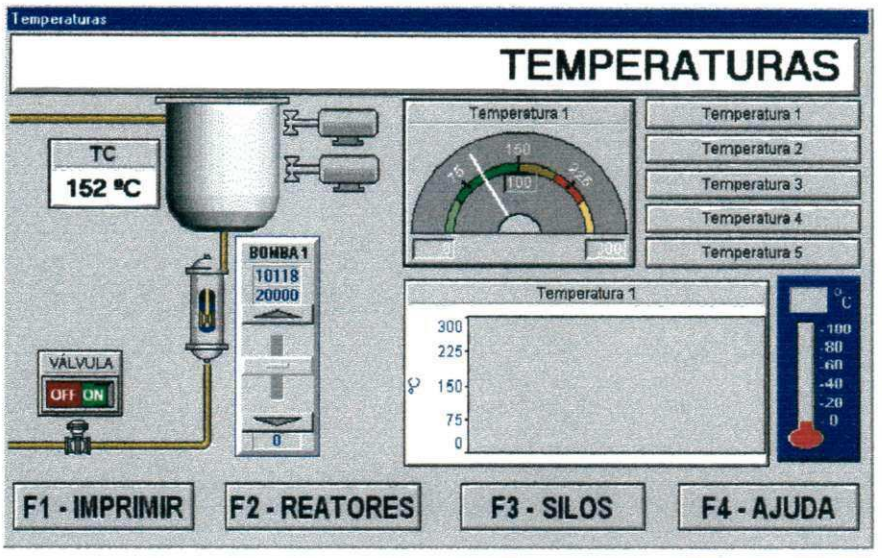


Figura 10 - Exemplo de Janela de uma Interface Homem-Máquina Industrial

Para a realização dessas tarefas nas IHMs de programas supervisórios existem funcionalidades e objetos de interação típicos [12,13], tais como:

- Botão liga/desliga
- Botão de reversão (*toggle).*
- Ajuste contínuo de variável
- Seleção de opções
- Apresentação de informação na forma gráfica e alfanumérica

Segundo estudos realizados por vários autores [1,15,16] dentre os aspectos a serem observados no projeto de interfaces industriais destaca-se o aspecto da navegação. Ainda segundo [12,13], as estruturas de navegação destas IHM podem ser classificadas em:

- Navegação direta: Navegação entre janelas, solicitada através de um botão ou de outro dispositivo de interação especificado.
- Navegação de retorno: Navegação de retorno a última janela ativa antes da janela atual.
- Navegação de escape: Navegação de retorno entre janelas quando a janela de origem só pode ser acessada a partir da janela de destino.

- Navegação *GOTO*: Navegação para uma janela da interface, independentemente da seqüência de navegação previamente especificada (ex: navegação para a janela de alarme).[1]

Mais fortemente do que em outros contextos, no ambiente industrial o projeto da interface homem-máquina é um processo crítico. Uma interface confusa pode levar a uma interpretação errada das informações por parte do operador ocasionando erros cujas conseqüências podem ser graves (um erro pode ocasionar uma parada de produção ou até impor risco de vida às pessoas envolvidas no processo).

O operador do programa supervisório necessita de acesso rápido a informação devendo também responder de modo rápido e inequívoco. A carga cognitiva sobre o usuário deve ser reduzida de modo a minimizar as chances de interpretação errada da informação que resultem no acionamento de "comandos" errados para o sistema. Portanto a interface demanda uma estrutura hierárquica, bem definida, com um elevado grau de objetividade e simplicidade na navegação. Estas características estão diretamente relacionadas com a natureza da atividade fim: o controle da planta industrial [13].

Outro fator importante nestas interfaces é que o operador tem que reagir a eventos e completar uma tarefa considerando o prazo final e a segurança do processo [15]. Assim, um fator de grande importância no projeto de uma interface industrial é o tempo.

As propriedades temporais da interação podem determinar a usabilidade de um sistema interativo [21]. Atrasos na resposta ao usuário fazem aumentar a taxa de erros cometidos pelos usuários [17,21]. E, por outro lado, respostas lentas do usuário também são inadequadas quando o sistema possui restrições temporais para a execução das tarefas [17]. Em sistemas críticos como os de controle industrial, atrasos na resposta do usuário a determinadas ações do sistema (como a ocorrência de um alarme) podem resultar em situações catastróficas (perda de produção, quebra de máquinas, risco de vidas as pessoas envolvidas no processo, etc.).

3.8 Considerações Finais

Neste capítulo foram abordadas as interfaces homem-máquina e a sua importância em um processo computacional interativo. Como foi enfatizado é de suma importância

[1] Em geral o sistema de supervisão industrial tem a capacidade de interferir na seqüência de interação em situações de ocorrência de alarme quando é aberta automaticamente a janela de alarmes.

que o componente de interface seja projetado levando em contas vários fatores que dizem respeito à usabilidade e à ergonomia, e não apenas o projeto visual.

Em particular foi discutida a importância deste componente para o desempenho de sistemas de supervisão e controle industrial.

Capítulo 4 – Avaliação de Interfaces Homem-Máquina

4.1 Introdução

Este capítulo aborda, de forma sucinta, os métodos de avaliação de interfaces homem-máquina encontrados na literatura.

Também são apresentados, de forma sucinta, os trabalhos de avaliação de interfaces com base no uso do formalismo das redes de Petri - método formal adotado para a modelagem e avaliação da interface industrial.

4.2 Avaliação de Interfaces Homem-Máquina

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 processo 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 [40]. Para o usuário, problemas na interface significam, problemas no sistema como um todo, já que o usuário vê o componente de interface como uma representação do sistema como um todo.

Inicialmente, as avaliações de sistemas interativos focalizavam o desempenho do computador. Nos anos 60, a avaliação de interfaces se limitava apenas a verificar o sucesso ou não na realização de uma dada tarefa, sem se preocupar com a qualidade da interação. Nos anos 70, e início dos anos 80, os projetistas ainda relegavam a avaliação a um *status* de menor importância, priorizando à elaboração de um projeto mais elaborado do ponto de vista funcional. Somente com a crescente complexidade dos métodos de interação, a avaliação passou a focalizar a relação usuário-computador e o desempenho do usuário, onde o maior objetivo da avaliação tem sido relacionar a capacidade e limitações

do usuário aos requisitos que estas capacidades e limitações impõem sobre a tecnologia dos sistemas interativos (estudo de fatores humanos) [40].

A avaliação da qualidade do componente de interface homem-máquina de um sistema computacional interativo pode ocorrer com base em um conjunto de propriedades e características relacionadas ao desempenho do sistema e à sua usabilidade. Os objetivos gerais de uma avaliação são [34]:

(i) A avaliação das potencialidades do projeto;

(ii) A avaliação dos impactos causados pelas decisões de projeto;

(iii) O diagnóstico de problemas relativos ao projeto, independentemente do tipo de interface, do *hardware* e *software* considerados, do estágio do projeto, da presença ou ausência de fatores humanos na avaliação, da abordagem metodológica adotada e do tipo de dados coletados.

Os estudos avaliatórios fundamentalmente envolvem o processo de fazer questionamentos, e então projetar meios de se conseguir respostas úteis [18]. Esses estudos podem ser *formativos* ou *somativos*.

A avaliação *formativa* é aquela que é executada durante todo o processo de concepção e desenvolvimento do projeto, gerando uma realimentação de forma a otimiza-lo. A avaliação *somativa* é aquela que é realizada ao término de diferentes etapas do desenvolvimento do projeto com o propósito de verificar, em relação a critérios pré-estabelecidos, o grau de sucesso em relação aos objetivos propostos para uma dada etapa de projeto.

Um dos pontos mais importantes para o sucesso do processo avaliatório é a escolha da forma na qual o processo avaliatório será conduzido. É interessante a troca de idéias entre avaliadores como forma de identificar potenciais dificuldades que possam surgir durante o processo de avaliação [18].

Outro ponto a ser observado para o sucesso do processo avaliatório consiste na escolha da metodologia de avaliação, que depende do tipo de questionamento que será realizado. Para isso deve-se observar alguns aspectos do processo [18]:

(i) Verificar quão exploratório é necessário que o estudo seja;

(ii) Verificar o quão autêntico o estudo precisa ser;

(iii) Contabilizar o número de pessoas envolvidas no processo avaliatório.

A partir da definição destes três parâmetros é possível definir uma metodologia que mais se adeque ao estudo, do ponto de vista de uma maior abrangência e um menor

número de limitações ao seu uso. É interessante observar que nem sempre apenas uma metodologia é adequada para o processo, sendo uma dada metodologia mais atraente devido a um aspecto do estudo, e outra metodologia em relação a outro, etc. Nestes casos, que são os mais comuns, se deve utilizar a combinação de vários métodos de forma a que se complementem, gerando uma resposta global que seja o mais próxima da desejada.

Para se definir os três parâmetros acima citados é necessário ter uma visão global do processo avaliatório, desde a intenção da avaliação, passando pelos recursos (humanos e materiais) disponíveis, até o público-alvo a que o produto se destina.

Em seguida, deve-se fazer um quadro sinóptico do processo avaliatório proposto, justificando a sua necessidade no processo de desenvolvimento ou possíveis aperfeiçoamentos da interface. Deve-se ter uma visão clara dos esforços que serão necessários para a realização do estudo. E, deve-se identificar o público-alvo dos resultados como forma de definir a forma de apresentação das informações resultantes, pois diferentes níveis de informações convencem diferentes categorias de pessoas, bem como influenciam na forma como serão utilizadas [34].

Não existe na literatura um consenso sobre a classificação de processos avaliatórios de usabilidade. Mas, de uma forma geral os processos avaliatórios de interfaces homem-máquina podem ser classificados em *ensaios de usabilidade*, *inspeções de usabilidade* e a utilização de *métodos formais* [34].

4.3 Ensaios de Usabilidade (Usability Testing)

Ensaios de usabilidade consistem basicamente de estudos de um processo interativo usuário-computador específico, em condições "reais" ou "controladas", onde especialistas em interfaces coletam dados sobre eventos relacionados com a interação propriamente dita e problemas afins ocorridos durante o uso da aplicação por uma amostra da comunidade usuária [34].

São testes realizados com usuários "reais" que estão envolvidos cotidianamente com o objeto sob avaliação. Com o uso desta estratégia o avaliador tem a possibilidade de coletar dados relativos a forma de interação entre os indivíduos e os sistemas computacionais, e podem detetar quais problemas estão relacionados com o componente de interface do sistema.

Também deve ser observada cuidadosamente a *confiabilidade* do teste, que diz

respeito à obtenção de resultados similares após a repetição dos ensaios sob condições similares; bem como a *validade* do teste em relação ao produto quando utilizado em um ambiente não-controlado.

Os mecanismos de avaliação mais comumente empregados em *ensaios de usabilidade* são [34]:

- Observações;
- Uso de Questionários;
- Entrevistas;
- Verbalização de Procedimentos (*Thinking Aloud*);
- Interação Construtiva;
- Ensaio Retrospectivo;
- Captura Automática diretamente da aplicação;
- Discussões em Grupo (*Focus Groups*);
- Retorno Imediato de Opiniões do Usuário (*User Feedback*).

Um detalhamento acerca desses mecanismos pode ser encontrado em [34].

4.4 Inspeções de Usabilidade (Usability Inspections)

São testes avaliatórios fundamentados na análise e julgamento de projetos, por avaliadores (ergonomistas, engenheiros de *software*, engenheiros de usabilidade), que investigam aspectos relativos à usabilidade segundo um conjunto de critérios, recomendações, normas ou heurísticas[2]. [34].

Dentre as estratégias de inspeção da usabilidade podemos citar:

- Revisões Sistemáticas (*Walkthroughs*);
- Inspeção Baseada em Diretrizes de Projeto, Guias de Estilo e Padrões;
- Avaliação Heurística;
- Inspeção Fundamentada na Perspectiva;

Um detalhamento acerca desses mecanismos pode ser encontrado em [34].

[2] Heurística é um conjunto de recomendações baseadas na experiência do avaliador.

4.5 Métodos Formais

São estratégias de avaliação baseadas na descrição formal de um conjunto de propriedades e características lógicas e temporais da IHM. Com o uso desses métodos é possível especificar a descrição formal dos requisitos e idéias de projeto em diferentes níveis de abstração. O uso de métodos formais na descrição da interface, nos estágios iniciais do projeto, reduz o tempo de projeto pois elimina a necessidade da construção de protótipos de outra forma, necessários à avaliação. A análise se dá no modelo, o qual é facilmente alterado caso algum problema seja encontrado. Através da análise é possível eliminar problemas encontrados e sugerir alternativas de interação. Os métodos formais, por serem baseados no uso de uma notação com semântica precisa, tornam possível a discussão de propriedades especificadas bem como a predição do desempenho da interface a ser construída. Com isso, é possível a verificação de vários aspectos do projeto antes mesmo de sua implementação (a exemplo do componente de navegação).

A seguir são relacionadas características consideradas essenciais para modelagem das interfaces [28,40]:

- *Existência de um formalismo*: Um sistema de interfaces necessita de uma representação formal que possibilite sua especificação de forma ambígua e facilite a sua 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.
- *Estrutura de dados e de controle*: 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.
- *Mecanismos de estruturação*: 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.
- *Capacidade de descrever paralelismo*: 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 interaja com o *software* através de um dispositivo puramente seqüencial, a exemplo do teclado.

A verificação de um sistema através de seu modelo pode ser efetuada a partir de uma *análise qualitativa* e/ou de uma *análise quantitativa* [25]. A *análise qualitativa* verifica a corretude lógica do modelo, através da investigação de suas propriedades gerais, tais como a *consistência léxica*, que diz respeito à existência dos objetos necessários para a realização de uma tarefa; e a *consistência sintática*, que trata do seqüênciamento das ações disponíveis no modelo. Na *análise quantitativa* o objetivo é efetuar medidas de desempenho do sistema, tais como a freqüência de ações, quantidade de passos (ações) necessários para realizar uma tarefa, consistência temporal, etc.

São encontradas na literatura várias propostas de formalismos para a representação do componente de interface de um sistema interativo. Dentre os formalismos existentes, um formalismo muito adotado para a modelagem e análise de interfaces são as redes de Petri [23].

A adoção das redes de Petri é justificada pela sua capacidade de descrever paralelismo, concorrência, assincronismo e não-determinismo, que são características típicas de interfaces gráficas. As Redes de Petri também permitem especificar a evolução entre estados do sistema em resposta as ações do usuário (possibilitando uma análise qualitativa do sistema modelado). Além do que, os modelos criados são passíveis de execução, isto é, permitem simular o comportamento do sistema modelado. E, dispõem de uma ferramenta computacional de apoio para a construção, simulação e análise dos modelos.

São vários os trabalhos de modelagem de interfaces tendo como base o formalismo das redes de Petri, dentre os quais Palanque e Bastide [4,5,24,25,26,27]; Li, Mugridge e Hosking [22]; Palanque, Bastide e Sengès [29,30]; Palanque, Bastide, Sibertin e Dourte [32]; Rosis, Pizzutilo e De Carolis [36,37]; Schlungbaum e Elwert [38,39]; Sousa [40] e Sousa e Turnell [41], os quais serão brevemente discutidos a seguir

Palanque e Bastide em [4,5,25,27], e Palanque, Bastide e Sengès em [30] apresentam um formalismo denominado ICO (Interactive Cooperative Objects) que é utilizado para especificar, projetar e implementar interfaces homem-máquina de sistemas interativos. O formalismo ICO é uma linguagem especialmente projetada para modelar e implementar interfaces homem-máquina dirigidas por eventos. Esse formalismo utiliza os conceitos de linguagem orientada a objetos (tais como objetos, atributos, entidades, etc.)

para definir os elementos do sistema e uma rede de Petri de alto nível para definir o comportamento do sistema.

Palanque e Bastide em [26], e Palanque, Bastide e Sengès em [29] mostram o uso do formalismo ICO na modelagem e análise de sistemas interativos de software em geral. Neste trabalho é abordado o modelo do sistema propriamente dito e das tarefas a serem executadas pelo usuário.

Palanque e Bastide em [24] apresentam a modelagem e análise em Redes de Petri Coloridas (CPN) de sistemas críticos de segurança enfatizando a interação homem-máquina (a interface do sistema).

Schlungbaum e Elwert em [38,39] apresentam uma notação própria utilizada para a modelagem de sistemas de interface com base em redes de Petri Colorida.

Palanque, Bastide, Sibertin e Dourte em [32] propõem uma extensão de redes de Petri denominada PNO (*Petri Net with Objects*) para modelar o diálogo em interfaces.

Rosis, Pizzutilo e De Carolis em [36,37] apresentam um formalismo visual, baseado em redes de Petri, denominado XDM, para o projeto e avaliação da interação em sistemas voltados para contextos específicos (interfaces cooperativas com usuário sem familiarização com o uso de computadores). Os autores também apresentam uma ferramenta para o projeto e análise de interações homem-máquina.

Em [22] têm-se a descrição da linguagem PUIST (Petri Net Based Graphical User Interface Specification Toll) que é uma linguagem visual baseada em Redes de Petri para especificar a forma estática e o comportamento dinâmico de interfaces homem-máquina gráficas.

Sousa em [40] e Turnell e Sousa em [41] propõem um método de projeto de interface do usuário que inicia pela identificação do perfil do usuário seguido por uma sequencia de passos, o primeiro dos quais é a análise da tarefa. O método segue com a descrição do modelo da interação que especifica, entre outros aspectos, a estrutura de navegação da interface. Esta estrutura de navegação é modelada em CPN e verificada de acordo com um conjunto de propriedades especificadas para garantir sua usabilidade. O modelo é uma representação genérica da estrutura de navegação de uma interface do usuário, podendo ser adaptado para diferentes contextos, reduzindo assim o tempo de projeto e análise. Este modelo tem o objetivo de sintetizar, independente dos dispositivos de interação, as situações de transição mais comuns em uma interface com o usuário.

Do ponto de vista das interfaces para programas de controle industrial, praticamente não se encontram trabalhos relacionados com modelagem e análise. Apenas dois modelos foram encontrados. Um destes modelos [13], construído no formalismo rede de Petri lugar-transição, segundo seus autores não se mostrou adequado à modelagem de interfaces, dada sua complexidade e extensão, como resultado das características redundantes das interfaces. O segundo modelo [12], proposto com o formalismo das CPN, embora se proponha a ser uma representação genérica dessa classe de interfaces, apresentou erros em sua definição, tais como a inexistência de elementos na estrutura da rede (arcos sem inscrições) e o uso inapropriado de CPNs e redes de Petri Lugar-Transição, o que inviabilizou o seu uso.

4.6 Considerações Finais

Neste capítulo foram apresentadas as estratégias de avaliação do componente interface de um sistema computacional interativo. Foi dada uma atenção especial à *análise formal* enquanto metodologia adotada para a realização deste trabalho.

Deve-se ressaltar que 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 exceção daquele proposto em [40,41].

Embora praticamente inexistam trabalhos sobre a modelagem e análise de interfaces industriais, existem na literatura diversos trabalhos sobre modelagem e análise de planta industriais, com controle via programa supervisório [9,10,45,46]. Assim, propõe-se a modelagem da interface com o mesmo formalismo visando uma integração destes modelos de modo a possibilitar uma análise global do sistema.

Uma vez que o foco da avaliação de uma interface é a análise da sua adequação ao usuário, a avaliação baseada apenas em métodos formais não é suficiente, sendo necessária também a realização de testes de usabilidade com representantes da comunidade de usuários. Com o resultado desta avaliação podem ser detectados problemas não avaliados através da análise formal, a exemplo de falhas no projeto visual, nas estratégias de interação, na escolha de dispositivos de interação, etc.

Capítulo 5 – Modelo de Navegação e Contexto Industrial Modelado

5.1 Introdução

Este capítulo aborda o modelo de navegação adotado neste trabalho [40,41]. Será detalhada sua estrutura e serão discutidas as análise que foram realizadas neste modelo.

Também será apresentada a interface homem-máquina do ambiente industrial modelada neste trabalho.

5.2 Modelo de Navegação

O modelo de navegação foi construído de forma a sintetizar as situações de transição mais comuns em uma interface com o usuário (independente dos dispositivos de interação) [40]. Essa independência de dispositivo implica que a execução de uma ação independe do mecanismo de interação ou dispositivo utilizado. Segundo este modelo, a transição entre estados da interface no decorrer de uma interação pode ser de forma direta ou indireta, condicional ou incondicional, mantendo o contexto atual da interação, causando uma mudança de contexto ou restaurando o contexto anterior [42].

A seguir são descritas as situações que, de forma genérica, representam uma navegação na interface. Esta navegação está associada à transição entre estados da interface durante uma interação.

- *Retorno na interação*: Esta situação corresponde à navegação de um ponto qualquer para um ponto anterior na interação. Pode resgatar o contexto anterior (*undo*) ou manter os resultados das operações já realizadas.
- *Avanço na interação*: Navegação para um ponto subseqüente na interação. Pode ser: *incondicional*, que é a transição entre pontos da interação independentemente

do estado atual da interação; ou *condicional*, que é a navegação de um ponto da interação para outro subseqüente, ou para um ponto qualquer, dependendo de uma condição a ser testada.

- *Retorno ao início da interação*: Navegação de um ponto qualquer da interação para o ponto inicial da interação.
- *Saída do sistema*: Navegação de um ponto qualquer para o estado final da interação.

5.2.1 Apresentação do Modelo

Na Figura 11 é apresentado o modelo de navegação, aplicado ao contexto da navegação em um browser. A interface do browser se constitui de janelas, botões e menus. Neste modelo se associa ao contexto da interação seu estado atual e um histórico de seus estados anteriores.

Um estado de interação, neste contexto, é representado pela janela ativa e por uma lista de ações que podem ser realizadas neste ponto da interação. A interação do usuário com o sistema resulta em uma navegação entre janelas e/ou na atualização das opções disponíveis.

A seguir é apresentado o conjunto de cores deste modelo.

- *WINDOW*: representa o conjunto das janelas existentes.
- *BUTTON*: representa o conjunto dos botões existentes.
- *MENU*: representa o conjunto dos menus existentes.
- *OP_MENU*: representa o conjunto das opções de menu.
- *ACTION*: representa a união de um botão ou uma opção de menu.
- *ACTWIN*: cor auxiliar representando um par (ação, janela).
- *ACTION_LIST*: representa uma lista de ações, que contém botões ou opções de menu disponíveis.
- *AA_LIST*: cor auxiliar representando um par (ação, lista de ações).
- *INTERFACE*: representa um par (janela, lista de ações).
- *INTERFACE_LIST*: representa uma lista de interfaces.
- *UNDO_LIMIT*: representa o limite de elementos para a lista de interfaces.

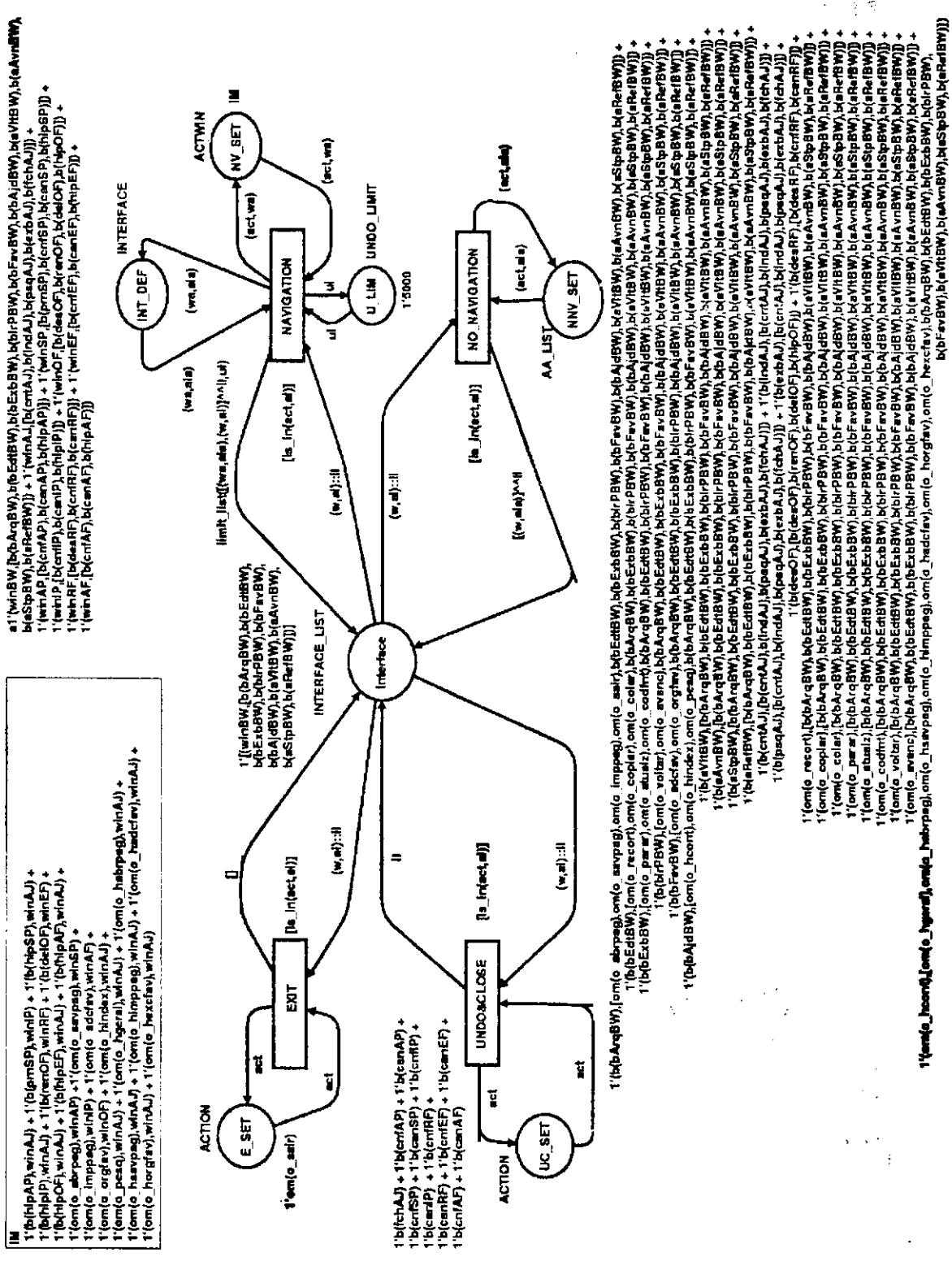


Figura 11 - Modelo de Navegação

Neste modelo, o lugar *Interface* contém uma lista com o primeiro termo correspondendo a interface ativa com suas opções de navegação e os demais termos representando os passos de navegação realizados anteriormente (ordem inversa de

ocorrência). No modelo são quatro as possibilidades de navegação (representadas por suas quatro transições), que são:

- Navegação entre janelas, representada pela transição *NAVIGATION*.
- Navegação dentro de uma mesma janela, representada pela transição *NO_NAVIGATION*.
- Fechamento de janela ou realização de *undo*, representada pela transição *UNDO&CLOSE*.
- Encerramento do programa, representado pela transição *EXIT*.

Como pode ser observado, cada uma dessas transições possui um ou mais lugares associados além do lugar *Interface*. E esses lugares representam as opções que podem ocorrer quando uma dessas transições dispara.

A transição *NAVIGATION* possui, além do lugar *Interface*, os seguintes lugares associados:

- *U_LIMIT*: Possui fichas da cor *UNDO_LIMIT*. Indica o limite máximo de undos[3] que o sistema pode realizar.
- *INT_DEF*: Possui fichas da cor *INTERFACE*. Possui a descrição de todas as janelas do sistema.
- *NV_SET*: Possui fichas da cor *ACTWIN*. Essas fichas são composta por duplas onde o primeiro elemento indica uma janela e o segundo a ação que deve ser executada para navegar para essa janela.

Para a transição *NAVIGATION* disparar, a ação a ser realizada (que no caso é a seleção de uma opção ou o pressionamento de um botão dentro das opções disponíveis na janela ativa) deve "casar" com uma das ações existentes no lugar *NV_SET* (a ação é o segundo membro da dupla), fazendo com que a janela relacionada com essa ação (primeiro elemento da dupla) seja adicionada à lista de interface no lugar *Interface*. A janela a ser acrescentada a lista é na verdade uma cópia da descrição da janela, contida no lugar *INT_DEF*.

A transição *NAVIGATION* possui uma guarda, que testa a função *is_in*. Esta função verifica se o elemento de interação que se deseja selecionar existe na lista de opções da janela ativa.

---

[3] A limitação deste recurso tem como objetivo evitar uma explosão de estados do modelo. Por outro lado, esta limitação permite representar uma situação prática na qual sempre haverá um limite no número de *undos* que o sistema vai disponibilizar ao usuário.

Existe um limite para o tamanho da lista no lugar *Interface*, e este limite é definido pela função *limit_list* que está no arco de saída da transição *NAVIGATION* para o lugar *Interface*.

A transição *NO_NAVIGATION* possui o lugar *NNV_SET* associado, além do lugar *Interface*. *NNV_SET* possui fichas da cor *AA_LIST*. É neste lugar que se encontram as ações que não provocam navegação.

Para a transição *NO_NAVIGATION* tem-se que o conjunto de opções do primeiro elemento da lista do lugar *Interface* é confrontado com uma das opções de navegação dentro de uma página contida no lugar *NNV_SET* (através da guarda da transição que executa a função *is_in*). Se existir uma opção comum, a transição dispara indicando que a opção foi selecionada. Essa seleção pode causar uma mudança nas opções disponíveis (seleção de um menu disponibiliza as opções relacionadas a esse menu) ou apenas indicar a seleção de uma opção (pressionamento de um botão simples).

Já a transição *UNDO&CLOSE* é utilizada para fechar uma janela ou voltar a um ponto anterior na navegação. Nesse modelo, essas duas possibilidades levam ao mesmo resultado, uma vez que disparada essa transição temos a retirada do primeiro item da lista no lugar *Interface*, nos levando ao estado imediatamente anterior na navegação. O lugar associado a essa transição é o *UC_SET*, que possui fichas da cor *ACTION*, que indicam as ações que provocam retorno na interação.

Para *UNDO&CLOSE* ocorrer, uma das opções disponíveis no lugar *UC_SET* deve "casar" com uma das opções do conjunto de opções contidas no primeiro elemento da lista no lugar *Interface* (teste feito pela função *is_in* que está contida na guarda desta transição).

E, a transição *EXIT* é utilizada para encerrar o aplicativo. Observe que seu lugar associado (o lugar *E_SET*, cujas fichas são da cor *ACTION*) só possui uma única ficha, que é a opção de sair do aplicativo. Essa opção deve "casar" com uma das opções da lista de opções contidas no primeiro elemento da lista de interfaces (através da guarda da transição, que contém a função *is_in*), habilitando a transição. E, com o disparo dessa transição, temos que o lugar *Interface* recebe uma lista vazia, causando um impasse no sistema, que indica o encerramento do aplicativo.

5.3.3 JanBatch

Essa janela pode ser aberta a partir das janelas *JanReactor*, *JanHist* ou *JanConveyor*, e ao ser aberta se sobrepõe à janela onde foi aberta. Nessa janela existem vários botões que servem para ajustar o nível de concentração da mistura do sistema, um botão para fechá-la e um botão para abrir a janela *JanBatchSet*. Apresentamos a seguir uma descrição desses botões:

- *BotBatchExit*: Esse botão fecha a janela *JanBatch*, retornando para a janela a partir da qual *JanBatch* foi aberta.
- *BotBatchSet*: Esse botão abre a janela *JanBatchSet*, que se sobrepõe à janela *JanBatch*.
- *BotBatchUp*: Esse botão é utilizado para incrementar a concentração da mistura ao passo de 1%.
- *BotBatchDown*: Esse botão é utilizado para decrementar a concentração da mistura ao passo de 1%.
- *BotBatchBarra*: Essa opção é na verdade uma barra deslizante que é utilizada para incrementar/decrementar a concentração da mistura em passos de tamanho variável.
- *BotBatchOk*: Esse botão é utilizado para setar a concentração da mistura para o valor definido utilizando os outros botões. Se esse botão não for selecionado e a janela for fechada, o novo valor de concentração não vai ser atualizado.

5.3.4 JanBatchSet

Essa janela ao ser aberta se sobrepõe à janela *JanBatch*. Essa janela é utilizada para entrar com um valor mais preciso de concentração da mistura. Nesta janela a entrada não é por meio de botões, mas pelo teclado. Ela oferece apenas um botão (*BotBatchSetExit*), que é utilizado para fecha-la e retornar à janela *JanBatch*. Se ao fechar essa janela, não for selecionado o botão *BotBatchOk* na janela *JanBatch*, o valor de concentração não será atualizado.

5.3.5 JanHist

Essa janela mostra um gráfico do desenvolvimento do sistema de mistura e armazenagem da concentração. Temos como opções de navegação nessa janela os seguintes botões:

- *BotHistHelp*: Usado para abrir a janela *HistHelp*.
- *BotHistCTime*: Utilizado para solicitar a informação do tempo atual do processo.
- *BotHistDefault*: Utilizado para restabelecer os valores padrão para a geração do gráfico.
- *BotHistV100*: Utilizado para definir o limite superior do gráfico como 100% (o limite inferior é sempre zero).
- *BotHistV50*: Utilizado para definir o limite superior do gráfico como 50%.
- *BotHist3m*: Utilizado para definir a taxa de atualização do gráfico para 3 minutos.
- *BotHist1m*: Utilizado para definir a taxa de atualização do gráfico para 1 minuto.
- *BotHistBarraVert*: É na verdade uma barra deslizante utilizada para facilitar a observação do valor de determinado ponto no gráfico em relação à concentração (escala vertical). Para exibir o valor em um ponto, desloca-se a barra para o ponto e verifica-se o valor na escala vertical.
- *BotHistBarraHor*: É na verdade uma barra deslizante utilizada para observar o valor de determinado ponto no gráfico em relação ao tempo (escala horizontal).
- *BotReactor*: Esse botão faz com que a janela *JanHist* seja fechada e seja aberta a janela *JanReactor*.
- *BotBatch*: Esse botão faz com que a janela *JanBatch* seja aberta e sobreposta a janela *JanHist*.
- *BotHist*: Esse botão é utilizado para navegar para a janela *JanHist*, mas como essa já é a janela ativa esse botão está indisponível.
- *BotMaintenance*: Esse botão faz com que a janela *JanMaintenance* seja aberta e sobreposta à janela *JanHist*.
- *BotConveyor*: Esse botão faz com que a janela *JanHist* seja fechada e seja aberta a janela *JanConveyor*.
- *BotAlarme*: Esse botão faz com que a janela *JanAlarme* seja aberta e sobreposta à janela *JanHist*.

5.3.6 JanHistHelp

Essa janela ao ser aberta sobrepõe a janela *JanHist* e apresenta uma caixa de texto explicando o funcionamento do gráfico, e um botão (*BotHistHelpExit*) que fecha essa janela tornando a janela *JanHist* novamente ativa.

5.3.7 JanMaintenance

Essa janela pode ser aberta a partir das janelas *JanReactor*, *JanHist* ou *JanConveyor*. Ao ser aberta se sobrepõe à janela onde foi solicitada. É uma janela que indica o estado de algumas partes do sistema (é uma janela de mensagem).

Oferece apenas um botão (*BotMaintExit*), que fecha a janela *JanMaintenance*, retornando para a janela a partir da qual foi aberta.

5.3.8 JanConveyor

Essa janela mostra por meio de animação o controle do enchimento de tonéis de mistura. Esse processo pode ser feito automaticamente ou manualmente. A seleção é feita através de um botão (*BotConveyorMA*) que alterna o estado de controle de automático para manual ou vice-versa. De acordo com o tipo de controle do processo, algumas opções de navegação disponíveis ficam disponíveis.

Os botões que fazem parte dessa janela são descritos a seguir:

- *BotConveyorBarra*: Esse botão serve para controlar a velocidade das esteiras do processo.
- *BotConveyorMA*: Utilizado para passar o processo de enchimento de tonéis do modo automático para manual ou vice-versa.
- *BotConveyorEst1*: Esse botão só está ativo se o processo de enchimento dos tonéis estiver no modo manual. Ele é utilizado para iniciar o enchimento de um novo tonel.
- *BotConveyorEst2*: Esse botão só está ativo se o processo de enchimento dos tonéis estiver no modo manual. Ele é utilizado para encher o tonel com a mistura.
- *BotConveyorEjet*: Esse botão só está ativo se o processo de enchimento dos tonéis estiver no modo manual. Ele é utilizado retirar o tonel da esteira, finalizando o processo de enchimento.
- *BotReactor*: Esse botão faz com que a janela *JanConveyor* seja fechada e seja aberta a janela *JanReactor*.
- *BotBatch*: Esse botão faz com que a janela *JanBatch* seja aberta e sobreposta à janela *JanConveyor*.
- *BotHist*: Esse botão faz com que a janela *JanConveyor* seja fechada e seja aberta a janela *JanHist*.

- *BotMaintenance*: Esse botão faz com que a janela *JanMaintenance* seja aberta e sobreposta à janela *JanHist*.
- *BotConveyor*: Esse botão é utilizado para navegar para a janela *JanConveyor*, mas como essa já é a janela ativa esse botão está indisponível.
- *BotAlarme*: Esse botão faz com que a janela *JanAlarme* seja aberta e sobreposta à janela *JanHist*.

5.3.9 JanAlarme

Essa janela pode ser aberta a partir das janelas *JanReactor*, *JanHist* ou *JanConveyor*. Ao ser aberta sobrepõe a janela a partir da qual foi aberta. Essa janela indica a ocorrência de alarmes dentro da planta industrial. Inicialmente, os botões de alarme estão indisponíveis, e ao ocorrerem alarmes esse botões vão se tornando ativos. Os botões presentes nesta janela são:

- *BotAlarmeClose*: Utilizado para fechar a janela *JanAlarme* e retornar para a janela a partir da qual *JanAlarme* foi requisitada.
- *BotAlarmeConc*: Esse botão se torna ativo quando ocorre um alarme de concentração da mistura. Ao ser selecionado ele é desativado, indicando que o alarme foi reconhecido.
- *BotAlarmeTemp*: Esse botão se torna ativo quando ocorre um alarme de temperatura da mistura. Ao ser selecionado ele é desativado, indicando que o alarme foi reconhecido.
- *BotAlarmeNível*: Esse botão se torna ativo quando ocorre um alarme de nível da mistura. Ao ser selecionado ele é desativado, indicando que o alarme foi reconhecido.
- *BotAlarmeAckAll*: Esse botão se torna ativo quando os três botões de alarme (*BotAlarmeConc*, *BotAlarmeTemp* e *BotAlarmeNivel*) se tornarem ativos. Ao ser selecionado, ele e os três botões de alarme, são desativos (indicando que todos os alarmes foram reconhecidos). Esse botão reconhece todos os alarmes de uma só vez. Mas, se estiver ativo e um dos outros três botões de alarme (*BotAlarmeConc*, *BotAlarmeTemp* ou *BotAlarmeNível*) for reconhecido, tanto esse botão quanto o botão *BotAlarmeAckAll* vão ser desativados.

5.4 Considerações Finais

Neste capítulo foi apresentado o modelo de navegação construído com o formalismo das CPN, e foram descritas as análises realizadas. A explicação detalhada foi necessária uma vez que este modelo serviu de base para a construção dos demais modelos propostos.

Também foi descrita a interface homem-máquina de um programa supervisório industrial utilizada para a verificação da adequação do modelo de navegação.

Considerando que o modelo se mostrou adequado ao estudo da navegação no contexto do *browser*, segundo seus autores [40,41]; cabe verificar sua adequação ao estudo da navegação no contexto de interfaces industriais, assunto tratado no próximo capítulo.

Capítulo 6 – Modelo CPN da Interface Industrial

6.1 Introdução

Neste capítulo será apresentado o modelo da interface industrial construído a partir do modelo de navegação. Este modelo contempla mecanismos de navegação que o modelo de navegação, originalmente concebido, não incorpora (a sobreposição de janelas), bem como a representação do estado dos elementos da interação (os quais podem estar disponíveis ou indisponíveis na janela ativa, dependendo do contexto).

Este modelo será detalhado em sua estrutura e comportamento. Serão apresentadas as análises realizadas no modelo, como também em um cenário de navegação na interface sob a forma de um diagrama de seqüência de mensagem (*MSC – Message Sequence Chart*) [2].

6.2 Modelo CPN da Interface Industrial

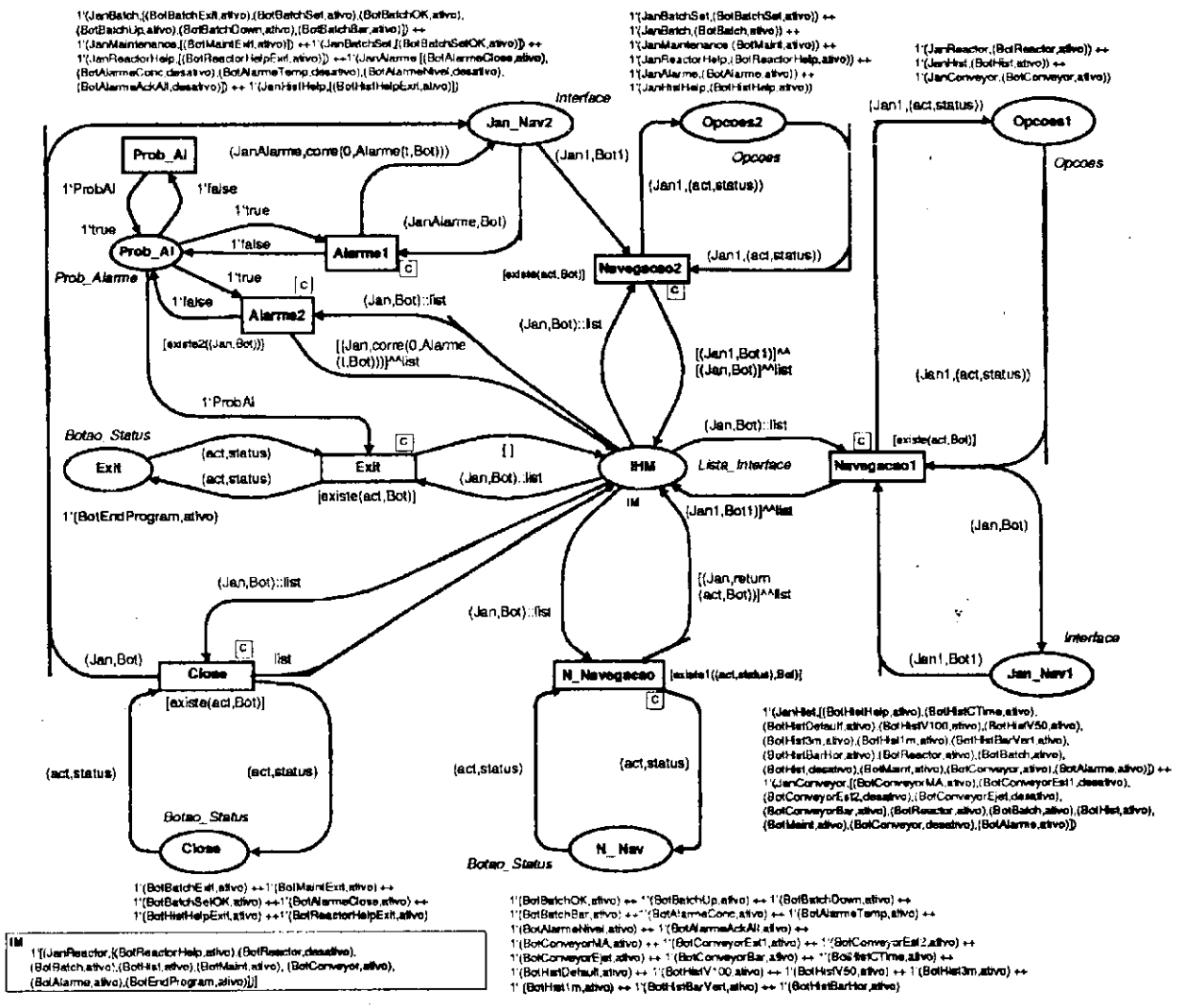
No processo de modelagem da interface descrita no capítulo anterior, verificou-se que o modelo de navegação necessitava alterações para ser aplicado a este contexto. Observou-se que nessa interface existiam dois tipos de navegação entre janelas, uma navegação que fecha a janela ativa e abre outra janela, e uma navegação que sobrepõe uma nova janela sobre a janela atualmente ativa; não sendo este último tipo tratado no modelo de navegação original. Com isso o modelo original teve que ser modificado para atender a esses novos requisitos.

Outra alteração necessária foi a inclusão da geração de alarmes aleatórios, para os três tipos de alarmes tratados na interface industrial modelada. A geração de alarmes ocorre independentemente da janela *JanAlarme* estar ativa.

Foi também necessária a incorporação do estado dos botões, para representar os estados ativo ou desativo, indicando a disponibilidade ou indisponibilidade do botão para a interação. No modelo de navegação todas as opções estavam sempre ativas.

Assim foi necessária a criação de funções no modelo para tratar esses novos requisitos. Foram criadas funções para a geração de alarmes (que mudam o estado dos botões de alarme), para o reconhecimento dos alarmes, para a mudança do estado dos botões da janela *JanConveyor* para o modo automático ou manual, e para verificação da existência e disponibilidade na janela ativa da opção de navegação que desejamos ativar. Essa funções serão melhor detalhadas no transcorrer deste capítulo.

Na figura 14 é apresentado o modelo CPN da interface industrial. E, em seguida é listado o nó de declaração:



- *Exit*: Esta transição modela o encerramento da execução do aplicativo.

Além dessas transições de navegação existem mais três transições utilizadas para a geração do alarme aleatório (*Alarme1*, *Alarme2* e *Prob_Al*).

Nas subseções seguintes serão detalhados os tipos de navegação e a geração do alarme.

6.2.1 Navegação entre Janelas Fechando a Janela Ativa

Esse tipo de navegação é modelado pela transição *Navegacao1* e os lugares *Jan_Nav1* e *Opcoes1*. Estes elementos estão descritos a seguir.

- *Jan_Nav1*: Possui fichas da cor *Interface*. As fichas contidas neste lugar indicam as janelas que ao serem abertas fecham a janela ativa, se tornando assim a janela ativa atual.
- *Opcoes1*: Possui fichas da cor *Opcoes*. Suas fichas relacionam uma ação de navegação (seleção de um botão) com uma janela, ou seja, indica qual janela deve ser aberta se uma dada opção for selecionada na janela ativa (se essa opção fizer parte das opções de navegação que causam a navegação com o fechamento da janela ativa).
- *Navegacao1*: Transição que modela o processo de navegação entre janelas fechando a janela ativa.

A transição *Navegacao1* tem como parâmetros de entrada o primeiro elemento da lista (que representa a janela ativa e suas opções de navegação) que está no lugar *IHM*, uma das fichas do lugar *Opcoes1* (que indica a ação a ser tomada na janela ativa e a nova janela a ser aberta) e uma das fichas do lugar *Jan_Nav1* (que contém a janela a ser aberta). Essa transição tem com guarda a função *existe* (explicitada no nó de declaração), cuja finalidade é verificar se a ação de navegação (opção de navegação) pretendida existe na janela ativa. Sendo verdadeira essa verificação a transição ocorre, ocasionando o fechamento da janela atual (o elemento que foi retirado da lista será depositado no lugar *Jan_Nav1*) e a abertura da nova janela (inserção do elemento que define a nova janela, na cabeça da lista no lugar *IHM*).

6.2.2 Navegação entre Janelas com a sobreposição de Janelas

Essa navegação é modelada pela transição *Navegacao2* e os lugares *Jan_Nav2* e *Opcoes2*. Segue uma descrição desses elementos.

- *Jan_Nav2*: Possui fichas da cor *Interface*. Suas fichas indicam as janelas que ao serem abertas se sobrepõem a janela ativa se tornado a nova janela ativa.
- *Opcoes2*: Possui fichas da cor *Opcoes*. As fichas contidas neste lugar relacionam uma ação de navegação (seleção de um botão) com uma janela. Ou seja, indicam qual janela deve ser aberta e sobreposta a atual se uma dada opção for selecionada na janela ativa (se essa opção fizer parte das opções de navegação que causam a navegação com o fechamento da janela ativa).
- *Navegacao2*: Transição que modela o processo de navegação entre janelas com a nova janela se sobrepondo à janela ativa.

A transição *Navegacao2* possui como parâmetros de entrada a cabeça da lista que está no lugar *IHM*, uma das fichas do lugar *Opcoes2* (que indica a ação a ser tomada na janela ativa e a nova janela a ser aberta) e uma das fichas do lugar *Jan_Nav2* (que contém a janela a ser aberta). Essa transição (assim como a transição *Navegacao1*) tem com guarda a função *existe*. Sendo verdadeira a verificação realizada pela função *existe*, a janela atual será sobreposta pela nova janela (a nova janela será adicionada na cabeça da lista no lugar *IHM*).

6.2.3 Fechamento da Janela Ativa

A transição *Close* e o lugar *Bot_Close* modelam essa navegação. O significado desses elementos é descrito a seguir.

- *Bot_Close*: Possui fichas da cor *Botao_Status*. Este lugar possui fichas que relacionam uma ação de navegação ao fechamento de uma janela (seleção de um botão de fechamento de janela).
- *Close*: Esta transição modela o processo de fechamento de uma janela que está sobreposta a outra, tornando a janela que estava em segundo plano a nova janela ativa.

A transição *Close* tem como parâmetros de entrada a cabeça da lista que está no lugar *IHM* e uma das fichas do lugar *Bot_Close* (que indica a ação a ser tomada na janela ativa para fecha-la). Esta transição (assim como a transição *Navegacao1*) tem com guarda a função *existe*. Sendo verdadeira a verificação feita pela execução da função *existe*, a janela ativa é fechada (é retirada da cabeça da lista do lugar *IHM* e devolvida ao lugar *Jan_Nav2*), retornando à janela a partir da qual essa janela tinha sido requisitada (nova cabeça da lista na janela *IHM*).

6.2.4 Encerramento da Execução do Aplicativo

Este tipo de navegação é modelado pela transição *Exit* e pelo lugar *Bot_Exit*. Esses dois elementos são descritos a seguir.

- *Bot_Exit*: Possui uma única ficha da cor *Botao_Status*. Essa ficha indica a ação de navegação (seleção de um botão) que encerra a execução do aplicativo.
- *Exit*: Esta transição modela o processo de encerramento da execução do aplicativo.

A transição *Exit* tem como parâmetros de entrada a cabeça da lista que está no lugar *IHM*, a ficha do lugar *Jan_Exit* (que indica a ação a ser tomada na janela ativa para encerrar o aplicativo) e a ficha do lugar *Prob_Al* (para impedir que o modelo continue gerando os alarmes após o encerramento do aplicativo). Essa transição (assim como a transição *Navegacao1*) tem com guarda a função *existe*. Sendo verdadeira a verificação feita pela execução da função *existe*, o aplicativo é encerrado (a lista no lugar *IHM* é devolvida vazia e o lugar *Prob_Al* fica com marcação vazia).

6.2.5 Seleção de Opção dentro da Janela Ativa

A transição *N_Navegacao* e o lugar *Bot_N_Nav* modelam a seleção de uma opção. Esse lugar e essa transição são explicados a seguir.

- *Bot_N_Nav*: Possui fichas da cor *Botao_Status*. Suas fichas relacionam uma ação de navegação (seleção de um botão) com uma janela, ou seja, indica quais opções existentes em uma janela do sistema resultam em uma navegação dentro da mesma janela. Observe que todas estas fichas tem os botões com o estado ativo, indicando que essa opção deve estar ativa para poder ser selecionada.
- *N_Navegacao*: Essa transição modela o processo de navegação dentro de uma mesma janela.

A transição *N_Navegacao* tem como parâmetros de entrada a cabeça da lista do lugar *IHM* e uma das fichas do lugar *Bot_N_Nav* (que indica a ação ser tomada dentro da janela). Aqui essa transição tem como guarda a função *existe1*, que testa se o elemento de interação que se deseja está na lista (como ocorre na função *existe*) e se o estado desse elemento é ativo, para que possa ser selecionado.

Com a ocorrência desta transição existem quatro possibilidades de acontecimentos no retorno da interface ativa à lista do lugar *Interface*, o que é decidido pela função *retorno* presente no arco de saída da transição que vai para o lugar *Interface*:

- Caso a ação a ser tomada seja a seleção do botão *BotConveyorMA*, os botões *BotConveyorEst1*, *BotConveyorEst2* e *BotConveyorEjet* irão mudar seu estado (de desativo para ativo ou vice-versa). Essa mudança é feita pela evolução da função *muda_status2* sempre que *BotConveyorMA* for selecionado.
- Caso a ação a ser tomada seja a seleção do botão *BotAlarmeConc*, ou *BotAlarmeTemp* ou *BotAlarmeNivel*, o botão selecionado terá seu estado alterado para desativo (observe que para uma opção estar disponível para seleção tem que estar com o seu estado no modo ativo). Essa mudança é feita a partir da função *muda_status* que é executada sempre que um dos botões de alarme (a menos do *BotAlarmeAckAll*) for selecionado. Quando a função *muda_status* é executada, além de mudar o estado do botão para desativo (reconhecer o alarme), ela também executa a função *test_AckAll* que verifica se o botão *BotAlarmeAckAll* está no estado ativo, desativando-o neste caso.
- Caso a ação a ser tomada seja a seleção do botão *BotAlarmeAckAll* (disponível apenas após a ocorrência dos três tipos possíveis de alarme, habilitando os botões *BotAlarmeConc*, *BotAlarmeTemp* e *BotAlarmeNivel*, sem ter ocorrido o reconhecimento de nenhum deles), os botões *BotAlarmeAckAll*, *BotAlarmeConc*, *BotAlarmeTemp* e *BotAlarmeNivel* terão seus estados alterados para desativo. Essa mudança é feita a partir da função *muda_status1* que ocorre sempre que o botão *BotAlarmeAckAll* for selecionado. Caso seja reconhecido um dos outros alarmes, tanto esse alarme quanto o botão *BotAlarmeAckAll* passarão para o estado desativo.
- Caso a ação a ser tomada seja a seleção de um botão que não muda seu estado nem o de nenhum outro botão da janela, a cabeça da lista é devolvida para o lugar *Interface* sem nenhuma alteração, indicando apenas que uma opção foi selecionada.

6.2.6 Geração do alarme aleatório

A geração de um alarme aleatório foi incorporada ao modelo para simular a geração de alarmes ocorridos devido a problemas na planta industrial monitorada. Neste trabalho são tratados três tipos de alarmes: um de concentração, um de temperatura e um de nível da mistura.

Ocorrendo um desses alarmes, o botão de reconhecimento de alarme correspondente deve mudar seu estado para ativo, indicando a ocorrência do alarme. Para a geração dos alarmes é utilizada uma estrutura formada pelos seguintes elementos:

- *Prob_Al*: Este lugar possui uma ficha (da cor *Prob_Alarme*) de valor booleano usada como restrição para a ocorrência dos alarmes. O alarme só pode ocorrer se o valor dessa ficha for *true*.
- *Prob_Al*: Transição que gera a probabilidade da ocorrência dos alarmes. Ela retira uma ficha do tipo *false* do lugar *Prob_Al* e retorna uma ficha com um valor booleano. Se a ficha for *true* o alarme pode ocorrer, se for *false* o alarme não pode ocorrer e essa transição ficará novamente habilitada.
- *Alarme1*: Transição utilizada para modelar a geração do alarme quando a janela *JanAlarme* está desativa.
- *Alarme2*: Transição que modela a geração do alarme quando a janela *JanAlarme* está ativa.

No caso da janela *JanAlarme* estar desativa, a transição modeladora da geração do alarme é *Alarme1*. Essa transição tem como parâmetros de entrada a ficha do lugar *Prob_Alarme* e a ficha do lugar *Jan_Nav2* que corresponde à descrição da janela *JanAlarme* e de suas opções de navegação. Com o disparo dessa transição é executada a função *corre* nas opções de navegação de *JanAlarme*. Esta função executa e testa o resultado da função *alarme*, e se os três tipos de alarmes já tiverem ocorrido, ela muda o estado do botão *BotAlarmeAckAll* para ativo.

A função *alarme* é responsável pela verificação do tipo de alarme que foi gerado (que é dado por um número aleatório entre um e três), tornando disponível o botão correspondente de reconhecimento de alarme.

No caso da janela *JanAlarme* ser a janela ativa, a transição que modela a geração do alarme é *Alarme2*. Essa transição recebe como elementos de entrada a cabeça da lista do lugar *IHM* e a ficha do lugar *Prob_Alarme*. Seu retorno é igual ao da transição *Alarme1* (executa a função *corre* sobre a lista de opções, que por sua vez executa a função *alarme* sobre essa mesma lista). Existem duas diferenças entre o que ocorre no disparo de *Alarme1* e *Alarme2*. Trata-se da execução de mais dois passos: o resultado é concatenado de volta à cabeça da lista; e existe uma guarda que executa a função *existe2*, a qual verifica se o primeiro elemento da lista é a descrição de *JanAlarme*.

6.2.7 Síntese do Modelo CPN para a Interface Industrial

Sintetizando, sempre que se deseje executar uma opção de navegação na interface ativa no modelo da interface industrial, essa opção é testada quanto à existência e disponibilidade (para o caso das opções que podem ter o estado desativo). Feita essa verificação a navegação ocorre. E esse ciclo se repete, intercalado pela geração aleatória do alarme, até a opção selecionada corresponda ao encerramento do aplicativo.

6.3 Análises Realizadas no Modelo

Foram realizadas várias análises no modelo para verificar a sua corretude lógica e seu comportamento. Foram realizadas simulações interativas e automáticas, gerado o grafo de ocorrência, e gerados MSCs para mostrar graficamente um cenário de navegação na interface. As análises realizadas com o auxílio da ferramenta *Design/CPN* [8], são descritas a seguir.

6.3.1 Simulações Interativas e Automáticas

No decorrer do desenvolvimento do modelo as simulações interativas foram utilizadas para verificar a corretude das partes do sistema. As partes testadas isoladamente foram referentes à navegação entre janelas fechando a janela atual, navegação entre janelas sobrepondo a janela atual com a nova janela, fechamento de janela, geração de alarmes (com a janela *JanAlarme* ativa ou desativa), reconhecimento de alarmes, mudança dos estados da janela *JanConveyor*, e encerramento do aplicativo.

O uso de simulações interativas em partes do sistema facilitou a verificação da evolução dessas partes do sistema. Para realizar essa análise as marcações iniciais dos lugares do sistema foram alteradas para que o sistema evoluísse apenas na parte que se desejava analisar.

Em seguida foram executadas simulações tanto interativas quanto automáticas no sistema como um todo para verificar sua evolução, observando os possíveis tipos de navegação ocorrer de forma integrada e garantindo a inexistência de impasses no sistema, exceto quando o aplicativo é encerrado.

6.3.2 Grafo de Ocorrência

Após verificar por meio de simulações interativas e automáticas que o sistema aparentemente evoluía corretamente, foi gerado o Grafo de Ocorrência (Occ) do modelo para verificar propriedades lógicas do sistema. O relatório do Grafo de Ocorrência gerado é mostrado a seguir.

```
------------------------------------------------------------------------------
Statistics
------------------------------------------------------------------------
  Occurrence Graph
    Nodes:   560
    Arcs:    3388
    Secs:    8
    Status:  Full

 Boundedness Properties
------------------------------------------------------------------------
  Best Integers Bounds            Upper  Lower
  IHM_Industrial'Bot_Close 1      6      6
  IHM_Industrial'Bot_Exit 1       1      1
  IHM_Industrial'Bot_N_ 1         21     21
  IHM_Industrial'IHM 1            1      1
  IHM_Industrial'Jan_Nav1 1       2      2
  IHM_Industrial'Jan_Nav2 1       6      4
  IHM_Industrial'Opcoes1 1        3      3
  IHM_Industrial'Opcoes2 1        6      6
  IHM_Industrial'Prob_Al 1        1      0

 Liveness Properties
------------------------------------------------------------------------
  Dead Markings:  16 [8,545,455,453,451,...]
  Dead Transitions Instances: None
------------------------------------------------------------------------------
```

O grafo gerado foi total, possuindo 560 nós e 3388 arcos. É interessante notar os limites superior e inferior nos lugares, pois eles coincidem com os esperados, dando mais uma garantia acerca da validade do modelo. Nos lugares *Bot_Close*, *Bot_Exit*, *Bot_N_Navigation*, *Bot_Opcoes1* e *Bot_Opcoes2*, o número máximo e mínimo de fichas é o mesmo. Isso era esperado, pois esses lugares possuem as opções disponíveis no sistema para efetuar navegação, e são apenas utilizadas para verificar se a ação existe na janela principal, retornando ao lugar após o teste.

O lugar *Jan_Nav1* possui no máximo e mínimo duas fichas. Isso é porque existem apenas três janelas que utilizam a navegação entre janelas fechando a janela atual. Então, para uma dessa janela ser aberta, uma outra deve estar no lugar *IHM* como a janela ativa. Ocorrendo a troca, uma janela foi retirada e uma outra foi adicionada no lugar *Jan_Nav1*.

Jan_Nav2 possui limite superior de seis fichas, o que indica que nenhuma das janelas que utiliza a navegação de sobreposição de janelas está ativa. E seu limite inferior

é quatro, indicando que a janela *JanBatch* foi sobreposta à janela ativa e em seguida foi aberta a janela *JanBatchSet* sobreposta a *JanBatch*.

O lugar *Prob_Al* possui limite superior uma ficha, que é a ficha necessária para o alarme aleatório ser gerado. E, no seu limite inferior ele não possui nenhuma ficha, que corresponde a situação do encerramento do aplicativo, impossibilitando a geração dos alarmes aleatórios.

Através da propriedade de vivacidade observa-se que não existem instâncias de transições mortas, indicando que todas as transições do sistema podem disparar pelo menos uma vez. Isso garante que todas as opções de navegação podem ocorrer.

Observa-se também que o sistema possui 16 marcações mortas. Essas marcações se devem às possibilidades dos botões cujos estados podem estar ativo ou desativo no momento do encerramento do aplicativo (quando ocorre a transição *Exit*, retorna uma lista vazia ao lugar *IHM* que causa um impasse no sistema). Existem três botões de alarme que podem variar seu estado, nos dando 2^3=8 possibilidades de estados. O botão de alarme *BotAlarmeAckAll* também muda seu estado, mas não entra nessa análise devido a sua mudança de estado estar vinculada aos outros três botões de alarme. Além destes botões existem os botões da janela *JanConveyor* que podem estar ativos ou desativos de acordo com o botão *BotConveyorMA*. Como esses botões mudam seus estados ao mesmo tempo, têm-se mais duas possibilidades. Combinando essas duas possibilidades às oito do caso dos alarmes temos 16 possibilidades de marcações mortas.

6.3.3 Diagrama de Seqüência de Mensagens

Os diagramas de seqüência de mensagens (*MSC – Message Sequence Charts*) consistem em uma linguagem gráfica e textual para a especificação e descrição das interações entre componentes do sistema. Eles contém uma descrição da comunicação assíncrona entre instâncias e podem ser utilizados como uma especificação geral do comportamento da comunicação de sistemas em tempo real. [2].

Usualmente os projetistas que analisam a especificação de interfaces não estão familiarizados com o formalismos de redes de Petri, assim, a explicação do comportamento destes componentes através de MSCs se torna uma abordagem muito útil. A representação de um evento por meio de uma mensagem é muito mais intuitiva do que a análise das marcações de um modelo CPN.

Um MSC básico contém uma descrição parcial do comportamento da comunicação entre um número de instâncias. Uma instância é uma entidade abstrata na qual pode-se observar parte da interação com outras instâncias ou com o meio-ambiente. Uma instância é denotada por uma barra vertical. O tempo ao longo de cada eixo corre do topo para a base [2].

Nesta análise do modelo foi gerado um MSC utilizando uma biblioteca do *Design/CPN*. O MSC foi particularmente útil na análise do comportamento do sistema sob condições específicas, tais como: após a ocorrência de um alarme, na navegação de um ponto a outro da interface, na mudança entre o modo manual e automático nos itens de *JanConveyor*, e para se obter uma visão geral do comportamento do sistema até o seu encerramento.

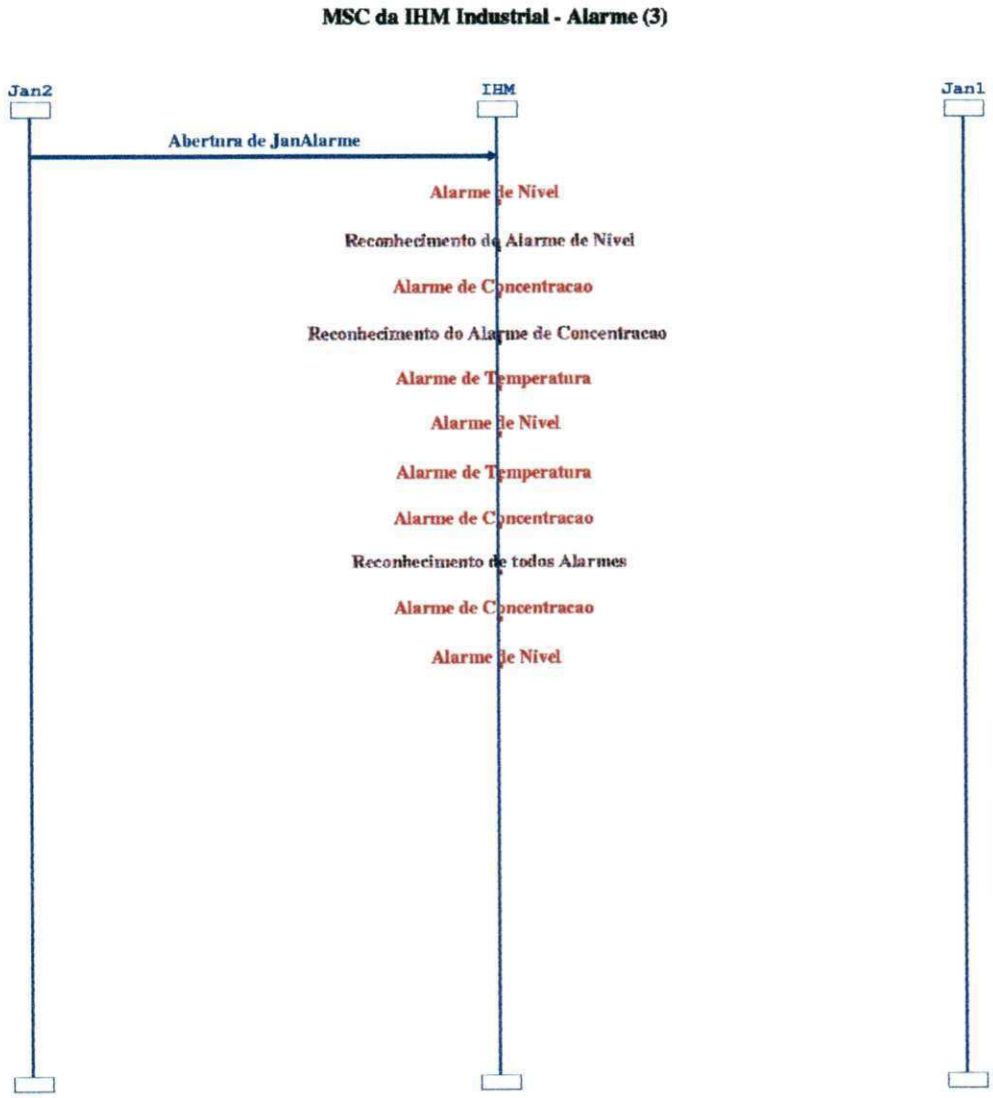


Figura 15 - MSC da Geração e Reconhecimento dos Alarmes

6.3.4 MSC de um Cenário de Navegação

Os MSCs das Figuras 16 e 17 ilustram o comportamento da interface industrial modelada em um possível cenário de navegação.

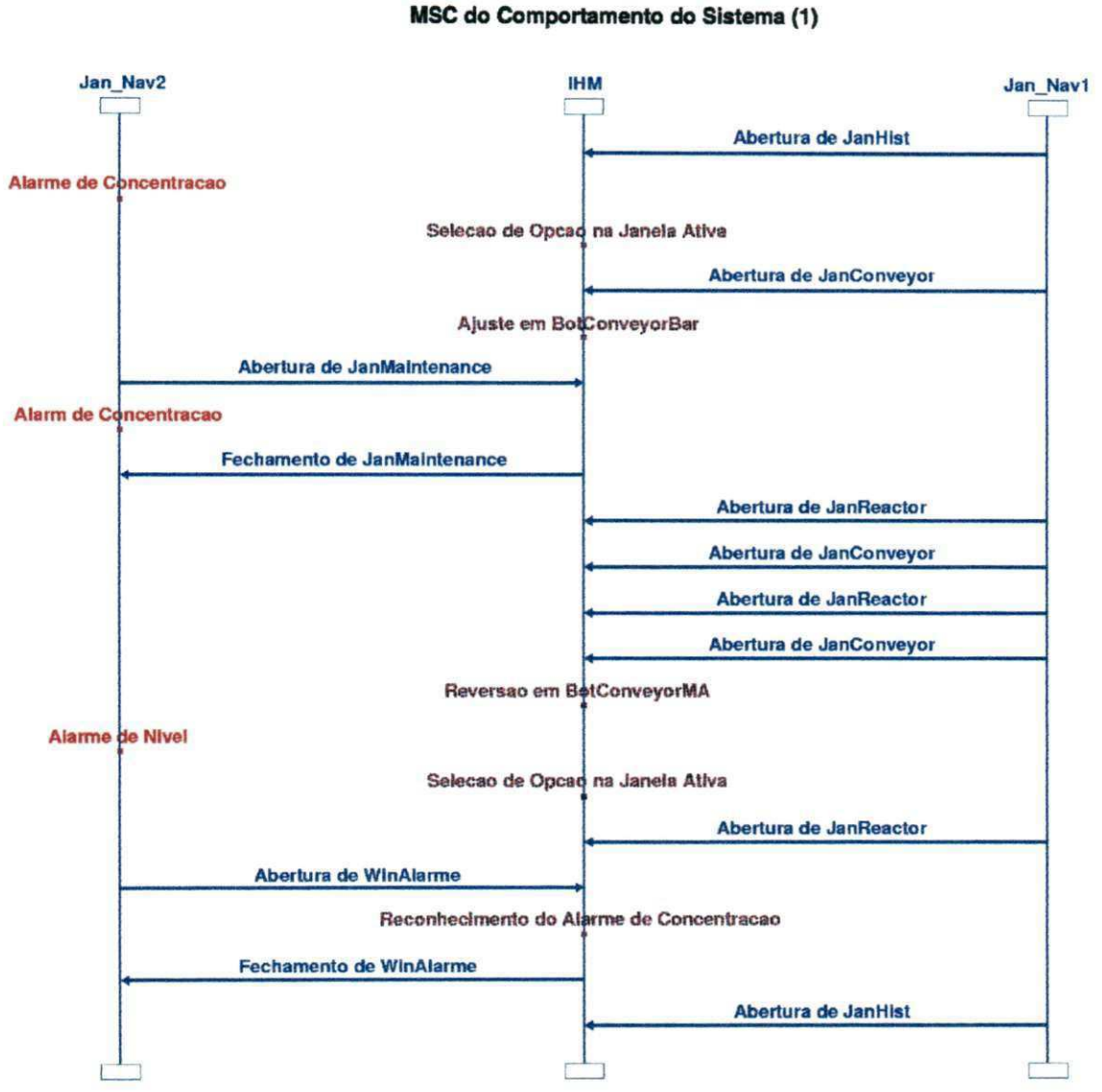


Figura 16 - MSC do Comportamento do Sistema (MSC1)

No MSC1, o operador entra no programa supervisório, na janela JanReactor. Então, checa o gráfico de tendência do processo abrindo a JanHist, onde uma das opções disponíveis é selecionada (um zoom). Neste instante ocorre um alarme de concentração da mistura. O operador ignora esse alarme e abre a janela JanConveyor para ajustar a velocidade do enchimento dos barris. Note que os alarmes são gerados aleatoriamente durante a análise, e a decisão de quando reconhece-los é uma prerrogativa do operador.

Em seguida é verificado o estado das partes do sistema (a esteira e o motor) na janela JanMaintenance. Neste ponto um novo alarme de concentração é gerado pelo sistema. Em seguida, o operador fecha JanMaintenance e alterna a visão entre as janelas JanReactor e JanConveyor e muda o processo de enchimento dos barris do modo automático para manual. Neste instante o sistema gera um alarme de concentração. Como o processo de enchimento está no modo manual, o operador deve retornar para dar continuidade ao enchimento dos barris. Durante o enchimento, o operador retorna à JanReactor, abre a janela de alarmes JanAlarme e reconhece o alarme de concentração. O operador fecha JanAlarme, que estava sobreposta à JanReactor, e abre JanHist para checar o gráfico de tendência.

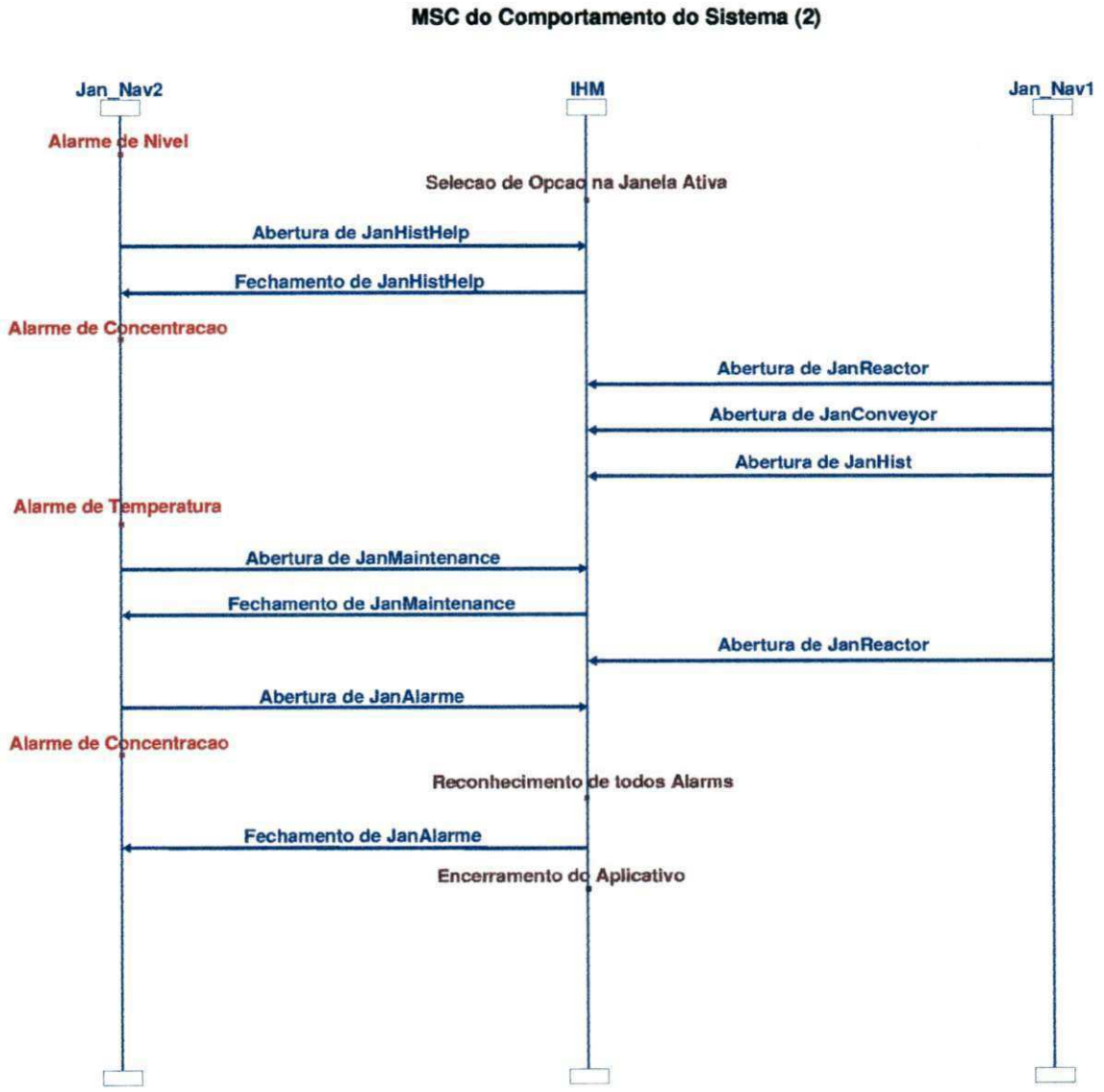


Figura 17 - MSC do Comportamento do Sistema (MSC2)

6.4 Considerações Finais

O modelo apresentado neste capítulo comprova que o modelo proposto em [40,41] possui um caráter genérico, pois a partir de modificações foi possível modelar a interface industrial apresentada neste capítulo.

De posse das análises realizadas no modelo pode-se observar que é possível estender o modelo original para representar novos tipos de navegação (como a sobreposição de janelas) e funcionalidades (como a ativação/desativação de botões de acordo com o contexto da interação).

Porém, devido à extensão das fichas com que o modelo trabalha, a análise do comportamento do sistema se torna complexa para pessoas que não estejam familiarizadas com o modelo (dificuldade que pode ser superada com o uso de MSCs). Outra limitação se deve à necessidade da construção de várias funções para tratar com as fichas do modelo, de modo a representar os diferentes tipos de interação [43].

No próximo capítulo será mostrada uma variação deste modelo, construída em CPN hierárquica temporizada. Nele, a navegação será separada do tratamento dos alarmes (apenas para facilitar a análise visual do modelo), e serão tratadas restrições temporais como forma de verificar a evolução do sistema de forma mais realista.

Capítulo 7 – Modelo CPN Temporizado da Interface Industrial

7.1 Introdução

Neste capítulo é apresentada uma versão do modelo apresentado no capítulo anterior, construído utilizando CPN hierárquica temporizada. A representação da navegação, e a representação da geração e tratamento de alarmes foram separados (para melhorar a análise visual do modelo).

Foram acrescidas ao modelo restrições temporais, como forma de verificar sua evolução em condições próximas às reais. Nesta nova versão do modelo é possível representar o tempo gasto na interação e o tempo de espera no tratamento dos alarmes. Com isso a geração e o tratamento dos alarmes foram modificados para contemplar o efeito das restrições temporais.

7.2 Modelo CPN Hierárquico da Interface Industrial

No modelo apresentado no capítulo anterior foram representados os caminhos de navegação possíveis entre as janelas da interface e os estados (ativo/desativo) dos objetos da interface usados para propósito de navegação, como resultado do caminho de navegação tomado pelo usuário. Com isso foi possível representar e analisar o estado de todos os objetos de interface, estando eles relacionados ou não com a janela ativa.

Neste novo modelo é acrescida a noção de tempo para a realização de tarefas de navegação, bem como para o tratamento dos alarmes gerados. O tratamento de alarmes foi modificado, devido às novas possibilidades de navegação que a inserção do tempo proporcionou, tais como a navegação forçada para janela de alarme devido a ocorrência de *timeout* de um alarme ocorrido anteriormente.

Na Figura 18 é apresentado o nível superior da hierarquia do modelo da interface o qual leva em conta os aspectos temporais da interação. E, em seguida temos o nó de declaração deste modelo.

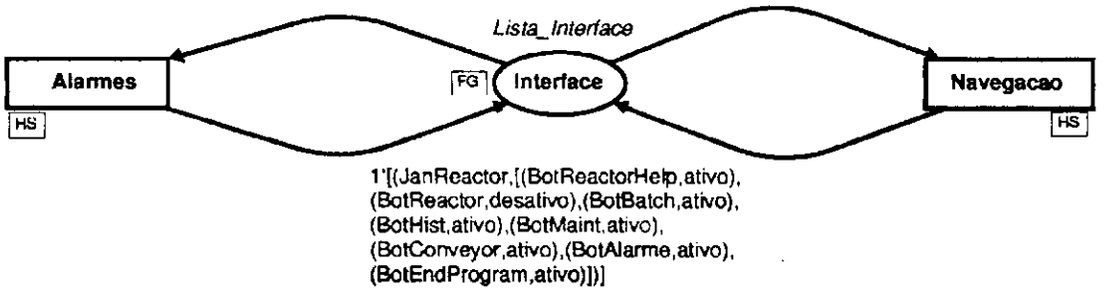


Figura 18 - Modelo CPN da Interface de Controle Industrial (Superpágina)

```
-------------------------------------------------------------------------------
color Janela = with JanReactor | JanBatch | JanHist | JanMaintenance |
               JanConveyor | JanAlarme | JanBatchSet | JanReactorHelp |
               JanHistHelp;
color Botao = with BotReactorHelp | BotBatchExit | BotBatchSet | BotBatchOK |
               BotBatchUp | BotBatchDown | BotBatchBarra | BotMaintExit |
               BotAlarmeConc | BotAlarmeTemp | BotAlarmeNivel | BotReactor |
               BotBatch | BotHist | BotMaint | BotConveyor | BotEndProgram |
               BotBatchSetOK | BotReactorHelpExit | BotConveyorMA |
               BotConveyorEst1 | BotConveyorEst2 | BotConveyorEjet |
               BotConveyorBarra | BotHistCTime | BotHistDefault | ButHistV100 |
               BotHistV50 | BotHist3m | BotHist1m | BotHistBarraVert |
               BotHistBarraHor | BotHistHelp | BotHistHelpExit | BotAlarmeClose;
color Status = with ativo | desativo;
color Botao_Status = product Botao*Status;
color Opcoes = product Janela*Botao_Status;
color Lista_Botao_Status = list Botao_Status;
color Interface = product Janela*Lista_Botao_Status;
color Lista_Interface = list Interface timed;
color Prob_Alarme = int with 1..5 timed;
color Tipo_Al = with Conc | Temp | Nivel;
color Tipo_Al_Bot = product Tipo_Al*Botao;
color Tipo_Al_Bot_Temp = Tipo_Al_Bot timed;

var y: Prob_Alarme;
var Jan, Jan1: Janela;
var Bot, Bot1: Lista_Botao_Status;
var act, act1, act2, Op, Op1: Botao;
var status, status1: Status;
var lista: Lista_Interface;
var ProbAl, ProbAl1: Prob_Alarme;
var interface: Interface;
var Bot_status: Botao_Status;
var al: Tipo_Al;
var al_Bot: Tipo_Al_Bot;
var al_bot_temp: Tipo_Al_Bot_Temp;

(* Funcoes para Verificar se o Elemento esta Disponivel *)
fun existe(act,[]) = false |
     existe(act,(Op,status)::Bot) =
             if Op=act then true
             else existe(act,Bot);

fun existe1((act,status1),[]) = false |
     existe1((act,status1),(Op,status)::Bot) =
             if status=status1 andalso Op=act then true
```

```
            else existe1((act,status1),Bot);

(* Funcoes para Mudar o Estado dos Botoes *)
fun muda_status(act,[]) = [] |
     muda_status(act,(Op,status)::Bot) =
           if act=Op
              then [(Op,desativo)]^^(Bot)
              else [(Op,status)]^^muda_status(act,Bot);

fun muda_status1(v1,v2,v3,[]) = [] |
     muda_status1(v1,v2,v3,(Op,status)::Bot) =
           if (Op=v1 orelse Op=v2 orelse Op=v3)
              then if status=ativo
                      then [(Op,desativo)]^^muda_status1(v1,v2,v3,Bot)
                   else [(Op,ativo)]^^muda_status1(v1,v2,v3,Bot)
           else [(Op,status)]^^muda_status1(v1,v2,v3,Bot);

fun retorno(act,Bot) =
    let
       val v1=BotConveyorEst1;
       val v2=BotConveyorEst2;
       val v3=BotConveyorEjet
    in
       if act=BotConveyorMA
          then change_status1(v1,v2,v3,Bot)
       else Bot
   end;

(* Funcoes para Gerar o Alarme Aleatorio *)
fun alarme(al,act,[]) = [] |
     alarm(al,act,(Op,status)::Bot) =
          let
              val Op1=BotAlarmeConc;
              val Op2=BotAlarmeTemp;
              val Op3=BotAlarmeNivel
          in
          if (Op=Op1 andalso al=Conc) orelse
             (Op=Op2 andalso al=Temp) orelse
             (Op=Op3 andalso al=Nivel)
         then [(Op,ativo)]^^alarme(al,act,Bot)
         else [(Op,status)]^^alarme(al,act,Bot)
         end;

fun existe2(Jan,Bot) =
       if Jan=JanAlarme then true
       else false;

fun Al_Temp(al) =
      if al=Temp then 5
      else if al=Conc then 7
      else 3;
-------------------------------------------------------------------------------
```

O significado de cada uma das cores definidas no nó de declaração é apresentado a seguir.

- *Janela*: Conjunto de todas as janelas da interface.
- *Botao*: Conjunto com todos os botões da interface.
- *Status*: Associado aos botões, pode ser ativo ou desativo.
- *Botao_Status*: Par que representa o nome de um botão e seu estado.

- *Opcoes*: Tripla (janela, botão, status) que associa um nome de janela a um nome de botão e seu estado.
- *Lista_Botao_Status*: Lista de elementos do tipo *Botao_Status*.
- *Interface*: Produto, com um nome de janela como primeiro elemento, e uma lista do tipo *Lista_Botão_Status* como segundo elemento.
- *lista_Interface*: Lista temporizada de elementos do tipo Interface.
- Prob_Alarme: (Probabilidade do alarme) Valor inteiro entre 1 e 5.
- Tipo_Al: Identificação dos alarmes possíveis.
- Tipo_Al_Bot: Dupla que associa a um alarme, o seu botão de reconhecimento.
- Tipo_Al_Bot_Temp: Elementos temporizados do tipo Tipo_Al_But.

Neste modelo o tratamento dos alarmes foi separado da navegação. O modelo de navegação é o mesmo do capítulo anterior acrescido apenas de um tempo hipotético padrão para representar o tempo de execução das tarefas de navegação. Por outro lado, o tratamento dos alarmes foi bastante modificado para possibilitar a verificação do comportamento do sistema em relação às considerações temporais. A seguir descreveremos cada um dos sub-modelos.

7.2.1 Modelo de Navegação

O modelo de navegação da interface é mostrado na Figura 19. Este modelo trata a navegação entre janelas da interface e a seleção de opções dentro da janela ativa. A lista no lugar *Interface* contém a informação sobre qual é a janela ativa. Para tecer considerações temporais sobre a execução das tarefas foi definido que qualquer tipo de navegação irá consumir 2 unidades de tempo, para simular o tempo gasto na execução de uma tarefa.

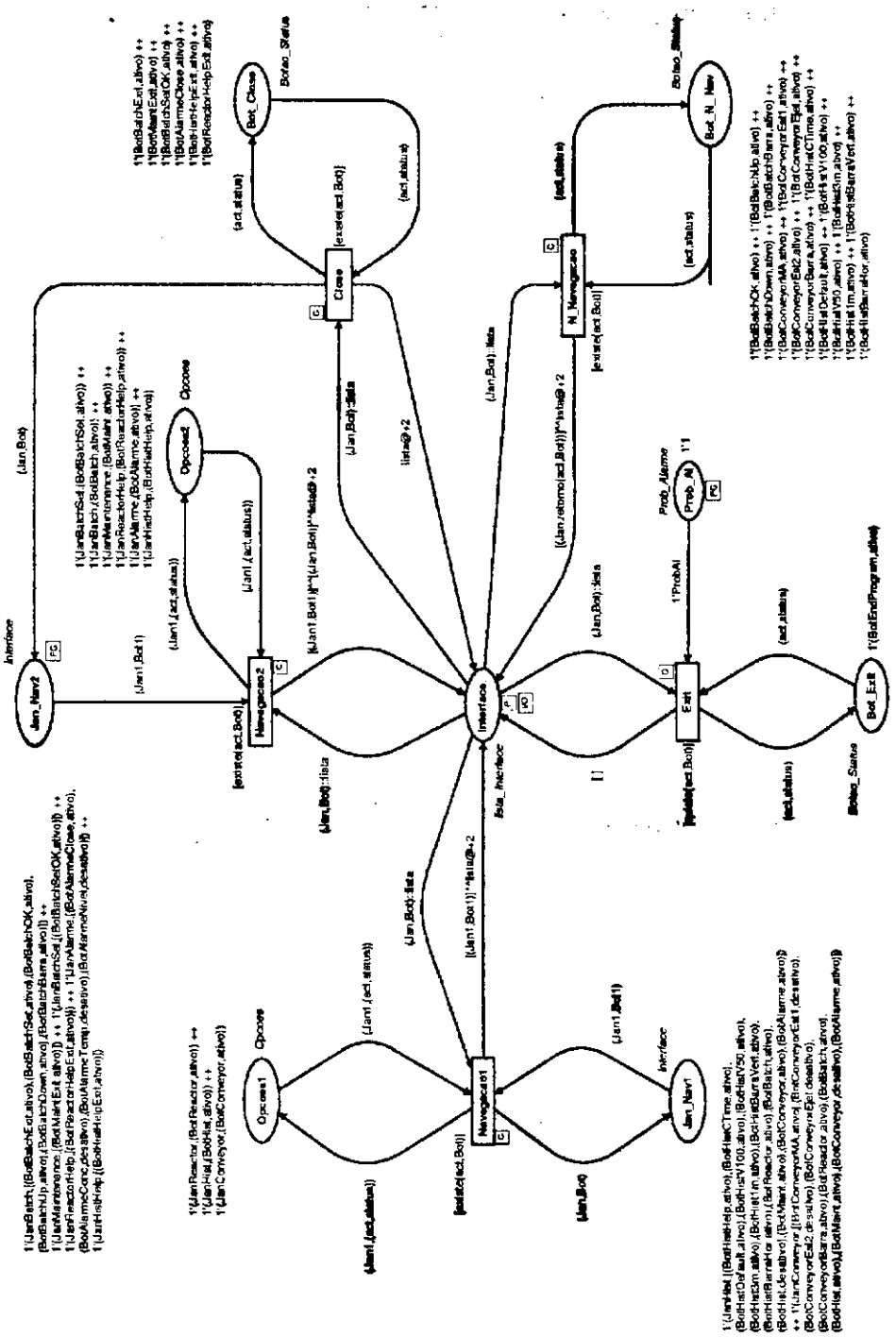


Figura 19 - Modelo CPN de Navegação (Subpágina)

Neste modelo foram mantidas as representações das 5 possibilidades de navegação descritas no capítulo anterior:

- Navegação entre janelas, ocasionando o fechamento da janela ativa e a abertura da janela solicitada. É modelada pela transição *Navegacao1* e pelos lugares *Jan_Nav1* e *Opcoes1*. *Navegacao1* tem como guarda a função *existe*, que verifica se a ação de navegação pretendida está disponível na janela ativa.

- Navegação entre janelas, com a janela requisitada sobrepondo a janela ativa, e se tornando a nova janela ativa. A transição *Navegacao2* e os lugares *Jan_Nav2* e *Opcoes2* modelam essa navegação. *Navegacao2* tem como guarda a função *existe*.
- Navegação entre opções dentro da janela ativa. A transição *N_Navegacao* e o lugar *Bot_N_NAV* modelam esta situação. A guarda de *N_Navegacao* é a função *existe1*, que verifica se a ação selecionada está disponível na janela ativa, e se seu estado é ativo (i.e. disponível para o usuário).
- Fechamento da janela ativa, a qual esta sobreposta à outra janela. Esta navegação é modelada pela transição *Close* e pelo lugar *Bot_Close*. *Close* tem como guarda a função *existe*.
- Encerramento da aplicação. É modelada pela transição *Exit* e pelos lugares *Bot_Exit* e *Prob_Al*. Quando *Close* dispara, o lugar *Interface* fica com uma lista vazia (impossibilitando novas ações de navegação) e a ficha de *Prob_Al* é retirada impossibilitando qualquer nova navegação ou geração de alarme.

7.2.2 Modelo de Geração e Tratamento dos Alarmes

Na Figura 20 é apresentado o modelo de geração e tratamento de alarmes. Para simular a geração dos alarmes aleatórios foi utilizado o artifício de gerar uma probabilidade de alarme, que é modelada pelo lugar *Prob_Al* e pela transição *Prob_Al*. Na ocorrência da transição *Prob_Al* é gerado um número aleatório entre 1 e 5, o qual é usado para limitar a possibilidade de ocorrência dos alarmes, uma vez que as transições *alarm1* e *alarm2* que modelam a ocorrência dos alarmes tem como restrição o fato do número aleatório gerado ser maior que 3.

A transição *alarm1* modela a ocorrência de um alarme quando a janela *JanAlarme* não está ativa. A esta transição estão associado os lugares *Prob_Al*, *alarmes*, *JanNav2* e *Al_Gen*. Para essa transição disparar, o valor da ficha em *Prob_Al* deve ser um número inteiro maior que 3. Ao ocorrer seu disparo é retirada uma ficha do lugar *alarmes*, que indica qual alarme ocorreu e qual é o botão que reconhece esse alarme; é também retirada a ficha do lugar *Jan_Nav2* que corresponde à descrição de *JanAlarme*. A lista de opções de *JanAlarme* é comparada com a ficha retirada do lugar *alarmes*, e através da função *alarme* o botão de reconhecimento da lista de opções de *JanAlarme* que corresponde ao alarme ocorrido tem seu estado mudado para ativo. É então depositada no lugar *Jan_Nav2*, a ficha correspondente a *JanAlarme*, e depositada em *Al_Gen* uma ficha contendo o tipo de alarme

gerado e seu respectivo botão de reconhecimento, acrescido de uma marca de tempo que indica o *timeout* desse alarme (onde os *timeouts* dependem do tipo de alarme gerado).

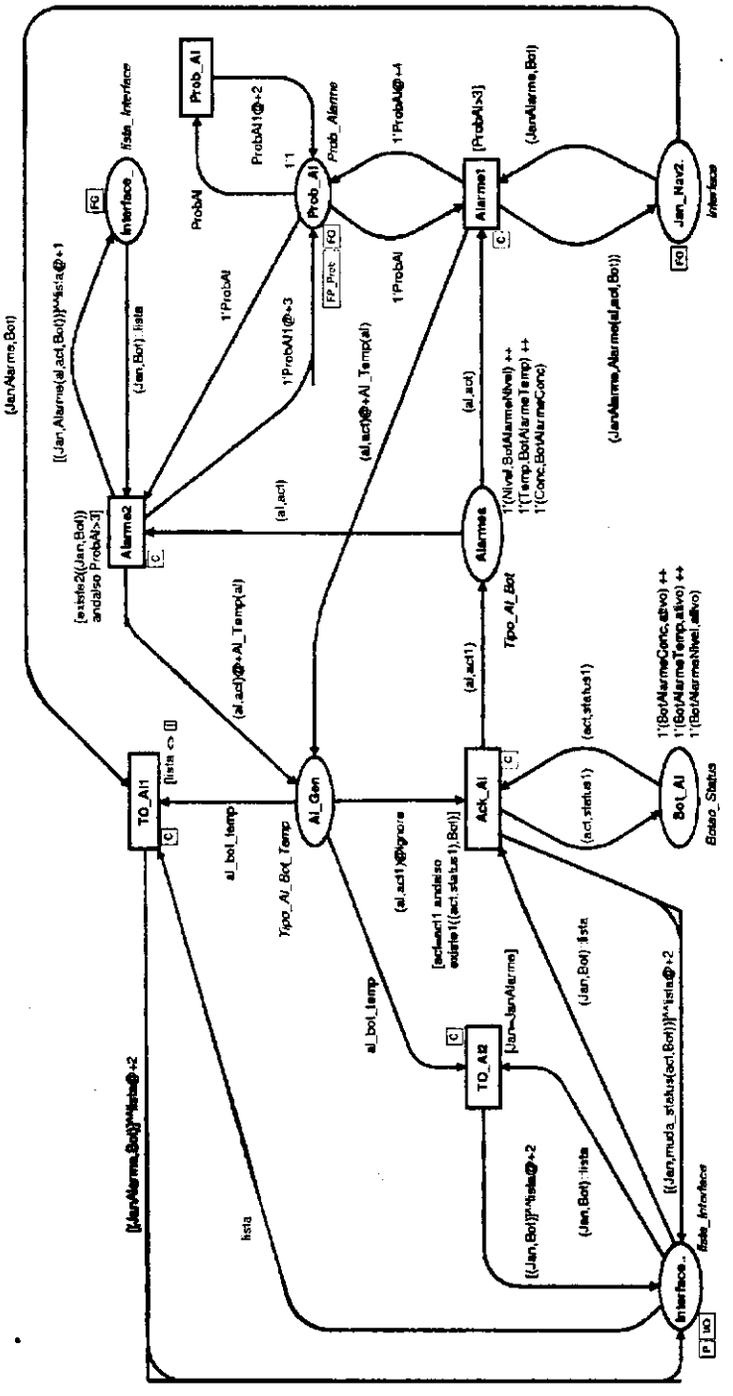


Figura 20 - Modelo CPN da Geração e Tratamento dos Alarmes (Subpágina)

A ocorrência da transição *alarme2* ocorre de forma similar a de *alarme1*. A diferença está no fato da descrição de *JanAlarme* se encontrar na lista do lugar *Interface*, ao invés de *Jan_Nav2*. Com isso é necessário retirar o primeiro elemento da lista (que é a

descrição de *JanAlarme*), executar as mesmas operações descritas para o caso do disparo de alarme1 e devolver a ficha modificada para a cabeça da lista no lugar *Interface*.

Com o alarme gerado (existência de fichas no lugar *Al_Gen*) existem 3 possibilidades de tratamento do alarme:

- Pode-se navegar até a *JanAlarme* para reconhecer o alarme antes da ocorrência de seu *timeout* (como pode ser observada no modelo de navegação).
- Pode ocorrer o *timeout* do alarme, o que força *JanAlarme* a se tornar a janela ativa, independentemente de ser possível navegar diretamente para esta janela (modelado pela transição *TO_Al1*). A ocorrência de *TO_Al1* pega a ficha que descreve *JanAlarme* no lugar *Jan_Nav2* e a insere na cabeça da lista no lugar *Interface*.
- Pode ocorrer o *timeout* do alarme quando *JanAlarme* está ativa. Neste caso é apenas sinalizado o *timeout* (modelado pela transição *TO_Al2*) e transcorre o tempo necessário para ocorrer esta ação.

O reconhecimento dos alarmes é modelado pela transição *Ack_Al* e pelo lugar *Bot_Al*. *Bot_Al* contém fichas que representam os botões de reconhecimento de alarmes. Essas fichas possuem o estado ativo. Inicialmente, em *JanAlarme*, esses itens estão desativos e só se tornam ativos na ocorrência de um alarme. Com isso o reconhecimento de um alarme só pode ser executado se já tiver ocorrido o respectivo alarme.

Ack_Al possui como guarda a função *exist1* que testa se o elemento de interação que se pretende selecionar está na lista de opções da janela ativa e, se o estado desse elemento está ativo, para que esteja disponível para seleção.

Uma outra restrição a ocorrência dos alarmes (além da existência da probabilidade) é que um alarme, uma vez gerado, só pode ser novamente gerado após o seu reconhecimento. Essa restrição pode ser observada no modelo, uma vez que a ficha que indica o tipo de alarme e o botão de reconhecimento, só retorna ao lugar *alarmes* após o disparo da transição *Ack_Al*.

7.4 Análises Realizadas no Modelo

No capítulo anterior a análise foi voltada para transição dos estados de interação através da verificação do modelo, sem no entanto levar em conta aspectos temporais nem o tratamento mais rigoroso dos alarmes. Neste capítulo a análise se volta para a evolução do sistema observando o tempo de realização das tarefas em conjunto com o tratamento de

alarmes. Para a análise foi utilizada a ferramenta *Design/CPN*, através de simulações automáticas e interativas, da geração do grafo de Ocorrência e da geração de um Grafo de Seqüência de Mensagens.

7.4.1 Simulações Automáticas e Interativas

Simulações interativas foram realizadas durante a construção do modelo para verificar a evolução temporal do modelo e sua corretude do ponto de vista de navegação e tratamento dos alarmes. As simulações automáticas foram realizadas no modelo completo para verificar a existência de impasses (outros além daquele relativo à situação de saída do sistema).

7.4.2 Geração do Grafo de Ocorrência (Occ)

O Occ gerado neste parte do trabalho foi parcial, uma vez que a rede é temporizada e cíclica. O que se desejava analisar através do Occ eram as marcações mortas da rede. Foi observado que a rede possui várias marcações mortas, mas que essas marcações mortas indicam o estado do sistema em relação o ocorrência dos alarmes e o andamento de seu tratamento no momento do encerramento do aplicativo.

7.4.3 MSC da Interface Industrial

O uso do MSC com este modelo deu uma visão geral acerca da evolução do sistema levando em conta as restrições temporais. Uma atividade de navegação sempre consome duas unidades de tempo (hipoteticamente), com isso pode ocorrer o caso de um alarme ser gerado pelo sistema durante uma ação de navegação. O tempo de início de cada ação é mostrado nos eixos à esquerda e à direita do MSC.

Outro fato a ser observado é com relação ao tratamento dos *timeouts*. Da forma como o sistema se apresenta não é possível abortar uma operação de navegação na ocorrência de um *timeout*. A operação de navegação deve ser totalmente completada para então o *timeout* ser tratado. Em sistemas com prazos críticos para os *timeouts* essa característica pode se tornar um problema, que deve ser estudado mais detalhadamente, uma vez que este trabalho não determina uma solução para este problema.

A geração de um alarme não consome tempo do sistema por não ser uma ação diretamente relacionada com a interação operador/interface.

A seguir é mostrado um MSC da evolução do sistema levando em conta os aspectos temporais e a geração e tratamento dos alarmes. Neste MSC podem ser observados os três tratamentos possíveis para os alarmes.

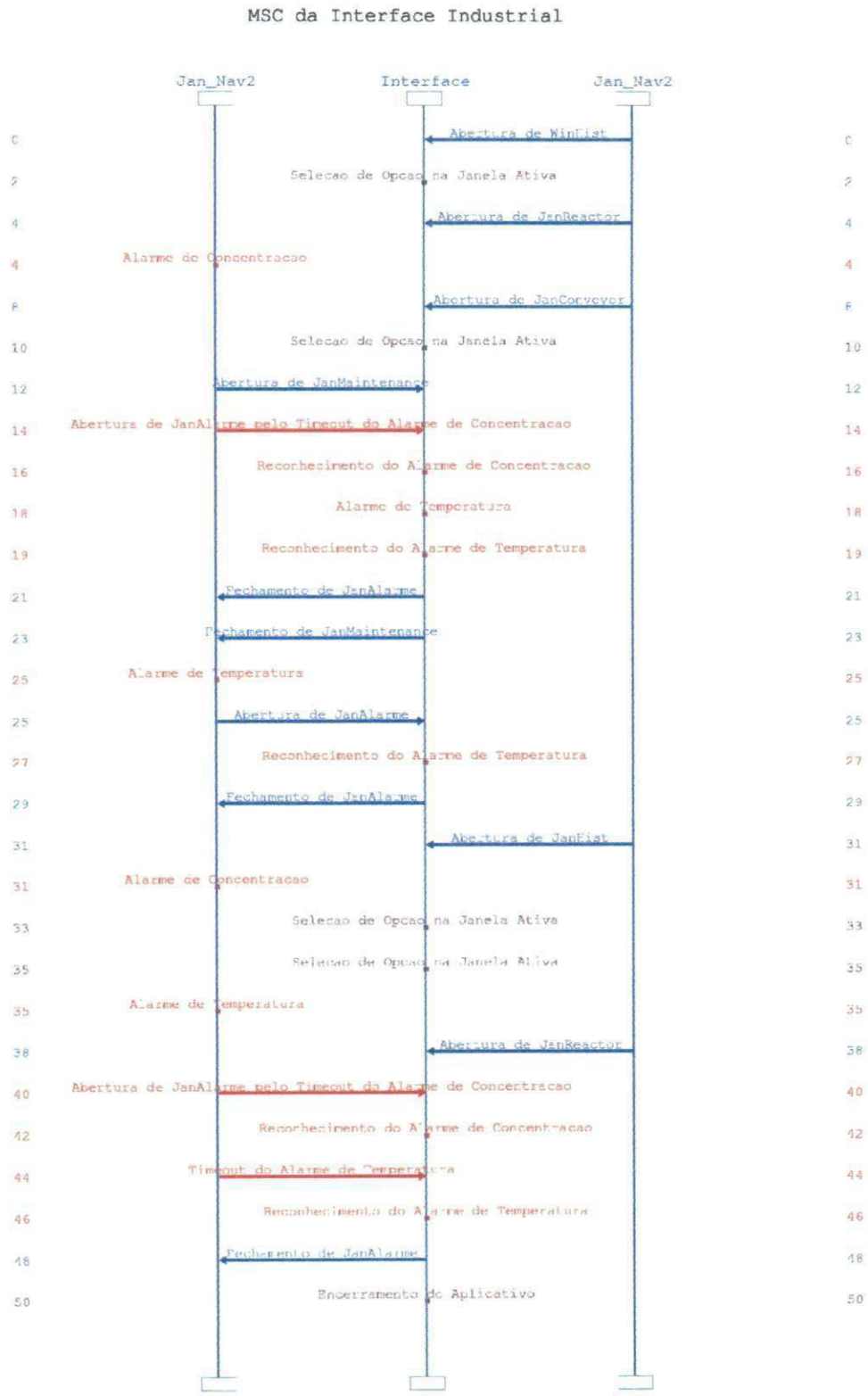


Figura 21 - MSC da Interface Industrial Levando em Conta as Restrições Temporais

No MSC, inicialmente o operador entra no aplicativo na janela *JanReactor*. Ele então navega para a janela *JanHist* e lá checa o gráfico de tendências e seleciona uma opção disponível (a opção de zoom). Então, ele retorna a *JanReactor* no momento em que ocorre um alarme de concentração, o qual é ignorado. Em seguida há a navegação para *JanConveyor* onde é ajustada a velocidade da esteira que transporta os barris para o enchimento. Então é aberta *JanMaintenance*, e logo em seguida *JanAlarme* é aberta devido ao *timeout* do alarme de concentração. Observe que entre *JanMaintenance* e *JanAlarme* não há um caminho de navegação direta, mas devido ao tratamento de alarmes, *JanAlarme* é sempre aberta na ocorrência de um *timeout*, independente da existência ou não de uma caminho de navegação.

O alarme é reconhecido, ocorre um alarme de temperatura enquanto o operador ainda está em *JanAlarme*. Este também é reconhecido. É fechada *JanAlarme* e *JanMaintenance*, e ocorre um novo alarme de temperatura. O operador então navega para *JanAlarme*, reconhece este alarme antes de ocorrer o seu *timeout* e fecha *JanAlarme*, retornando para *JanConveyor*.

Em seguida é aberta *JanHist* e gerado um alarme de concentração, ignorado pelo operador que muda a escala de visualização do gráfico duas vezes. É gerado um alarme de temperatura, que também é ignorado. É aberta *JanReactor* mas logo em seguida *JanAlarme* é aberta devido ao *timeout* do alarme de concentração. O operador então reconhece esse alarme. Ocorre então o *timeout* do alarme de temperatura, mas como *JanAlarme* já é a janela ativa, esse *timeout* é apenas indicado nesta janela. O operador então reconhece esse alarme, fecha *JanAlarme* e encerra a execução do aplicativo.

7.5 Considerações Finais

A partir da modelagem e análise deste sistema verificou-se que o modelo pode ser utilizado para o contexto de uma interface industrial. Pode ser observado que com o uso de hierarquia a compreensão do modelo ficou bem mais fácil.

Porém, permanece o problema das fichas extensas. E, verificamos a necessidade de estudar o caso do *timeout* de um alarme que ocorre durante uma ação de navegação.

Capítulo 8 – Conclusões

Este trabalho apresentou e discutiu a aplicação de um modelo de navegação com estrutura genérica construído no formalismo das redes de Petri coloridas para apoiar a análise do componente de navegação no projeto de interface homem-máquina. Foram apresentados os resultados do uso do modelo de navegação em um estudo de caso relativo a uma aplicação industrial.

O estudo de caso se dividiu em duas etapas. Na primeira etapa o modelo de navegação foi instanciado para o caso industrial sem levar em conta os aspectos temporais da interação. Nesta etapa foram abordados os aspectos de navegação e funcionalidade encontrados nestes tipos de interfaces.

Na segunda etapa foi acrescido ao modelo o conceito de tempo na evolução do sistema, bem como foi reformulada a geração e o tratamento dos alarmes que ocorrem no sistema de forma a verificar o seu comportamento face as restrições temporais. Foi também verificado o comportamento do sistema com relação ao tempo gasto na sua evolução (atribuindo-lhes um valor hipotético). Finalmente foi verificado o comportamento do sistema em relação a ocorrência de eventos que possuem restrições temporais, especificamente o *timeout* dos alarmes

Conclui-se assim que os modelos desenvolvidos neste trabalho possuem uma estrutura genérica, podendo ser utilizados para representar diferentes contextos de interfaces, contanto que o conjunto de cores seja ajustado para representar o novo conjunto de objetos (botões, janelas, menus) e ações.

8.1 Discussão da abrangência dos resultados

Nesta seção apresentaremos o alcance dos resultados e suas limitações com base nos resultados obtidos a partir da análise dos modelos, bem como as conclusões gerais obtidas com a realização deste trabalho.

8.1.1 Alcance dos resultados:

O modelo de navegação se mostrou adequado a modelagem de interface de programas supervisórios industriais baseada em janelas e eventos.

Com o modelo podemos representar o comportamento do sistema levando em conta o tempo de execução das tarefas e as restrições temporais a ações críticas do sistema sob estudo.

O modelo também é adequado para representar a funcionalidade das interfaces a partir do estado dos objetos de interação, representando assim o contexto da interação.

Aspectos importantes da usabilidade de interfaces podem ser verificados através da análise formal do modelo. Dentre estes aspectos se encontram a reiniciabilidade, que é verificada através da presença de *home markings* (*marcações recorrentes*) nos estados possíveis do modelo; e o acesso a pontos específicos e reversibilidades na interação, que são verificados através da análise das marcações da rede.

8.1.2 Limitações:

Uma limitação do modelo proposto é que embora seja adequado para representar interfaces industriais em geral, para sistemas de maior porte (a medida que aumenta o número de janelas e objetos de interação) sua topologia dificulta uma análise visual dos componentes. Mas essa dificuldade de análise é superada com o uso de MSC na análise da evolução do sistema.

Uma outra limitação em relação ao alcance dos resultados obtidos é que apenas com a avaliação formal da interface não é possível garantir a sua usabilidade. Faz-se necessária a construção de um protótipo da interface modelada de modo a realizar testes de usabilidade com um grupo de usuários. Desta forma é possível validar a melhora no componente de navegação da interface.

8.1.3 Conclusões Gerais:

Com a realização deste trabalho conclui-se que o modelo de navegação é adequado para o contexto de sistemas industriais, e que suas extensões tornaram viável a modelagem do comportamento da classe de interface em estudo ao introduzir as restrições temporais.

8.2 Propostas de trabalhos futuros

Considerando as limitações citadas acima são propostas de continuidade para este trabalho:

- Aplicação dos modelos propostos a um contexto real de interfaces homem-máquina de sistemas industriais automatizados para se avaliar o efeito de seu uso sobre a usabilidade destes sistemas.
- Realização de testes de usabilidade com o protótipo construído a partir de uma interface modelada junto a usuários do sistema, com o propósito de validar o processo de otimização do componente de navegação da interface do sistema sob estudo.
- Introdução, no modelo, de restrições temporais baseadas em modelos estocásticos do comportamento do usuário, principalmente em situações críticas. O objetivo é verificar o comportamento do sistema com base um uma visão mais realista dos tempo gasto na realização de tarefas.
- Exploração de novas funcionalidades das IHM industriais de forma a ampliar a abrangência do modelo.
- Verificação do comportamento do sistema em situações críticas não contempladas nas análises realizadas neste trabalho, tais como o a ocorrência de um alarme durante a realização de ações.

Referências Bibliográficas

[1] Ambler, S. W., "User Interface Design: Tips and Techniques", AmbySoft Inc. White Paper, Cambridge, 1998, http://www.ambysoft.com/userInterfaceDesign.pdf

[2] Andersson, M. and Bergstrand, J., "Formalizing Use Cases with Message Sequence Charts", Department of Communication Systems at Lund Institute of Technology, May 1995.

[3] Barfield, L. "The User Interface – Concepts and Design". Ed. Addison-Wesley, England, 1993.

[4] Bastide, R.; Palanque, P. "Conformance and Compatibility Between Models as Conceptual Tools for a Consistent Design of Interactive Systems". CHI99 Workshop on Tool Support for Task-Based User Interface Design, Pittsburg, USA, May, 1999.

[5] Bastide, R.; Palanque, P. "A Petri Net Based Environment for the Design of Event-Driven Interfaces". Proceedings of the 16th International Conference on Application and Theory of Petri Nets, p. 66-83, Turin, June, 1995.

[6] Braga, J. D. M.; Figueiredo, J. C. A. "Curso de Redes de Petri". I Seminário Regional dos Estudantes de Engenharia do Nordestes, UFPB, Campina Grande, PB, abril, 1997.

[7] Costa, L. S. S. et alli, "Manufatura Integrada por Computador – Sistemas Integrados de Produção: Estratégia, Organização, Tecnologia e Recursos Humanos", Ed. Campus, Rio de Janeiro, RJ, 1995.

[8] Design/CPN Online. url: www.daimi.au.dk/designCPN/, 2000.

[9] Desroches, A. A. "Modelling and Control of Automatic Manufacturing Systems". Ed. IEEE Computer Society Press, Washington, USA, 1990.

[10] Dicesare, F.; Harhalakis, G.; Proth, J. M.; Silva, M.; Vernadat, F. B. "Practice of Petri Nets in Manufacturing". Ed. Chapman & Hall, UK, 1993.

[11] Farias, G. F.; Turnell, M. F. V. Q., "Modelagem de Aspectos Funcionais de Interfaces Homem-Máquina de Sistemas Industriais Em Redes de Petri Coloridas Hierárquicas", Anais do XIII Congresso Brasileiro de Automática, 2000.

[12] Farias, G. F. "Modelo de Funcionalidades de Interface Homem-Máquina Industriais". Relatório de Projeto de Pesquisa, Pós-Graduação em Engenharia Elétrica da UFPB, Campina Grande, PB, 1999.

[13] Farias, G. F.; Turnell, M. F. V. Q. "Modelagem em Redes de Petri da Interface Homem-Máquina de Sistemas Industriais". Anais do XII Congresso Brasileiro de Automática, Uberlândia, MG, 1998.

[14] Farias, G. F.; Turnell, M. F. V. Q., "Uso de Programas Supervisórios na Automação de Processos Industriais – Uma Visão de Interfaces do Usuário", Anais do XI Congresso Brasileiro de Automática, São Paulo, SP, 1996.

[15] Farias, G. F., "Diretrizes para Projeto de Interfaces Homem-Máquina Aplicadas a Sistemas de Supervisão de Processos Industriais", Dissertação de Mestrado, Curso de Pós-Graduação em Engenharia Elétrica, UFPB, 1994.

[16] Farias, G. F.; Turnell, M. F. V. Q., "The Use of Supervisors Software in the Industrial Automation Process Control form the User Interface Perspective", Proceedings of the 1996 IEEE International Conference on Systems Man and Cybernetics, Vol. 1, p. 756-761, Beeijing, China, 1996.

[17] Fields, B.; Wright, P.; Harrison, M. "Time, Tasks and Errors". SIGCHI Bulletin, Vol. 28, n° 2, p. 53-56, April, 1996.

[18] Harvey, J. "Evaluation Cookbook". Institute for Computer Based Learning, Heriot-Watt University, Edinburgh, 1998. http://www.icbl.hm.ac.uk/ltdi.

[19] Jensen, K. "Coloured Petri Nets – Basic Concepts, Analysis Methods and Pratical Use – Vol. 1". Ed. Spring-Verlag, USA, 1992.

[20] Jensen, K. "Coloured Petri Nets – Basic Concepts, Analysis Methods and Pratical Use – Vol. 2". Ed. Spring-Verlag, Germany, 1995.

[21] Johnson, C.; Gray, P. "Temporal Aspects of Usability". SIGCHI Bulletin, Vol. 28, n° 2, p. 32, April, 1996.

[22] Li, X.; Mugridge, W. B.; Hosking, J. G. "A Petri Net-Based Visual Language for Specifying GUIs". Proceedigns of the IEEE Symposium in Visual Languages, p. 50-57, 1997.

[23] Murata, T. "Petri Nets: Properties, Analysis and Applications". Proceedings of the IEEE, n° 77 (4), p. 541-580, April, 1989.

[24] Palanque, P.; Bastide, R. "Formal Specifications as a Tool for Objective Assessment of Safety-Critical Interactive Systems". IFIP TC13 Interact'97 Conference, p. 155-162, Sydney, Australia, July, 1997.

[25] Palanque, P.; Bastide, R. "A Design Life-Cycle for the Formal Design of User Interfaces". (FAHCI'96) BCS-FACS Workshop on the Formal Aspects of the Human-Computer Interface, Shefield, UK, September, 1996.

[26] Palanque, P.; Bastide, R. "Performance Evaluation as a Tools for Evaluating the Formal Design of Interactive System". (CESA'96) IMACS Conference on Computational Engineering in System Application, Special Session on Performance Evaluation, IEEE Press, Lille, July, 1996.

[27] Palanque, P.; Bastide, R. "Spécifications Formelles pour L'ingéniere des Interfaces Homme-Machine". Revue Techniques et Sciences Informatiques, Vol. 14, Nj 4, p. 473-500, 1995.

[28] Palanque, P; Bastide, R. "Theoretical Foundations of Recent Formal Approaches in HCI design". Research Symposium CHI'94. Boston, 23-30 April 1994.

[29] Palanque, P.; Bastide, R,; Sengès, V. "Validating Interactive System Design Through the Verification of Formal Task and System Models". (EHCI'95) 6th IFIP Working Conference on Engineering for Human-Computer Interaction Grand Targhee Resort, Wyoming, USA, August, 1995.

[30] Palanque, P.; Bastide, R.; Sengès, V. "Automatic Code Generation from a High-Level Petri Net Based Specification of Dialogue". (EWHCI'94) East-West Conference on Human Computer Interaction, St. Petesburg, Russia, August, 1994.

[31] Palanque, P.; Bastide, R; Paternó, F. "Formal Specifications as a Tool for Objective Assessment of Safety-Critical Interactive Systems". IFIP TC13 Interact'97 Conference, p. 155-162, Sydney, Australia, July, 1997.

[32] Palanque, P.; Bastide, R.; Sibertin-Blanc, C.; Dourte, L. "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.

[33] Queiroz, J. E. R. "Validação de uma Metodologia de Projeto de Interfaces Usuário-Computador". Dissertação de Mestrado, Pós-Graduação em Engenharia Elétrica da UFPB, Campina Grande, PB, 1994.

[34] Queiroz, J. E. R. "Abordagem Híbrida para a Avaliação da Usabilidade de Interfaces com o Usuário". Tese de Doutorado, Pós-Graduação em Engenharia Elétrica da UFPB, Campina Grande, PB, 2001.

[35] Ribeiro, Z. B. S. "Supervisores Distribuídos para Sistemas Flexíveis de Manufatura utilizando Redes de Petri". Dissertação de Mestrado, Departamento de Engenharia Elétrica, UFPB, Campina Grande, PB, agosto, 1999.

[36] Rosis, F. de; Pizzutilo, S.; De Carolis, B. "Formal Description and Evaluation of User Adapted Interfaces". Int. J. Human-computer Studies n° 49, p. 95- 120, 1998.

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

[38] Schlungbaum, E.; Elwert, T. "Dialogue Graphs - A Formal and Visual Specification Technique for Dialogue Modeling". Proceedings of BCS-Workshop: Formal Aspects of the Human Computer Interface, 1996.

[39] Schlungbaum, E.; Elwert, T. "Tadeus – a Model-based Approach to the Development of Interactive Software Systems", Rostocker Informatik-Berichte, nº 17, p. 93-104, 1995.

[40] Sousa, M. R. F. "Avaliação Iterativa de Especificação de Interfaces com Ênfase na Navegação". Tese de Doutorado, Pós-Graduação em Engenharia Elétrica da UFPB, Campina Grande, PB, 1999.

[41] Sousa, M. R. F.; Turnell, M. F. Q. V. "User Interface Based on Coloured Petri Nets Modelling and Analysis". Proceedings of the 1998 IEEE International Conference on Systems Man and Cybernetics, San Diego, USA, 1998.

[42] Sousa, M . R. F.; Turnell M. F. Q. V.; Figueiredo J. C. A. "Modelagem do Fluxo de Informações em Interfaces usuário-computador". Anais da XXIII Conferência Latino Americana de Informática – CLEI'97, Valparaíso, Chile, Nov. 1997, pp. 813-822.

[43] Turnell, M. F. Q. V.; Scaico, A.; Sousa, M. R. F.; Perkusich, A. "Industrial User Interface Evaluation Based On Coloured Petri Nets Modelling and Analysis". Proceedings of the Eighth Workshop on the Design, Specification and Verification of Interactive Systems, p. 144-157, Glasgow, Scotland, June, 2001.

[44] Turnell, M. F. V. Q., "Conceitos e Projeto de Interfaces Usuário-Computador", Notas de Aula da Disciplina Projeto de Interface Homem-Máquina, Curso de Pós-Graduação em Engenharia Elétrica, UFPB, 2000.

[45] Viswanadhan, N.; Narahari, Y. "Performance Modelling of Automated Manufacturing Systems". Ed. Prentice Hall, New Jersey, USA, 1992.

[46] Zhou, M.; Dicesare, F. "Petri Net Synthesis for Discret Event Control of Manufacturing Systems". Ed. Kluwer Academic Publisher, Massachusettes, USA, 1993.

Bibliografia Consultada

[I] Farias, G. F. "Revisão Bibliográfica sobre o Uso de Redes de Petri na Modelagem de Sistemas Interativos". Relatório de Projeto e Pesquisa, Pós-Gradução em Engenharia Elétrica, Campina Grande, PB, 1999.

[II] Farias, G. F.; Turnell, M. F. Q. V. "Modelling the User Interface of an Industrial System in Coloured Petri Nets". Proceedings of the 1998 IEEE International Conference on Systems Man and Cybernetics, San Diego, USA, 1998.

[III] Hix, D.; Hartson, H. R. "Developing User Interfaces, Ensuring Usability Through Products & Process". Ed. John Willey and Sons Inc., 1993.

[IV] Johnson, C.; Gray, P. "Assesing the Impact of Time on User Interface Design". SIGCHI Bulletin, Vol. 28, n° 2, p. 33-35, April, 1996.

[V] Mezzanotte, M.; Paternó, F. "Including Time in the Notion of Interactor". SIGCHI Bulletin, Vol. 28, n° 2, p. 57-61, April, 1996.

[VI] Nielsen, J. "Usability Testing". In: Graviel Salvendry (Ed), Handbook of Human Factors and Ergonomics, p. 1617-1633, Ed. John Willey & Sons Inc., New York, 1997.

[VII] O'Donnell, P.; Draper, S. W. "How Machine Delays Change User Strategies". SIGCHI Bulletin, Vol. 28, n° 2, p. 39-42, April, 1996.

[VIII] Palanque, P.; Bastide, R. "Time Modelling in Petri Nets for The Design of Interactive Systems". SIGCHI Bulletin, Vol. 28, n° 2, p. 43-46, April, 1996.

[IX] Shneiderman, B. "Designing the User Interface: Strategies for Effective Human-Computer Interaction". Third Edition, ISBN 0-201-69497-2, 1998.

[X] Shneiderman, B. "Universal Usability". Communications of the ACM, 43 (5), p. 85-91, May, 2000.

[XI] Thomas, R. C. "Long-term Variation in User Actions". SIGCHI Bulletin, Vol. 28, n° 2, p. 36-38, April, 1996.

[XII] Treu, S. "User Interface Evaluation – A Structured Approach". Ed. Plenum Press, New York and London, 1994.