UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

UM AMBIENTE PARA A ANÁLISE E SIMULAÇÃO DE SISTEMAS MODELADOS POR REDES DE PETRI

Dissertação submetida à Universidade Federal de Santa Catarina para a obtenção do grau de Mestre em Engenharia Elétrica

CARLOS ALBERTO MAZIERO

Florianópolis, junho de 1990

UM AMBIENTE PARA A ANÁLISE E SIMULAÇÃO

DE SISTEMAS MODELADOS POR REDES DE PETRI

CARLOS ALBERTO MAZIERO

Esta dissertação foi julgada adequada para a obtenção do título de

MESTRE EM ENGENHARIA ELÉTRICA

Especialidade **ENGENHARIA ELÉTRICA**, área de concentração **SISTEMAS DE CONTROLE E AUTOMAÇÃO INDUSTRIAL**, e aprovada em sua forma final pelo programa de Pós-Graduação.

Prof. JEAN MARIE FARINES, Dr. Ing.
Orientador

Prof. JOSÉ CARLOS MOREIRA BERMUDEZ, Ph. D.
Coordenador do Curso de Pós-Graduação em Engenharia Elétrica

BANCA EXAMINADORA

Prof. JEAN MARIE FARINES, Dr. Ing.
Presidente

Prof. CARLOS ALBERTO HEUSER, Dr. Ing.

Prof. JOSÉ EDUARDO R. CURY, Dr. D'Etat

Prof. JONI DA SILVA FRAGA, Dr. Ing.

Resumo

Este trabalho apresenta o Sistema ARP de Análise e Simulação, que integra diversas ferramentas para validação, verificação, simulação e avaliação de desempenho de sistemas modelados em Redes de Petri ordinárias, com Temporização e com Temporização Extendidas.

Inicialmente, o modelo Rede de Petri (RdP) é formalmente descrito, com suas principais extensões, em particular as temporais. A modelagem em RdP é estudada a partir de noções de modularidade e estruturação hierárquica. As principais técnicas de validação, verificação e avaliação de desempenho de sistemas modelados por Redes de Petri são abordadas.

Apresenta-se a seguir a estrutura geral do Sistema ARP, e as ferramentas que o compõe: análise via grafo de acessibilidade, análise estrutural por invariantes, verificação de equivalência de linguagem, simulação e avaliação de desempenho. Para cada ferramenta são detalhados seus objetivos, métodos, algorítmos empregados e resultados fornecidos.

Finalmente, demonstra-se a capacidade do sistema descrito, aplicando suas ferramentas ao estudo de problemas de interesse real em Automação Industrial e Sistemas Distribuídos: são estudadas a proposta FIP ao Padrão Field-Bus, o protocolo MMS da norma MAP e também um caso de otimização de Sistemas Flexíveis de Manufatura.

Abstract

In this work the ARP Analysis and Simulation System is presented. ARP is composed by several tools for validation, verification, simulation and performance evaluation of systems modeled by ordinary, time and extended time Petri Nets.

First the Petri Net model and some extensions are formally described. The modelling task with Petri Nets is studied using modularity and structuring concepts. Validation, verification and performance evaluation techniques for Petri Net models are discussed.

Secondly, the ARP's general structure and its component tools, such as reachability graph analysis, structural analysis by invariants, language equivalence verification, simulation and performance evaluation, are overviewed. For each tool, the objetives, algorithms and results are presented.

Finally, system's capability is demonstrated by apllying the described tools to the study of Distributed Systems and Industrial Automation problems like the Field-Bus FIP proposal, the MAP's MMS protocol and the optimisation of Flexible Manufacturing Systems.

À Lucia, incansável companheira

Agradecimentos

Aos profs. do LCMI, pelo sempre presente incentivo e cooperação. Em particular ao prof. Jean-Marie Farines, pela orientação, interesse e apoio na realização deste trabalho.

Aos demais membros do LCMI, por seu espírito de equipe, companheirismo e sobretudo por sua amizade.

A meus pais Dorali e Neiva, a meus irmãos e a todos os amigos que me auxiliaram durante esta caminhada.

Ao CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico, pelo suporte financeiro que me foi concedido.

Índice Geral

1. Introdução

A necessidade de redução de custos e aumento de confiabilidade impõe como meta para a indústria sua automação, através da integração das diversas operações do processo de produção. Esta integração é garantida pela existência de Sistemas Informáticos Distribuídos que permitem melhorias substanciais em termos do desempenho global, do compartilhamento dos recursos, da disponibilidade e da confiabilidade dos serviços fornecidos.

Além disso, a crescente automação dos processos de produção, baseada no uso de máquinas com comando numérico, controladores lógicos programáveis, robôs e outros equipamentos, torna necessário o uso de uma filosofia de desenvolvimento adequada, uma metodologia que seja capaz de modelar corretamente estes dispositivos e suas interações, sem dependência das tecnologias envolvidas. Essa modelagem costuma ser complexa, pois problemas de sincronização, exclusão mútua, disponibilidade de recursos e outros, associados a mecanismos automáticos, são freqüentes.

Um dos aspectos prioritários no desenvolvimento de software para aplicações distribuídas e de automação industrial é a disponibilidade de ferramentas que permitam provar as propriedades dinâmicas das especificações, realizar protótipos, simulações e avaliar seu desempenho. A complexidade de tais aplicações leva à utilização de Técnicas de Descrição Formal (TDF) para descrever de forma completa, clara e sem ambigüidades suas especificações.

A principal motivação para o uso de TDFs consiste em servir de base na construção de ferramentas que permitam automatizar atividades como a análise do comportamento lógico e do desempenho, a verificação das especificações, a implementação destas e a determinação da conformidade do software resultante em relação aos requisitos iniciais.

O objetivo central do trabalho desenvolvido no LCMI-DEEL-UFSC é a construção de um ambiente de análise (de comportamento e desempenho) e simulação que permita tratar problemas de Automação Industrial e Sistemas Distribuídos.

Escolheu-se como formalismo de base para tal ambiente a Rede de Petri (RdP), pois esta permite descrever com clareza as relações de causalidade encontradas nos problemas a tratar: seqüência, concorrência, conflitos e sincronização.

Além disso, extensões são possíveis neste mecanismo, que permitem introduzir restrições de tempo e representação de dados [Diaz 86].

O modelo RdP serve de base à construção de ferramentas automáticas para várias etapas do desenvolvimento de um sistema: modelagem, validação, verificação de conformidade, avaliação de desempenho, implementação e testes.

O Sistema ARP, descrito nesta dissertação, é um primeiro passo na construção de um ambiente que proporcione ferramentas automáticas para a análise e simulação de sistemas modelados por Redes de Petri ordinárias, com temporização e com temporização extendidas.

Esta dissertação está estruturada da seguinte forma: o cap. 2 introduz o modelo Rede de Petri, sua definição formal e suas principais extensões. O projeto usando Redes de Petri também é estudado, detalhando-se a construção, validação, verificação, avaliação de desempenho e implementação de sistemas modelados por Redes de Petri.

O cap. 3 descreve o sistema ARP e as ferramentas que este integra. Cada ferramenta tem detalhados seus objetivos, os algorítmos empregados e os resultados fornecidos. Por fim, o cap. 4 busca exemplificar a utilização de algumas das ferramentas do sistema ARP, através do estudo de exemplos reais nas áreas de Sistemas Informáticos Distribuídos e Automação Industrial.

2. O Modelo Rede de Petri

Para o desenvolvimento de sistemas complexos torna-se necessário o uso de Técnicas de Descrição Formal (TDF), para a correta representação de sua especificação.

Uma TDF se caracteriza por seu poder de expressão, que permite representar de forma clara, correta e sem ambigüidade as especificações de um sistema, e por seu poder de análise, que facilita a detecção de erros.

A principal motivação para o uso de TDFs consiste em servir de base na construção de ferramentas que permitam automatizar atividades como a análise do comportamento lógico e do desempenho, a verificação das especificações, a implementação destas e a determinação da conformidade do software resultante em relação aos requisitos iniciais [Vissers 88].

Existe hoje um grande número de abordagens diferentes para as TDFs, baseadas em formalismos como Máquina de Estado Finita, Rede de Petri, Gramáticas Formais, Álgebras de Processos (p.ex. CCS), Tipo de Dados Abstratos, Lógica Temporal e formalismos híbridos.

Apresentar-se-á a seguir o formalismo Rede de Petri, utilizado como base do presente trabalho.

2.1. O Formalismo Rede de Petri

Proposta em 1962 por Karl Petri em sua tese de doutorado "Kommunikation mit Automaten" (comunicação com autômatos), a Rede de Petri (RdP) é um formalismo que permite a modelagem de sistemas dinâmicos com alto grau de paralelismo. Seu poder de expressão é suficiente para representar todas as relações de causalidade entre processos, como situações de concorrência, paralelismo, operações seqüenciais, conflitos, não determinismos e sicronizações.

A partir do modelo básico proposto por Petri (normalmente chamado RdP clássica ou ordinária), muitas extensões são propostas, com a finalidade de aumentar seu poder de expressão, principalmente no tocante a temporização e estruturas de dados.

A importância do uso da RdP como formalismo, reside nas facilidades oferecidas para a análise do comportamento do sistema modelado e de seu desempenho.

Em particular, o modelo RdP permite automatizar as várias etapas do processo de desenvolvimento de um sistema: especificação, validação, verificação de conformidade, análise de desempenho e até mesmo a implementação. As ferramentas assim obtidas poderão ser facilmente integradas num mesmo ambiente.

2.1.1. Definição do Modelo de Base

A RdP é um modelo do tipo estado-evento, onde cada evento possui pré-condições a serem observadas para sua ocorrência e pós-condições decorrentes desta, que por sua vez podem ser pré-condições de outros eventos.

A RdP é representada por um grafo com dois tipos de elementos, chamados lugares e transições, interligados por arcos direcionados ponderados. Os lugares representam os estados locais ou condições do sistema, cuja existência é indicada pela presença de uma ficha no respectivo lugar. O estado global do sistema é representado pela distribuição das fichas na rede, chamada "marcação" [Peterson 81].

Os eventos do sistema são representados por transições. A cada evento corresponde uma transição, cujo disparo indica sua ocorrência. Relacionam-se os eventos às condições através dos arcos que interligam as transições aos lugares.

Na fig. 2.1, que busca modelar uma operação de usinagem, tem-se um exemplo de rede de Petri com arcos de peso unitário:

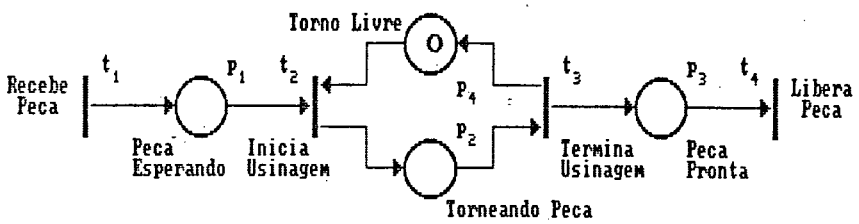

Fig. 2.1: Rede de Petri

Formalmente define-se a RdP clássica ou ordinária como uma quíntupla RdP= $\{P,T,I,O,M_0\}$ onde:

P : Conjunto finito de m lugares $\{p_1, p_2,..., p_m\}$.

T : Conjunto finito de n transições $\{t_1, t_2,..., t_n\}$, $P \cap T = \{\}$.

I:TxP->N : Função de entrada das transições. I_{ij} indica o peso do arco que sai do j-ésimo lugar para a i-ésima transição.

O:TxP->N : Função de saída das transições. O_{ij} indica o peso do arco que sai da i-ésima transição para o j-ésimo lugar.

M_0:P->N : Marcação (n^o de fichas) inicial de cada lugar.

A dinâmica da RdP é representada pela evolução das fichas, causada pelos disparos das transições sensibilizadas, que provocam a transferência das fichas entre os lugares da rede, alterando sua marcação e o estado do sistema modelado.

Uma transição t_i está sensibilizada pela marcação M quando todas as suas pré-condições forem satisfeitas:

$$\forall p_j \in P \; M(p_j) \geq I(t_i,p_j), \text{ ou seja, } M \geq I(t_i)$$

O disparo de t_i a partir da marcação M gera uma nova marcação M':

$$M'(p_j) = M(p_j) + O(t_i,p_j) - I(t_i,p_j) \quad \forall p_j \in P,$$

$$t_i$$

$$\text{ou seja, } M \rightarrow M' \qquad M' = M + O(t_i) - I(t_i)$$

Uma seqüência "s" de transições $\{t_j\ t_{j+1} \ldots t_{j+k}\}$ é uma seqüência de disparos a partir de M_i se existem marcações $M_i, M_{i+1},\ldots,M_{i+k}$ de modo que:

$$M_i \xrightarrow{t_j} M_{i+1} \xrightarrow{t_{j+1}} \ldots \xrightarrow{t_{j+k}} M_{i+k} \rightarrow M_{i+k+1} \text{ ou } M_i \xrightarrow{s} M_{i+k+1}$$

O conjunto $[M_0]$ de marcações acessíveis a partir da marcação inicial M_0 é definido por:

$$[M_0] = \{M_i / \exists\, s / M_0 \xrightarrow{s} M_i\}$$

Esse conjunto pode ou não ser finito, dependendo da topologia da rede e de M_0.

O grafo de marcações acessíveis G associado à rede de Petri com $[M_0]$ finito é formado pelo conjunto de marcações acessíveis $[M_0]$ e pelo conjunto U de arcos que representam os disparos de transição que levam de uma marcação a outra:

$$U = \{(M_j,t_i,M_k) / \exists\, t_i \in T / M_j \xrightarrow{t_i} M_k,\ \forall M_j,M_k \in [M_0]\}$$

2.1.2. Extensões ao Modelo de Base

Através de extensões ao modelo de base busca-se aumentar seu poder de expressão, permitindo que contenha informações não representáveis no modelo ordinário, como

temporizações e manipulação de dados, e também aumentando a concisão da representação, reagrupando as partes do modelo com comportamento semelhante.

a. RdP com Arcos Inibidores

Supre uma deficiência intrínseca ao modelo RdP, o "teste a zero". Possui um tipo especial de arco de entrada de transição que só permite seu disparo quando a marcação do lugar associado for nula [Peterson 81]. Obtém-se um ganho real na capacidade de modelagem, embora se perca em poder de análise.

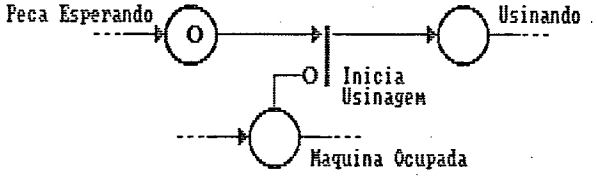

Fig. 2.2: RdP com arcos inibidores

b. RdP com Temporização

A introdução do tempo no formalismo RdP pode ser efetuada de várias formas, sendo mais difundidos os modelos de Sifakis, Ramchandani e Merlin [Menasche 82].

No modelo de Sifakis, a cada lugar é associado um atraso fixo durante o qual as fichas recém chegadas são consideradas "não-disponíveis" (não podem ser consideradas para o disparo de transições), tornando-se disponíveis após decorrido o tempo de espera.

No modelo de Ramchandani associa-se a cada transição uma duração de disparo fixa. Ao disparar, a transição retira as fichas dos lugares de entrada, as retém durante um certo tempo e deposita-as nos lugares de saída.

Um modelo mais geral é proposto por Merlin [Merlin 76], chamado RdP com temporização (RdPT), que associa a cada transição da rede um intervalo da forma seguinte:

$$[SEFT,SLFT] \in Z^+, 0 \leq SEFT \leq SLFT < \infty$$

O valor SEFT (Static Earliest Firing Time) indica o tempo mínimo durante o qual a transição deve estar sensibilizada antes de seu disparo. O valor SLFT (Static Latest Firing Time) indica o tempo máximo no qual a transição deve disparar, se ainda sensibilizada. É importante observar que o disparo da transição tem duração zero [Menasche 82].

Formalmente uma RdPT é um par RdPT= $\{RdP,D_0\}$ onde:

RdP : RdP ordinária $\{P,T,I,O,M_0\}$

D_0 : $T \rightarrow (R^+ \cup 0) \times (R^+ \cup \infty)$

$\forall t_k \; D_0(t_k) = \{[SEFT_k, SLFT_k], 0 \leq SEFT_k \leq SLFT_k\}$

$D_0(t_k)$ é o intervalo de disparo inicial ou estático de t_k. Os intervalos são também chamados domínios de disparo.

Um exemplo simples de RdPT é dado na fig. 2.3:

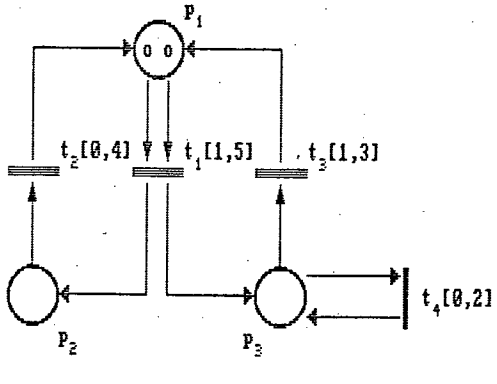

Fig. 2.3: RdP com temporização

O estado S:(M,D) de uma RdPT é caracterizado por sua marcação M e pelo conjunto D dos domínios dinâmicos de disparo $[DEFT_j,DLFT_j]$ de cada transição t_j sensibilizada por M, que podem diferir dos domínios estáticos. A partir de S uma transição t_k é disparável em um instante Θ_k sse:

$$I(t_k) \leq M \quad e \quad DEFT_k \leq \Theta_k \leq Min\,(DLFT_j)\ \forall t_j \in D$$

O novo estado S', com marcação M' e domínio dinâmico de disparo D' define-se como:

$$M' = M + O(t_k) - I(t_k)$$

$\forall t_j$: se $I(t_j) > M'$ então

$D'(t_j)$ é indefinido

senão

se $I(t_j) > M$ então

$D'(t_j) = D_0(t_j)$

senão

$D'(t_j) = [Max\,(0, DEFT_j - \Theta_k), DLFT_j - \Theta_k]$

Deve-se observar que esta extensão não considera a ocorrência de transições multi-sensibilizadas (t_i está multi-sensibilizada se $\exists M_j$ / $\forall p_k\ I(t_i,p_k) \geq 2^*\ M_j(p_k)$). Para uma transição multi-sensibilizada deveria ser considerado um intervalo de disparo distinto para cada nível de sensibilização.

Para estudar o sistema sob a ótica de seu desempenho, é importante considerar também a probabilidade da ocorrência de cada evento da RdP que o modela, definindo uma nova extensão: a RdP com Temporização Extendida, ou RdPTE [Roux 85].

Na RdPTE, atribui-se a cada domínio de disparo de transição da rede uma função de densidade de probabilidade, que pondera o instante de disparo Θ_k da transição t_k dentro do intervalo permitido.

Formaliza-se uma RdPTE como um par $\{RdPT,F_0\}$ onde:

RpPT : Rede de Petri com temporização $\{P,T,I,O,M_0,D_0\}$

F_0 : $T \rightarrow F^+$, funções de densidade de probabilidade.

$f_j(x) \geq 0$ se $x \in [EFT_j,LFT_j]$

$= 0$ se $x \notin [EFT_j,LFT_j]$ $\forall t_j \in T$

F_0 é o conjunto inicial ou estático de funções de densidade de probabilidade de disparo (F^+ é o conjunto de funções não-negativas). Como o domínio de disparo das transições, o conjunto de funções de densidade de probabilidade F também muda com o estado da rede, tornando-se dinâmico.

A fig. 2.4 exemplifica a RdPTE (f_j indica a função de densidade de probabilidade estática da transição t_j).

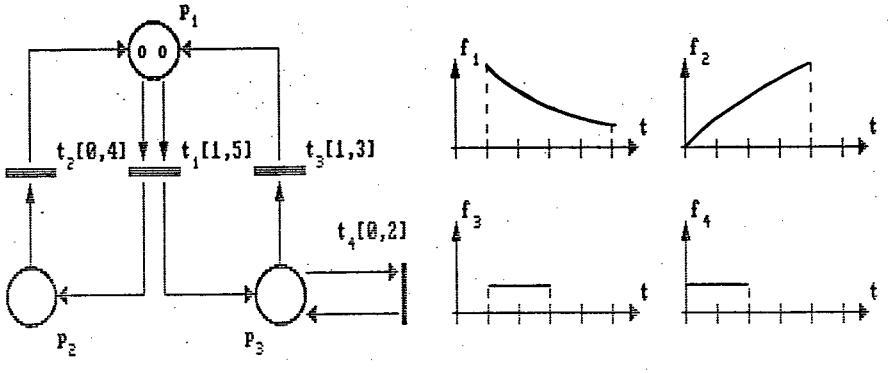

Fig. 2.4: RdP com temporização extendida

Como na RdPT, o estado em uma RdPTE é caracterizado por sua marcação, por seus domínios dinâmicos de disparo e pelas funções de densidade de probabilidade dinâmicas associadas aos mesmos. A regra de mudança de estado envolve, além da

mudança de marcação e domínios, a reavaliação das funções de densidade de probabilidade, sendo detalhada em [Roux 85].

c. RdPs de Alto Nível

A RdP ordinária representa satisfatoriamente a evolução da parte de controle do sistema modelado, mas não permite uma clara representação dos dados sob manipulação. Diversas extensões de RdPs foram propostas de modo a tornar mais concisa a descrição de sistemas complexos, permitindo também a manipulação de dados. Abaixo são destacadas algumas delas:

RdP Colorida: Cada ficha possui uma cor que caracteriza sua identidade. Os arcos são rotulados por expressões definindo o padrão de fichas coloridas a consumir ou produzir em um lugar. Presta-se bem à representação compacta de várias instâncias de uma mesma rede genérica, cada uma identificada por uma cor [Jensen 85].

RdP Predicado-Transição: apresenta fichas parametrizadas por n-uplas de constantes ou variáveis. Os arcos são rotulados por expressões definindo o padrão de fichas a produzir ou consumir. Associa-se a cada transição um predicado indicando as relações entre os valores dos parâmetros das fichas associadas ao disparo da transição [Courtiat 87].

RdP Predicado-Ação: permite distingüir claramente a parte "controle" (uma RdP ordinária) e a parte "dados" de uma especificação. A cada transição da rede é associado um rótulo da forma "when P(x) do x:= F(x)", onde P e F atuam sobre o conjunto X de variáveis que representam a parte "dados" da especificação. O disparo da transição só ocorre quando P for satisfeito [Courtiat 87].

RdP Numérica: semelhante à extensão Predicado-Ação, introduzindo o conceito de fichas com identidade (do mesmo modo que nas RdPs Predicado-Transição) e a distinção entre a condição de sensibilização e a de disparo de uma transição [Diaz 87].

RdP a Objetos: formada pela estrutura de controle própria à RdP e por uma representação de dados na forma de objetos. Sua marcação é feita através de fichas com identidade definida por esses objetos, que são testados, movimentados e alterados pelas transições da rede. O uso da noção de objeto para a representação de dados permite estruturar melhor os dados do sistema e facilita o mapeamento direto dos objetos físicos manipulados [Cantú 90].

Algumas das extensões acima têm um poder de expressão equivalente ao da RdP ordinária, beneficiando-se das opções de análise e validação desta, mas com um poder de modelagem bastante superior, devido à sua concisão [Courtiat 87]. Este é o caso das RdPs Coloridas e Predicado-Transição. As demais possuem um poder de expressão bem maior, restringindo-se no entanto suas possibilidades de análise.

2.2. O Uso da RdP para o Desenvolvimento de Sistemas

A construção de um sistema através de RdP ou outra TDF passa por etapas bem definidas. A partir de sua especificação informal, procede-se à modelagem do sistema usando a TDF escolhida. A especificação formal então obtida pode ser analisada, buscando-se testar sua correção e sua adequação à especificação de referência.

As técnicas de análise empregadas devem prover informação suficiente para orientar possíveis alterações do modelo. Obtendo-se então um modelo correto, sua eficiência pode ser avaliada através de técnicas de análise de desempenho.

O modelo correto e eficiente do sistema pode então ser implementado, ou traduzido para uma linguagem executável. A importância do emprego de uma TDF reside

no fato de a mesma permitir automatizar a maior parte do processo acima descrito [Farines 89].

2.2.1. Técnicas de Modelagem Usando RdP

Uma primeira técnica de modelagem é a abordagem por refinamentos sucessivos. Constrói-se inicialmente um modelo do comportamento geral do sistema, que é aos poucos detalhado através da explosão de nodos em sub-processos completos já completamente validados.

A técnica de refinamento é baseada no conceito de nodo de substituição [Huber 89], que é um lugar ou transição relacionado a um sub-modelo.

O processo de refinamento força uma visão estruturada hierárquica do sistema, facilitando a construção e compreensão do modelo, devido à abstração de detalhes.

Esta técnica permite tanto a construção "top-down", pelo refinamento de nodos em si, quanto a "bottom-up", pela reutilização de sub-estruturas já definidas.

Uma característica importante da modelagem por refinamentos é a validação progressiva do modelo, pois os sub-modelos são validados antes de sua inserção ao modelo global, que por sua vez também é validado antes de recebe-los.

Outra técnica de modelagem é a chamada construção modular de sistemas. Neste caso, o modelo de cada entidade distinta do sistema é construído e validado independente das demais entidades. As interações entre diferentes entidades são representadas por transições associadas a trocas de mensagens (? para recepção e ! para envio), como pode ser observado na fig. 2.5:

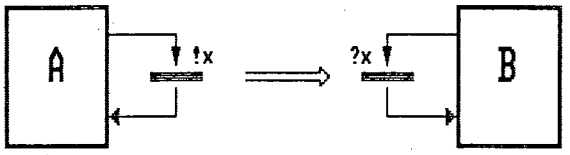

Fig. 2.5: Construção modular

É importante observar que uma entidade pode estar internamente dividida em sub-entidades, cujas interações definem o comportamento da entidade. Isto caracteriza uma visão hierárquica estruturada do sistema, como na modelagem por refinamentos.

A interligação entre os diversos módulos é efetuada via uma estrutura especial denominada meio de comunicação. O comportamento do meio é importante para a determinação do comportamento global do sistema [Souissi 89].

Uma primeira forma de interligação é o uso de um lugar compartilhado, ou envio assíncrono de mensagem, pois utiliza um lugar intermediário para representar a mensagem circulante (fig. 2.6).

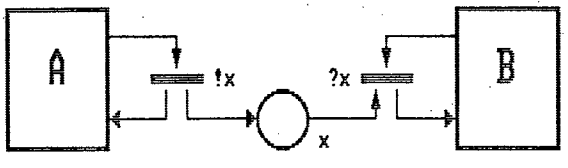

Fig. 2.6: Troca assíncrona de mensagem

É importante observar que no caso assíncrono o número de mensagens em trânsito pode crescer indefinidamente, tornando-se um impecilho ao estudo da rede. Pode-se utilizar um lugar auxiliar para limitar a "n" o número de mensagens em trânsito (fig. 2.7).

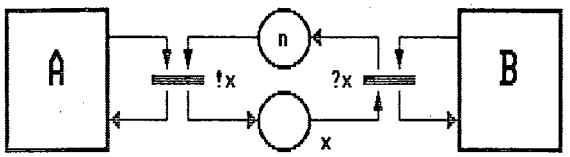

Fig. 2.7: Limitação de mensagens em trânsito

Um caso particular da interligação via lugar compartilhado é a troca de mensagem com reconhecimento, que pode ser decomposta em duas trocas de mensagem assíncronas (fig. 2.8).

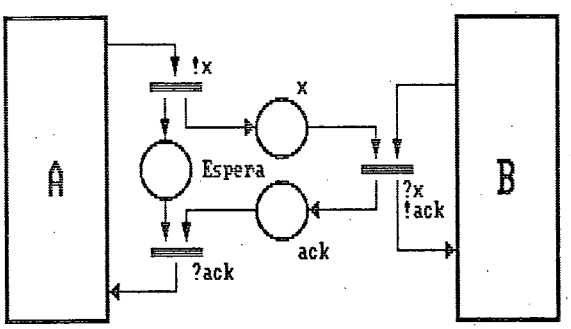

Fig. 2.8: Troca de mensagem com reconhecimento

Outra forma de interligação é a fusão das transições que representam o envio e a recepção de uma mensagem ("rendez-vous"). Este processo caracteriza um envio síncrono de mensagem, no qual somente são liberadas as entidades comunicantes após a troca efetiva da mensagem (fig. 2.9).

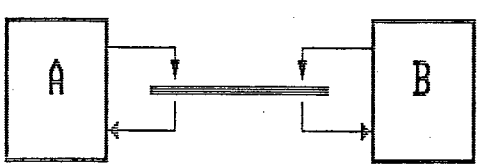

Fig. 2.9: Fusão de transições

Uma visão mais detalhada sobre interligação de processos via meios de comunicação pode ser encontrada em [Souissi 89].

2.2.2. Análise do Comportamento

É necessário verificar que o modelo construído a partir da especificação informal do sistema seja correto, não possuindo erros de modelagem. Para tal é necessário que a RdP apresente um conjunto de boas propriedades: limitação, reiniciação e vivacidade.

Da análise destas propriedades conclui-se sobre a correção do modelo e conhece-se algumas características do comportamento do sistema (como "realizabilidade física", ausência de bloqueios, etc).

O estudo do comportamento do sistema e de sua adequação às especificações desejadas levam à determinação de seu comportamento através da análise estrutural do modelo (concluindo a respeito de seus invariantes) e/ou da verificação (equivalência entre o modelo obtido e um modelo de referência que corresponde às especificações desejadas).

a. As Propriedades Básicas da RdP

Estas propriedades auxiliam na determinação e correção do comportamento dinâmico da rede, ajudando a verificar se está modelando corretamente o sistema desejado e a detectar eventuais erros ou falhas [Esteban 85]. As propriedades básicas mais relevantes são:

Limitação: Uma RdP é K-limitada para uma dada M_0 se:

$$M_i(p_j) \leq K \quad \forall M_i \in [M_0], \forall p_j \in P$$

Se K=1, a rede é dita salva ou binária, pois o número de fichas num lugar é 0 ou 1 (o recurso existe ou não).

Como as fichas representam recursos disponíveis, a limitação de uma rede indica que o sistema modelado possui recursos e operações finitas, como um sistema real [Esteban 85].

Reiniciação: Uma RdP é reiniciável se:

$$\forall M_k \in [M_0] \; \exists s <> \{\} \; / \; M_k \xrightarrow{s} M_0$$

Sendo o sistema reiniciável, a partir de qualquer estado acessível pode-se retornar ao estado inicial executando uma seqüência finita de operações.

Vivacidade: Uma RdP é viva se qualquer transição puder ser disparada a partir de qualquer marcação alcançável:

$$\forall M \in [M_0], \forall t_i \in T, \exists s \supset t_i \; / \; M \xrightarrow{s} M'$$

Se a rede de Petri é viva então não há bloqueios nem partes inativas, pois todas as transições pertencem a alguma seqüência de disparo a partir de qualquer estado.

A partir da vivacidade pode-se derivar uma sub-propriedade chamada quase-vivacidade. Uma transição é quase-viva se pode ser disparada ao menos uma vez a partir de M_0.

Uma marcação em bloqueio ("dead-lock") não sensibiliza nenhuma transição. A vivacidade é condição suficiente, mas não necessária, para a ausência de bloqueios na rede.

Um "live-lock" é um ciclo de disparos de transições repetitivo, sem saída, não passando por M_0.

As propriedades básicas podem ser estendidas para as rdPT e rdPTE, considerando-se estados ao invés de marcações.

Para a determinação das propriedades básicas é necessário construir o grafo de estados acessíveis da rede. A vivacidade e a reiniciação somente podem ser inferidas se a rede for limitada, gerando um grafo finito ("decidability") [Brams 83].

b. Análise Estrutural da Rede

A estrutura da rede permite inferir parcialmente a respeito de suas propriedades básicas. Sendo insensível ao estado inicial da rede, a análise estrutural permite estudar seu comportamento dinâmico sem analisar o grafo de estados.

Resultam da análise estrutural da rede informações sobre seqüências cíclicas de funcionamento (conjuntos de transições com execução cíclica, passando pelos mesmos estados) e componentes conservativas (conjuntos de lugares cuja soma ponderada de fichas é constante para qualquer estado).

Assim, um lugar pertencente a uma componente conservativa é limitado, enquanto uma transição pertencente a uma seqüência cíclica é potencialmente viva. A análise estrutural não se restringe ao estudo das propriedades básicas, pois as seqüências cíclicas e as componentes conservativas fornecem dados importantes sobre o comportamento específico do sistema modelado.

Existem basicamente duas abordagens para a análise estrutural de Redes de Petri: a decomposição em sub-redes e o cálculo de invariantes. Na decomposição são destacadas da RdP sub-redes com características especiais, chamadas Máquinas de Estado (ME) e Grafos de Eventos (GE).

Em uma ME cada transição possui somente um arco de entrada e um de saída, de peso unitário, caracterizando a conservação das fichas circulantes. Em um GE cada lugar possui somente um arco de entrada e um de saída, de peso unitário, indicando ausência de conflitos e comportamento cíclico.

Através do cálculo de invariantes são obtidos os invariantes de lugar (IL) e de transição (IT) da rede. Um IL indica um conjunto de lugares no qual a soma de fichas é constante para qualquer estado alcançável, indicando a conservação de recursos. Toda ME é um IL, mas o contrário não é sempre verdadeiro.

Um IT indica um funcionamento cíclico da rede: o disparo na ordem adequada das transições que o compõem faz a rede voltar à marcação de onde partiu, realizando um ciclo. Todo GE é um IT, mas o contrário não é sempre verdadeiro.

Desta forma, ILs e MEs correspondem às componentes conservativas da rede, enquanto ITs e GEs às suas seqüências cíclicas de funcionamento.

c. Verificação

A verificação de um modelo em RdP busca provar que este satisfaz as especificações de referência desejadas para o sistema que modela, pois mesmo estando o modelo correto sob a ótica das propriedades básicas e dos invariantes, pode não representar exatamente o comportamento desejado do sistema.

Pode-se verificar um modelo de várias formas. Através de simulação pode-se observar seu comportamento, embora seja um método não exaustivo. Pode-se também verificar asserções lógicas (axiomas) sobre seu grafo de estados, provando a validade de certas seqüências de eventos características do sistema [Courtiat 87].

Os métodos de verificação mais completos baseiam-se na noção de equivalência entre o comportamento do modelo e o desejado na especificação do sistema. Neste caso os eventos do sistema são classificados em externos, quando sua ocorrência pode ser observada por uma entidade externa ao mesmo, ou internos, caso contrário.

Três classes de equivalência são especialmente importantes: a de linguagem, a observacional e a forte. Todas se baseiam na comparação do modelo com a especificação, estado a estado, se diferenciando na forma de interpretar os eventos internos [Bougé 88].

É importante observar que, para uma verificação completa do modelo (usando-se equivalência, por exemplo), é necessária a construção de seu grafo de estados, que portanto deve ser finito e factível.

2.2.3. Avaliação de Desempenho

Medir o desempenho de um sistema significa extrair parâmetros indicando o quanto ele é eficiente na realização de seu objetivo. Para tal o modelo do sistema deve representar fielmente o comportamento temporal do sistema, pois o conceito de desempenho está intimamente ligado à noção de tempo.

Basicamente dois métodos são definidos para a medida de desempenho de um sistema a partir de RdP: por análise ou por simulação.

A análise pode ser efetuada via RdPs estocásticas, atribuindo-se a cada transição da rede um tempo de disparo estocástico (probabilidade exponencial decrescente). Obtém-se normalmente seu grafo de estados e trata-se este como um processo markoviano [Diaz 86].

Este método possui uma boa base teórica (processos estocásticos) e seus resultados são bastante precisos, mas a possibilidade de emprego somente de atrasos estocásticos é uma grande limitação à modelagem.

Outra forma de análise é apresentada em [Roux 85], sobre o grafo de classes de estados obtido da rede. Para cada classe de estados é obtido o tempo médio e sua probabilidade de acesso, sendo descartadas as classes com probabilidade inferior a um dado limite.

Provendo resultados bastante precisos, este método envolve manipulações de funções de densidade de probabilidade de forma relativamente complexa (somas, produtos

e integrações de funções exponenciais e normais), o que aumenta bastante seu custo de implementação.

Na avaliação via simulação, o modelo é executado de forma repetitiva, como em sistemas tipo GPSS [Schreiber 74]. Para cada simulação são extraídos os valores de interesse, sendo apresentados ao final como uma média das diversas medições.

A complexidade da rede tem pouca influência na eficiência do método, que depende basicamente de um rápido mecanismo de disparo das transições da rede. Este método é bastante versátil, provendo resultados aproximados mas confiáveis.

2.2.4. Implementação do Modelo

Duas abordagens são normalmente propostas para a execução de especificações modeladas em RdPs: a tradução e a interpretação.

A tradução consiste em transladar a especificação em RdP numa linguagem alvo a ser posteriormente compilada e executada. Este processo torna-se complexo na medida em que cresce o poder de expressão do modelo, como nas RdPs de alto nível [Cantú 90].

A interpretação consiste em examinar o modelo detectando suas transições sensibilizadas, disparar uma delas segundo uma estratégia bem definida e atualizar a marcação, usando um mecanismo independente da especificação. Este processo emprega a especificação validada, o que elimina eventuais erros introduzidos na translação para um código executável. Além disso, sendo o mecanismo de execução independente da especificação, torna-se bastante útil para prototipagem, evitando recompilações.

Entretanto, o principal problema relacionado à interpretação da rede é a baixa velocidade do mecanismo. Técnicas como as apresentadas em [Cantú 90] permitem torná-la mais eficiente.

2.3. Uso de Ambientes para RdP

Como sistemas reais normalmente levam a redes complexas, torna-se necessário o uso de ferramentas de software para construir, validar e implementar o modelo obtido do sistema [Diaz 87].

Este fato levou à construção de ambientes para a utilização do modelo RdP no projeto de sistemas. Tais ambientes auxiliam normalmente na construção e validação do modelo, e mais recentemente em sua implementação.

Um dos primeiros ambientes construídos para RdP foi o OGIVE [Pradin 79], no LAAS-CNRS-France. Tratando redes ordinárias, permite a representação modular de sistemas complexos pela interligação de sub-modelos. Determina as propriedades básicas da rede, através do grafo de estados acessíveis ou de reduções sucessivas do modelo. Determina invariantes de lugar e transição e possui uma versão simples da verificação por equivalência de linguagem.

Outro ambiente relevante é o PROTEAN [Billington 88], baseado em RdPs numéricas. Permite a composição de redes, para auxiliar na construção do modelo. Suas ferramentas de análise são baseadas sobre o grafo de estados acessíveis, permitindo observar as propriedades gerais e estudar as seqüências de eventos aceitas pela rede.

Diversos outros ambientes existem, alguns deles específicos para determinadas aplicações, como protocolos de comunicação. Em [Jensen 86] é realizado um importante estudo crítico sobre características necessárias a ferramentas de software para o auxílio ao projeto com RdPs.

2.4. Conclusão

Neste capítulo, foi introduzido formalmente o modelo RdP e suas principais extensões, tanto a nível de temporização e probabilidade quanto a nível de manipulação de dados.

Também foram apresentadas técnicas de modelagem, validação, verificação e avaliação de desempenho de modelos obtidos em RdP. A possibilidade de automação das técnicas apresentadas foi destacada através da descrição de ambientes para o projeto com RdP.

O próximo capítulo é dedicado à descrição do Sistema ARP e das ferramentas que o compõe.

3. O Ambiente ARP

O esforço de automatizar metodologias de validação e verificação de RdPs no LCMI-DEEL-UFSC deu origem a um sistema que reúne dentro do mesmo ambiente, caracterizado por uma interface homogênea, várias ferramentas para Redes de Petri ordinárias, com temporização e com temporização extendida.

O sistema construído denomina-se ARP (Analisador de Redes de Petri), e faz parte de um projeto homônimo (Ambiente para Redes de Petri), que visa a construção de um ambiente de auxílio à modelagem, validação, verificação e implementação de sistemas usando RdPs a Objeto [Farines 89].

3.1. Descrição Geral do Sistema ARP

A arquitetura do sistema ARP pode ser observada na fig. 3.1. Sua construção modular permite o desenvolvimento e a rápida inserção de novas ferramentas, pois todo acesso a dispositivos (vídeo, teclado, arquivos) estão a cargo de interfaces padrão.

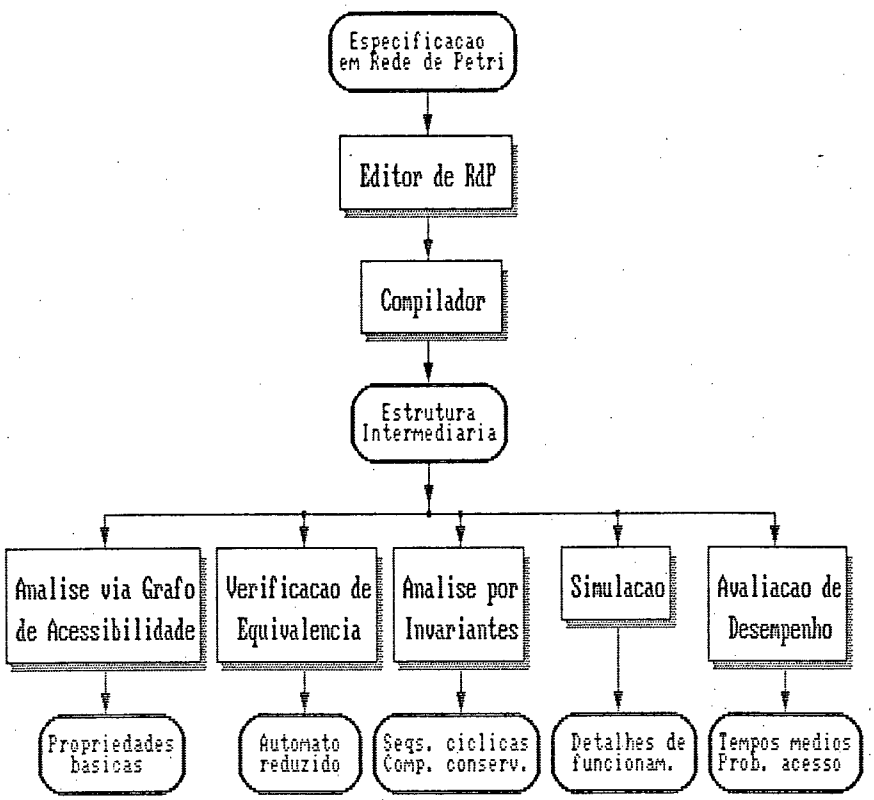

Fig. 3.1: Arquitetura do Sistema ARP

A entrada de uma Rede de Petri no sistema ARP efetua-se através de uma linguagem de descrição (anexo 1), que emprega uma sintaxe semelhante à descrição de dados em Pascal.

Essa linguagem integra a versão atual (2.3) do sistema ARP. Entretanto está em estudo uma nova linguagem compatível à usada no sistema SRPO [Cantú 90], com características de modularidade que simplificam a descrição de redes complexas. Outro objetivo da nova linguagem é a integração do ARP ao SRPO, que passa pela definição de uma linguagem de entrada compatível a ambos.

A descrição da rede na linguagem de entrada é posteriormente compilada na forma de uma estrutura intermediária que contém a descrição da rede e que servirá como entrada das demais ferramentas.

A partir dessa estrutura intermediária e de entradas específicas, cada ferramenta realiza sua operação e produz como saída um texto contendo os resultados obtidos.

3.1.1. Gerenciamento de Interface

A interface do sistema com o usuário está baseada em menus e janelas, visando permitir uma fácil utilização. Além disso, são proporcionadas telas de ajuda em vários pontos do programa.

As funções de interface a dispositivos (vídeo, teclado) foram agrupadas em módulos específicos, proporcionando segurança e rapidez nas alterações em caso de migração para outros equipamentos.

Os módulos de interface que compõem o ARP são:

- **Janelas**: gerenciamento de janelas na tela de texto: abertura, encerramento, movimentação, áreas sobrepostas.
- **Texto**: construção, visualização, edição e apagamento de textos armazenados em memória, resultantes dos módulos de análise ou de descrição de redes.
- **Arquivos**: leitura e escrita de arquivos e gerenciamento de diretório.
- **Teclado**: diversos tipos de menus de escolha e entrada de valores numéricos com crítica, observando os intervalos de validade dos mesmos.

3.1.2. Estrutura Intermediária de Descrição da Rede de Petri

As informações necessárias para caracterizar as RdPs utilizadas no sistema se apresentam, após a compilação da rede editada, sob a forma de uma estrutura de dados denominada "Estrutura Intermediária de Descrição da RdP", que contém os seguintes dados:

- Topologia e dimensões da rede.
- Nome da rede.
- Nome dos nodos da rede (lugares e transições).
- Marcação inicial.
- Temporização estática de cada transição (SEFT,SLFT).
- Curvas de densidade de probabilidade de cada transição (tipo e coeficientes associados).

A topologia da rede é descrita através de um conjunto de listas encadeadas de arcos que simula a notação matricial I(t,p) / O(t,p). São definidas 4 listas encadeadas de arcos:

- PRE_TRANS [i] : arcos que precedem a transição t_i.
- SUC_LUGAR [i] : arcos que sucedem o lugar p_i.
- SUC_TRANS [i] : arcos que sucedem a transição t_i.
- PRE_LUGAR [i] : arcos que precedem o lugar p_i.

Nestas listas, cada arco corresponde à estrutura mostrada na fig. 3.2:

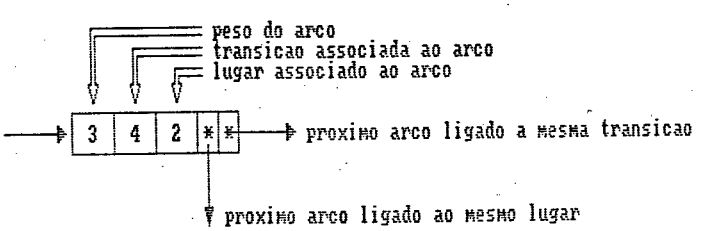

Fig. 3.2: Arco de rede de Petri

A seguir é mostrado um exemplo da estrutura intermediária de descrição da RdP. Para a RdP da fig. 3.3, as figs. 3.4 e 3.5 representam as estruturas de dados que descrevem sua topologia.

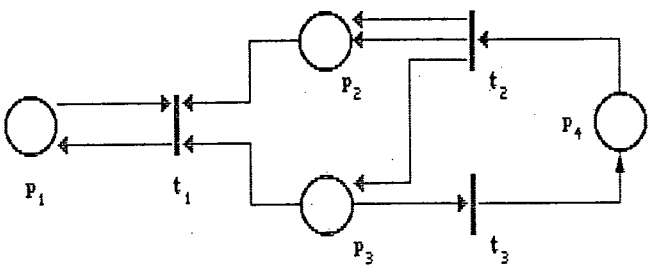

Fig. 3.3: Rede de Petri

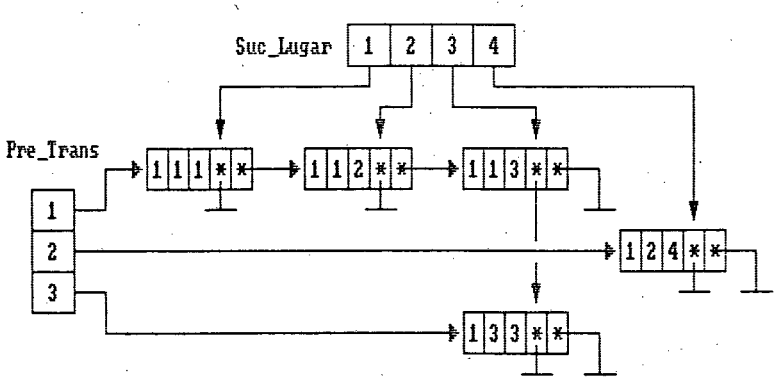

Fig. 3.4: Representação de I (t,p)

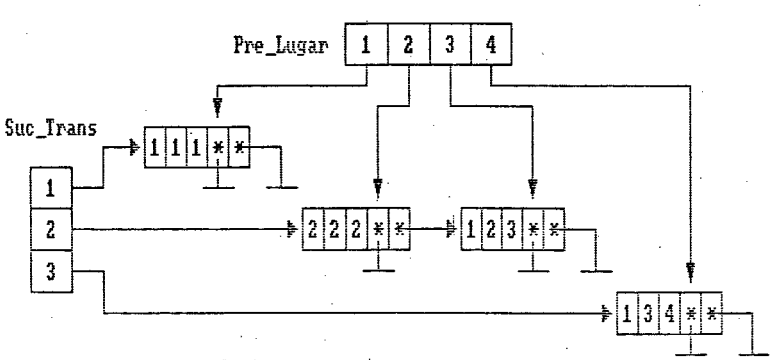

Fig. 3.5: Representação de O (t,p)

Esta estrutura é bastante versátil e compacta, permitindo rápido acesso a todos os arcos ligados a um nodo, seja lugar ou transição.

O armazenamento dos demais dados da rede é simples, sob a forma de vetores de inteiros (para a marcação de cada lugar e a temporização de cada transição), de reais (para os coeficientes das curvas de densidade de probabilidade das transições) ou de strings (para os nomes dos lugares e transições).

3.2. Análise via Grafo de Acessibilidade

Esta análise busca provar as propriedades básicas da rede, através da enumeração de todos os estados alcançáveis a partir de seu estado inicial. Este procedimento dá origem a um grafo de estados acessíveis, sobre o qual são realizadas inferências visando a detecção das propriedades.

3.2.1. Construção do Grafo de Acessibilidade

O disparo de todas as transições sensibilizadas da rede em cada estado (a partir do estado inicial) vai gerando novos estados até esgotar as possibilidades de disparo de transições; isto produz uma árvore, cujos nodos são os estados alcançados, os arcos são os disparos de transições que levam de um estado a outro e a raiz é o estado inicial da rede.

A pesquisa de estados acessíveis em um ramo da árvore encerra ao encontrar um estado terminal (não sensibiliza nenhuma transição) ou um estado duplicado (já enumerado anteriormente). Agrupando-se os estados duplicados da árvore obtida forma-se então o grafo de acessibilidade da rede.

A fig. 3.6 mostra uma RdP e seu grafo de acessibilidade:

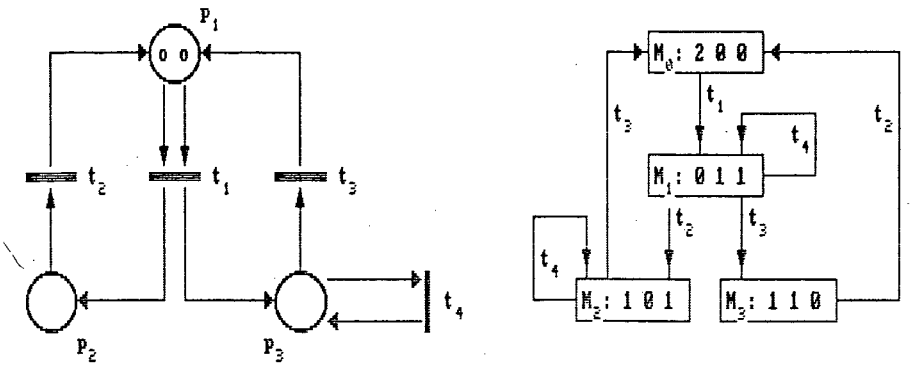

Fig. 3.6: RdP e grafo de acessibilidade

A topologia da rede pode causar o crescimento do número de fichas (fig 3.7a), tornando o grafo infinito (fig. 3.7b):

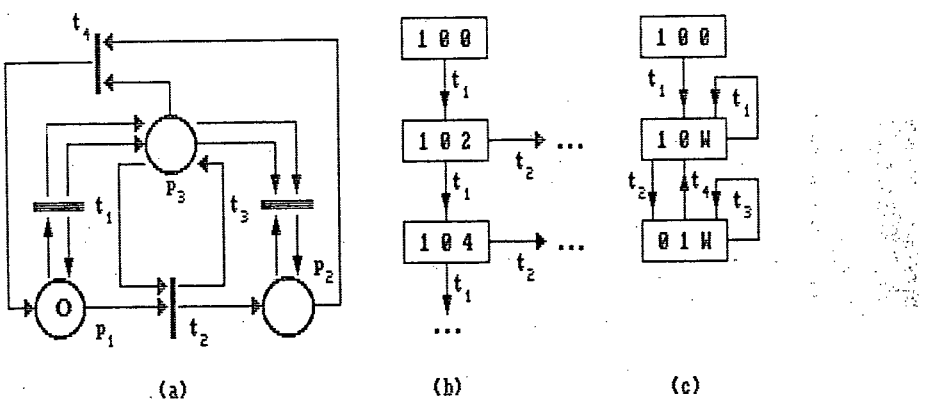

Fig. 3.7: Rede c/ crescimento de fichas

Este problema é contornado em [Peterson 81] usando-se um símbolo especial "W" para simbolizar o crescimento de fichas em um lugar. Assim, o grafo de acessibilidade da rede torna-se finito (fig. 3.7c).

Observando o grafo da fig. 3.7c, nada se pode concluir a respeito das propriedades de reiniciação ou vivacidade, devido à não limitação da rede. Por exemplo, as seqüências de eventos abaixo são possíveis mas não estão representadas:

t_1 t_2 t_3

M_0:(1 0 0) -> (1 0 2) -> (0 1 2) -> (0 1 0): bloqueio

t_1 t_2 t_4 t_2 t_4

M_0:(1 0 0) -> (1 0 2) -> (0 1 2) -> (1 0 1) -> (0 1 1) -> (1 0 0): M_0

O bloqueio e a reiniciação da rede não são indicados no grafo da fig. 3.7c. Uma proposição para contornar tal problema é exposta em [Menasche 82], baseando-se na noção de limiar de um lugar:

$$\text{Limiar}(p) = \text{Max}(I(t,p)) \; \forall t \in T$$

A marcação de lugar só recebe W se tiver crescimento de fichas e for maior que o limiar do lugar. Desta forma o grafo de estados acessíveis reflete melhor o comportamento de uma rede não limitada:

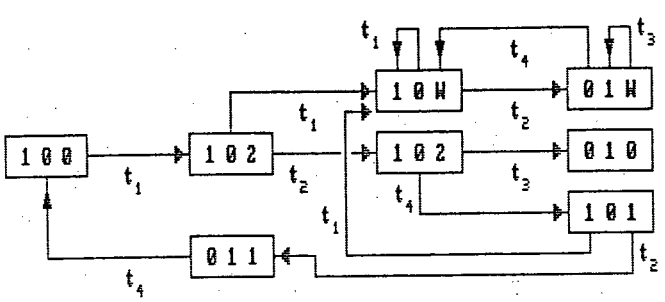

Fig. 3.8: Grafo de acessibilidade considerando limiar

Deve-se observar que a reiniciação e a vivacidade não são decidíveis em RdPs não limitadas. Porém, o uso do limiar permite detalhar melhor a evolução deste tipo de rede e

obter informações complementares.

Durante a construção do grafo é necessário comparar cada estado gerado com os demais já enumerados, buscando duplicações. Para tornar mais rápida esta tarefa, emprega-se uma técnica de "Hashing".

Para cada estado gerado calcula-se um código, função de sua marcação e domínio de disparos. Estados iguais possuem portanto o mesmo código, enquanto estados diferentes possuem em sua maioria códigos diferentes. São mantidas diversas listas de estados alcançados, uma para cada código distinto.

Partindo do código de hashing obtido para o estado gerado, percorre-se a lista dos estados que possuem o mesmo código, comparando somente cada um destes com estado gerado. Caso não seja encontrado estado igual, inclui-se este na lista.

Esta técnica possibilita uma redução significativa no número de comparações necessárias entre estados para a construção do grafo de acessibilidade (em média da ordem de 50 vezes).

3.2.2. Enumeração em Redes com Temporização

A enumeração em RdPT torna-se mais complexa devido à inclusão dos intervalos de disparo das transições pois, sendo a escala de tempo contínua, podem existir infinitos instantes de disparo para uma transição e infinitos estados sucessores, com a mesma marcação mas domínios dinâmicos distintos. Isto torna impossível enumerar todos os estados acessíveis.

Em [Menasche 85] propõe-se um método enumerativo usando classe de estado, que representa a união de todos os estados alcançáveis pelo disparo de uma transição, descritos através da mesma marcação M e domínio dinâmico de disparo D, união dos domínios dos estados pertencentes à classe.

A classe inicial C_0 contém somente o estado inicial da rede, com marcação M_0 e domínio D_0 composto pelos intervalos estáticos [SEFT,SLFT] das transições sensibilizadas por M_0.

O domínio dinâmico de disparo de uma classe de estados compõe-se de dois tipos de equações, sendo Θ_j o instante de disparo de t_j:

$$DEFT_j \leq \Theta_j \leq DLFT_j \quad e \quad \Theta_j - \Theta_k < G_{jk}$$

A matriz G indica as relações entre os instantes de disparo de transições sensibilizadas simultaneamente.

Na implementação atual do sistema ARP, a classe de estados C'(M',D') gerada pelo disparo de uma transição t_k em uma classe C(M,D) é calculada a partir das relações seguintes:

$$M' = M + O(t_k) - I(t_k)$$

$\forall t_j$: se $I(t_j) > M'$ então $D'(t_j)$ é indefinido

senão

se $I(t_j) > M$ então $D'(t_j) = D_0(t_j)$

senão

$$DEFT_j' = Max(0, DEFT_j - Min(DLFT_i \ \forall t_i))$$

$$DLFT_j' = DLFT_j - DEFT_k$$

Esta forma de cálculo apresenta a vantagem da rapidez, mas não considera as relações G_{jk} entre transições sensibilizadas em paralelo.

A simplificação no cálculo do domínio de disparo no sistema ARP visa eliminar a necessidade de programação linear de números inteiros, que implicaria num custo de

implementação e execução elevado. Esta simplificação afeta em geral somente os valores dos domínios associados ao disparo das transições paralelas, preservando o comportamento qualitativo da rede (limitação, vivacidade, reiniciação).

Espera-se, em uma próxima versão do ARP, incluir as relações G_{jk} às equações de cálculo de classes de estados.

3.2.3. Resultados Fornecidos pelo Módulo

As propriedades da rede são verificadas inspecionando-se o grafo de acessibilidade obtido da rede. São indicadas, além das propriedades básicas, as transições que apresentam multi-sensibilização (no caso de RdPT e RdPTE) e as sequências de disparos que levam a "dead-locks", "live-locks" ou crescimento de fichas.

A limitação da rede é observada pela não existência do valor "W" entre as marcações. A rede é reiniciável se o grafo de acessibilidade for fortemente conexo. Cada transição da rede é viva se aparecer ao menos uma vez em cada componente fortemente conexa do grafo.

Cada "dead-lock" corresponde a uma componente fortemente conexa do grafo, com somente um estado. Cada "live-lock" corresponde a uma componente fortemente conexa com mais de um estado e que não contenha o estado inicial.

Investigadas as propriedades, são apresentadas através do módulo de manipulação de textos. Além do texto das propriedades, são construídos textos para a apresentação do grafo de acessibilidade obtido e das marcações ou classes de estados alcançadas.

O módulo implementado possui cerca de 1300 linhas de código fonte para entrada de dados, geração do grafo de acessibilidade, teste das propriedades e escrita dos resultados obtidos. Sua capacidade é cerca de 2000 estados (em micros de 640 kbytes).

3.3. Análise Estrutural de Redes de Petri

Como visto no capítulo anterior, são possíveis dois tipos de abordagens para a análise estrutural: a decomposição em sub-redes, para redes binárias, e o cálculo de invariantes, para o modelo geral.

A decomposição de uma RdP em sub-redes exige um grande esforço computacional no caso de redes complexas, por ser um método de busca exaustiva na rede. Possui também a limitação de só tratar redes com arcos de peso unitário.

O cálculo de invariantes é efetuado através de métodos de álgebra linear aplicados à matriz de incidência da RdP. Apresenta menor esforço computacional e aceitação de redes ponderadas, fornecendo também invariantes que não são ME nem GE. Por isso foi escolhido para implementação no ARP.

3.3.1. Cálculo de Invariantes

Definida a estrutura de uma RdP pela quádrupla R: (P,T,I,O) com "m" transições, "n" lugares e matriz de incidência C = O - I, todo vetor $x \in Z^n > 0$ solução de $C * x = 0$ é um invariante de lugar da RdP, sendo x_j a ponderação do lugar p_j na formação do invariante.

Todo vetor $y \in Z^m > 0$ solução de $C^T * y = 0$ é um invariante de transição, onde y_j indica o número de disparos da transição t_j para formar o ciclo. A seqüência de disparo das transições não é diretamente calculável.

a. Obtenção da Base de Invariantes

O cálculo de invariantes de lugar compõe-se basicamente de duas etapas: cálculo de uma base linearmente independente do espaço solução de C * x = 0 e extração dos invariantes a partir da base obtida.

Para obter a matriz base de invariantes de lugar (B), dada a matriz de incidência da rede (C), deve-se construir uma matriz C' da seguinte forma:

$$C' = [\, C^T \mid I \,]$$

onde $I \in Z^{nn}$ é a matriz identidade de ordem n. Triangulariza-se C' de modo a apresentar-se na forma abaixo:

$$C' = \left[\begin{array}{ccccc|ccccc} x & x & .. & x & x & x & x & .. & x \\ : & : & .. & : & : & : & : & .. & : \\ x & x & .. & x & x & x & x & .. & x \\ \hline 0 & 0 & .. & 0 & 0 & & & & \\ : & : & .. & : & : & & B & & \\ 0 & 0 & .. & 0 & 0 & & & & \end{array}\right] \quad \text{rank } (C^T)$$

$$\underbrace{\qquad\qquad}_{n}$$

A partição inferior esquerda é nula. A sub-matriz B é a base de invariantes de lugar. Para o cálculo da matriz base dos invariantes de transição substitui-se C^T por C na construção de C '. O processo acima é conhecido em álgebra linear por obtenção do "kernel" da matriz C^T.

b. Extração dos Invariantes da Base

A base obtida pode conter valores positivos e negativos, mas os invariantes devem conter unicamente valores positivos, devido a sua própria definição. Além disso, a combinação linear de invariantes também é um invariante; deste modo, deverão ser extraídos da base todos os vetores positivos que representem os invariantes mínimos da rede [Colom 89].

A extração dos invariantes da base obtida inicia-se por seu escalonamento, obtendo um pivot positivo isolado em uma coluna para cada linha da matriz:

$$B = \begin{vmatrix} p_1 & x & .. & 0 & x & .. & 0 & x & .. & 0 & x & .. & x \\ 0 & 0 & .. & p_2 & x & .. & 0 & x & .. & 0 & x & .. & x \\ : & : & & : & : & & : & : & & : & : & & : \\ 0 & 0 & .. & 0 & 0 & .. & 0 & 0 & .. & p_q & x & .. & x \end{vmatrix}$$

onde $p_i \in Z_0^+$ são os pivots de cada linha e $x \in Z$ são inteiros quaisquer. A obtenção dos invariantes mínimos positivos então só pode ser efetuada via sobreposições de linhas (sem troca de sinal, devido aos pivots isolados), para o cancelamento de valores x eventualmente negativos.

A seguir, a extração propriamente dita dos invariantes mínimos a partir da base obtida consiste em efetuar todos os possíveis cancelamentos de elementos negativos da base através de combinações lineares de suas linhas, gerando linhas positivas que são os invariantes desejados. Todas as possíveis opções de cancelamento são testadas, e as novas linhas geradas são incluídas na base.

Para evitar a explosão combinatória de linhas definem-se duas heurísticas: o teste de suporte e o de geradores. Define-se como suporte de um vetor "v" (Sup(v)) o conjunto dos índices de suas colunas não nulas. Desta forma, uma nova linha deve ser incluída na base somente quando seu suporte não contiver nenhum suporte de outra linha da base.

Evita-se assim a inclusão de invariantes não mínimos na base.

Define-se como conjunto gerador de uma linha "i" (Ger(i)) o conjunto de índices das linhas iniciais da base que a geraram. Deve-se somente combinar linhas cujos conjuntos geradores tenham intersecção nula, evitando que a mesma combinação de linhas da base seja efetuada mais de uma vez.

Estabelece-se então o seguinte algorítmo, partindo da base $B \in Z^{qm}$ escalonada e com pivots positivos:

B_{kj}: elemento da linha k e coluna j da matriz B

B^L_k: vetor linha k da matriz B

Início

{ *montar conjuntos geradores das linhas iniciais* }

∀i ∈ [1..q] faça Ger(i) := {i}

{ *para cada coluna da base* }

∀j ∈ [1..m] faça

{ *para cada combinação entre negativo e positivo* }

∀i ∈ [1..q] | B_{ij} < 0 faça

∀k ∈ [1..q] | B_{kj} > 0 faça

{ *testar conjunto de geradores* }

se Ger(i) ∩ Ger(k) = { } então

{ *gerar linha combinação* }

x := B_{kj} * B^L_i - B_{ij} * B^L_k

{ *testar conjunto suporte* }

se ∀ w ∈ [1..q] Sup(B^L_w) ⊄ Sup(x) então

{ *inserir a nova linha na base* }

q := q + 1

B^L_q := x

Ger(q) := Ger(i) U Ger(k)

fim se

fim se

fim para

suprimir de B a linha B^L_i

fim para

fim para

fim

3.3.2. Exemplo de Utilização do Algorítmo

Como ilustração do processo de obtenção dos invariantes descrito, será apresentada a obtenção dos invariantes mínimos de lugar da RdP da fig 3.9.

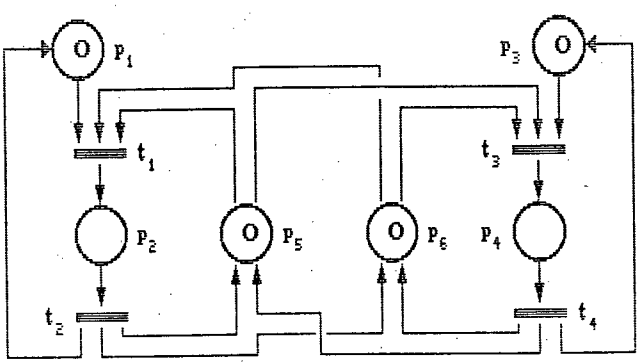

Fig. 3.9: Exemplo de Rede de Petri

A matriz de incidência C é a seguinte (os valores nulos são representados por pontos):

$$C(t,p) = \begin{array}{c|cccccc} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ t_1 & -1 & 1 & . & . & -1 & -1 \\ t_2 & 1 & -1 & . & . & 1 & 1 \\ t_3 & . & . & -1 & 1 & -1 & -1 \\ t_4 & . & . & 1 & -1 & 1 & 1 \end{array}$$

Escalonando-se a matriz transposta aumentada C':

$$C' = \left|\begin{array}{cccccccccc} -1 & 1 & . & . & 1 & . & . & . & . & . \\ . & . & -1 & 1 & . & . & 1 & . & . & . \\ . & . & . & . & 1 & 1 & . & . & . & . \\ . & . & . & . & . & . & 1 & 1 & . & . \\ . & . & . & . & -1 & . & -1 & . & 1 & . \\ . & . & . & . & -1 & . & -1 & . & . & 1 \end{array}\right|$$

Escalonando a matriz base dos invariantes obtida B:

$$B = \left|\begin{array}{cccccc} 1 & 1 & . & . & . & . \\ . & . & 1 & 1 & . & . \\ -1 & . & -1 & . & 1 & . \\ -1 & . & -1 & . & . & 1 \end{array}\right| \quad -> \quad \left|\begin{array}{cccccc} 1 & . & . & -1 & . & -1 \\ . & 1 & . & 1 & . & 1 \\ . & . & 1 & 1 & . & . \\ . & . & . & . & 1 & -1 \end{array}\right|$$

Para cancelar o negativo em B_{14} podem ser combinadas as linhas (1,2) ou (1,3), todas possuindo geradores distintos:

$B^L_5 = B_{24} * B^L_1 - B_{14} * B^L_2$

$= 1 * [\ 1\ .\ .-1\ .-1\ ] - (-1) * [\ .\ 1\ .\ 1\ .\ 1\ ]$

$= [\ 1\ 1\ .\ .\ .\ .\ ]$

$Sup\ (B^L_5) = \{1,2\}$

$Ger\ (B^L_5) = Ger\ (B^L_1)\ U\ Ger\ (B^L_2) = \{1,2\}$

$B^L_6 = B_{34} * B^L_1 - B_{14} * B^L_3 = [\ 1\ .\ 1\ .\ .-1\ ]$

$Sup\ (B^L_6) = \{1,3,6\}$

$Ger\ (B^L_6) = Ger\ (B^L_1)\ U\ Ger\ (B^L_3) = \{1,3\}$

Os suportes de B^L_5 e B^L_6 não contém outros suportes de linha da base, sendo portanto nela inseridos e suprimido B^L_1, cujo negativo foi cancelado:

$$B = \left|\begin{array}{cccccc|l} . & 1 & . & 1 & . & 1 & Ger = \{2\} \\ . & . & 1 & 1 & . & . & Ger = \{3\} \\ . & . & . & . & 1 & -1 & Ger = \{4\} \\ 1 & 1 & . & . & . & . & Ger = \{1,2\} \\ 1 & . & 1 & . & . & -1 & Ger = \{1,3\} \end{array}\right.$$

A próxima coluna a tratar é a 6, devido a B_{36} e B_{56}:

$B^L_6 = 1 * B^L_3 - (-1) * B^L_1 = [\ .\ 1\ .\ 1\ 1\ .\ ]$

$Sup\ (B^L_6) = \{2,4,5\}$

$Ger\ (B^L_6) = Ger\ (B^L_1)\ U\ Ger\ (B^L_3) = \{2,4\}$

$B^L_7 = 1 * B^L_5 - (-1) * B^L_1 = [\ 1\ 1\ 1\ 1\ .\ .\ ]$

$Sup\ (B^L_7) = \{1,2,3,4\}$

$Ger\ (B^L_7) = Ger\ (B^L_1)\ U\ Ger\ (B^L_5) = \{1,2,3\}$

$B^L{}_6$ pode ser incluida à base, pois seu suporte não contém o suporte de outra linha e seus geradores são distintos. O suporte de $B^L{}_7$ contém os suportes de $B^L{}_2$ e $B^L{}_4$, impedindo sua inclusão. Não restando mais negativos na base, os invariantes mínimos de lugar da rede da fig. 3.9 são:

$$B = \left|\begin{array}{cccccc|l} . & 1 & . & 1 & . & 1 & = \{p_2\ p_4\ p_6\} \\ . & . & 1 & 1 & . & . & = \{p_3\ p_4\} \\ 1 & 1 & . & . & . & . & = \{p_1\ p_2\} \\ . & 1 & . & 1 & 1 & . & = \{p_2\ p_4\ p_5\} \end{array}\right.$$

3.3.3. Resultados Fornecidos pelo Módulo

Além da apresentação da base de invariantes e dos invariantes mínimos obtidos, são efetuados testes sobre estes. O teste de sub-redes indica quais invariantes de lugar são Máquinas de Estados ou quais invariantes de transição são Grafos de Eventos.

O teste de cobertura indica quais nodos (lugares ou transições) pertenceram a todos os invariantes e quais não pertenceram a nenhum dos invariante obtidos. Em particular, uma transição pertencente a todos os invariantes da rede está presente em todas as suas seqüências cíclicas; caso não pertencer a nenhum invariante, a transição não é viva, e portanto a rede também não o é.

O sistema ARP permite ao usuário a definição de um conjunto de nodos obrigatórios e um de nodos proibidos. O sistema então apresenta como resultado somente os invariantes que passam pelos nodos obrigatórios e que não passam pelos nodos proibidos indicados, permitindo ao usuário precisar as características dos invariantes que deseja extrair.

Deve-se observar que para esta análise não consideram-se as restrições temporais da rede, mas somente sua topologia. Portanto, alguns invariantes de transição detectados

para RdPTs podem não existir, caso as restrições temporais o impeçam. Os invariantes de lugar não são afetados pela temporização da rede [Roux 85].

3.4. Verificação de Equivalência de Linguagem

A verificação de equivalência consiste em provar que o modelo formal da especificação do sistema se comporta de modo equivalente à especificação de referência do mesmo [Bolognesi 87b]. Uma das formas de estudar a equivalência entre ambos passa pela utilização de sistemas de transição (autômatos de estados finitos).

Classificam-se os eventos de um sistema como visíveis (observáveis ou externos), quando sua ocorrência é detectável por uma entidade externa ao mesmo, ou invisíveis (não-observáveis ou internos) caso contrário.

A equivalência entre dois sistemas de transição pode ser definida em vários níveis [Bougé 88], dos quais três são aqui destacados. O primeiro nível, denominado equivalência de linguagem ou de traços máximos, verifica-se quando a forma mínima das linguagens geradas (seqüências de eventos observáveis) por ambos for igual.

Por outro lado temos as formas de equivalência baseadas no conceito de bissimulação entre estados. Dois estados bissimilares devem estar aptos a simular um ao outro, em termos de seqüências de eventos, levando a estados ainda bissimilares.

A equivalência forte leva em conta todos os eventos dos sistemas considerados (visiveis e invisíveis), enquanto a observacional considera somente seus eventos visíveis [Bolognesi 87b, Bougé 88]. A equivalência forte é mais restritiva que a observacional, e esta que a de linguagem.

Num primeiro tempo implementou-se no sistema ARP a verificação de equivalência de linguagem. As demais classes de equivalência serão objeto de estudos futuros.

3.4.1. O Processo de Verificação de Equivalência

O processo de verificação de equivalência de linguagem inicia-se com a obtenção do autômato finito de estados que representa o comportamento do modelo. Em RdP este é obtido a partir da construção de seu grafo de acessibilidade.

Indicam-se no autômato como eventos visíveis aqueles relacionados ao comportamento que se deseja verificar, e invisíveis os demais.

A seguir transforma-se o autômato de estados finito não-determinista em seu equivalente determinista, e minimiza-se seu número de estados. Os algorítmos para tais operações são bastante conhecidos, podendo ser encontrados em [Aho 86].

O próximo passo consiste em comparar o autômato determinista obtido (representando o comportamento da implementação do sistema), e um autômato finito determinista de referência que representa a especificação do comportamento desejado para o sistema.

Essa comparação é efetuada através do mesmo algorítmo de minimização de estados descrito em [Aho 86], pois o processo de minimização de estados consiste em agrupar os estados de um autômato em grupos de estados equivalentes (em linguagem).

Para tal submetem-se ambos os autômatos simultaneamente ao algorítmo, como se fossem um autômato único, obtendo-se então os grupos de estados equivalentes. Se cada grupo de estados equivalentes obtido contiver ao menos um estado de cada autômato, então ambos são equivalentes em linguagem.

Exemplos detalhados do emprego desta técnica podem ser observados no cap. 4 e também em [Novali 86].

3.4.2. Resultados Fornecidos pelo Módulo

A partir da definição, pelo usuário, das transições da rede a considerar visíveis, são fornecidos como resultados o autômato determinista mínimo representando o

seqüenciamento entre as transições visíveis e também os caminhos (seqüências de disparo) possíveis nesse autômato.

Caso também seja definida a especificação do comportamento desejado, esta é comparada com o autômato mínimo obtido do modelo, sendo então apresentados os grupos de estados equivalentes entre ambos, e a conclusão sobre sua equivalência ou não.

A verificação de equivalência de linguagem permite a observação de uma abstração do comportamento do sistema, sendo mais eficiente na redução da complexidade do modelo que as técnicas de redução de RdP usando regras estruturais, pois as regras de redução de autômatos são mais eficazes e genéricas que as de redução de redes [Dufau 84].

3.5. Simulação de Redes de Petri

A simulação de um modelo em RdP é uma importante forma de verificar seu comportamento, especialmente quando outras formas (via equivalência, por exemplo) não são factíveis devido à não-limitação da rede ou tamanho excessivo do grafo de acessibilidade.

Além disso, por permitir efetuar a evolução passo-a-passo de uma RdP, a simulação torna-se uma importante ferramenta para a depuração do modelo, permitindo compreender melhor o comportamento do sistema e suas limitações.

Observa-se porém que, sendo um processo não-exaustivo, os resultados fornecidos por uma simulação são dependentes do usuário, que determina o comportamento a estudar e detecta eventuais incorreções.

3.5.1. Descrição do Simulador

O ARP possui um simulador que permite efetuar a evolução passo-a-passo de uma RdP, provendo diversas facilidades que tornam seu uso simples e eficiente.

No simulador construído, são apresentadas ao usuário as transições disparáveis no estado atual (composto por sua marcação e seu domínio de disparos). Este escolhe então uma para disparar (e define seu instante de disparo, em caso de RdPT).

Com o disparo da transição escolhida um novo estado é criado, sendo considerado como novo estado atual. O estado anterior é então armazenado em uma lista de estados percorridos, possibilitando futuros retornos.

Pode-se alterar a marcação ou o domínio de disparos do estado atual da simulação. Um histórico de simulação registra as alterações efetuadas, transições disparadas e estados percorridos, podendo ser consultado a qualquer instante.

Além disso, determinados estados de interesse podem ser memorizados com um nome dado pelo usuário, podendo ser visualizados ou restaurados como estado atual a qualquer instante.

O comandos disponíveis no simulador são:

- **Disparar:** apresentam-se as transições disparáveis, das quais uma é escolhida para disparo.
- **Retornar:** permite retornar a um estado anterior na lista de estados percorridos, caso não esteja vazia.
- **Editar:** permite alterar a marcação ou os intervalos dinâmicos de disparo do estado atual.
- **Visualizar:** permite observar estados da lista de estados já percorridos.
- **Memorizar:** memorização de estados de interesse, para retorno posterior, usando nomes dados pelo usuário.
- **Histórico:** consulta ao histórico de simulação.

Para melhor acompanhar a simulação, apresenta-se na tela do simulador a marcação do estado atual e uma representação resumida da lista de estados percorridos, na qual são indicadas eventuais duplicações de estados ou bloqueios.

Essa representação contém ainda o nome de cada estado percorrido, se memorizado, ou sua profundidade em relação ao estado inicial, o nome de cada transição disparada e o número de transições ainda não disparadas em cada estado.

3.6. Avaliação de Desempenho em Redes de Petri

A avaliação de desempenho de um sistema a partir de seu modelo em RdPTE pode ser analítica ou baseada numa estatística obtida por repetidas simulações aleatórias da rede.

Sendo a primeira forma bastante complexa para implementação, foi escolhida para o ARP a segunda forma de avaliação.

3.6.1. Descrição Geral do Avaliador de Desempenho

Duas opções são possíveis no método empregado: uma que privilegia as medidas entre estados e outra entre eventos (disparos de transições).

Na opção a estados, o usuário define um estado inicial e "n" estados destino para a rede. A rede evolui a partir do estado inicial até alcançar um dos estados destino, quando é armazenado o valor obtido para o tempo de acesso àquele destino. O estado inicial é restaurado e o processo de evolução repete-se até os valores dos tempos médios de acesso atingirem a precisão desejada.

Na opção a eventos são determinados os eventos (transições) inicial e destinos, e a rede evolui livremente a partir de seu estado inicial. A contagem do tempo inicia-se na

ocorrência do evento inicial e termina na ocorrência de um evento destino, quando é armazenado o tempo de acesso obtido àquele destino. Neste momento pode ou não ser restaurado o estado inicial da rede, a critério do usuário. O processo repete-se como na opção a estados.

Para contornar situações de bloqueio ("dead-lock" e "live-lock") são definidos outros critérios para o fim de um ciclo: nenhuma transição sensibilizada ou excesso de transições disparadas em um ciclo.

Usando este método podem ser obtidos diversos parâmetros que permitem avaliar o desempenho de um sistema, como o tempo médio de acesso a um estado, atraso médio entre eventos e probabilidade de acesso a um estado ou entre eventos. É também possível extrair valores relativos aos nodos da rede, como atraso médio de disparo e número médio de disparos de cada transição por ciclo e marcação média de cada lugar.

Para bem definir o comportamento que deseja estudar, o usuário tem acesso às seguintes opções de entrada: marcações inicial e destinos, eventos inicial e destinos, nº máximo de disparos, precisão desejada para os resultados e definição de probabilidade de disparo entre transições conflitantes (ponderação de conflitos).

Na opção a eventos, ao ser alcançado um evento destino, a marcação inicial da rede pode ou não ser restaurada para o início de um novo ciclo, também a critério do usuário. A não restauração da marcação causa uma evolução livre da rede.

Durante a avaliação são indicados na tela os valores atuais dos tempos médios de acesso e o número de acessos efetuados a cada destino, dando uma visão da convergência dos valores finais ao usuário, que pode decidir sobre o término da avaliação.

3.6.2. O Mecanismo de Evolução da Rede

O núcleo do módulo de avaliação de desempenho é o mecanismo de evolução da rede, que se encarrega da determinação e disparo das transições sensibilizadas, da

atualização do estado da rede e da gerência do tempo.

Este mecanismo baseia-se em uma fila de transições sensibilizadas, ordenadas por instantes de disparo crescentes, para que a primeira transição da fila sempre seja a próxima a disparar. Os instantes de sensibilização e disparo das transições são controlados através de um contador de tempo virtual, que é atualizado a cada disparo de transição.

Para acelerar a determinação das transições sensibilizadas define-se para cada transição da rede um índice de sensibilização I_{SENS}, que é atualizado a cada disparo de transição:

$$I_{SENS}(t_j) = \sum I(t_j,p_k) \quad \forall p_k / I(t_j,p_k) > M(p_k)$$

Se uma transição t_j está sensibilizada em uma marcação M, seu índice de sensibilização I_{SENS} será nulo.

No disparo de uma transição são retiradas fichas de seus lugares de entrada, dessensibilizando transições, e colocadas fichas em seus lugares de saída, sensibilizando novas transições. O disparo de uma transição efetua-se através dos seguintes passos:

- Retira-se a primeira transição da fila (t_d), para que seja disparada.
- O contador de tempo virtual é atualizado com o instante de disparo de t_d.
- Cada lugar de entrada de t_d tem atualizada sua marcação e os I_{SENS} das transições que o têm como lugar de entrada. Caso alguma destas transições seja dessensibilizada, é retirada da fila de sensibilizadas.
- Cada lugar de saída de t_d tem atualizada sua marcação e os I_{SENS} das transições que o têm como lugar de entrada. Caso alguma transição seja sensibilizada, é sorteado um instante de disparo para a mesma e insere-se-a na fila, na posição referente a seu instante de disparo.

3.6.3. Disparo de Transições em Instantes Aleatórios

O sorteio do instante de disparo de uma transição deve considerar seu intervalo estático de disparo e sua função de densidade de probabilidade distribuída sobre esse intervalo.

Para cada função de densidade de probabilidade distinta é construída (por integração) uma curva de distribuição acumulada de probabilidade, sempre a partir da função de densidade de probabilidade normalizada, com área subjacente unitária e definida sobre o intervalo [0,1].

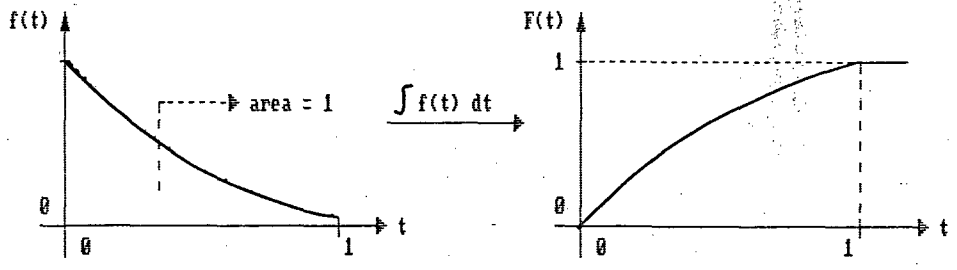

Fig. 3.11: Funções de densid. e distrib. de probabilidade

O sorteio do instante de disparo de uma transição, respeitando seu intervalo e sua função de densidade de probabilidade, passa pelos seguintes passos:

- Sorteio de um valor aleatório real com distribuição uniforme sobre o intervalo [0,1].
- Ponderação deste valor através da curva de distribuição acumulada de probabilidade:

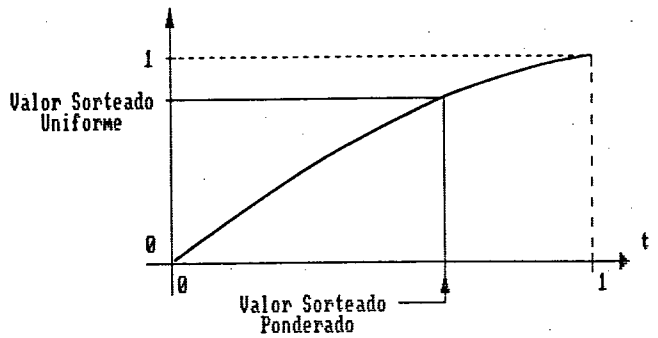

Fig. 3.12: Ponderação do valor aleatório

- Ajuste do valor sorteado e ponderado ao intervalo de disparo estático da transição:

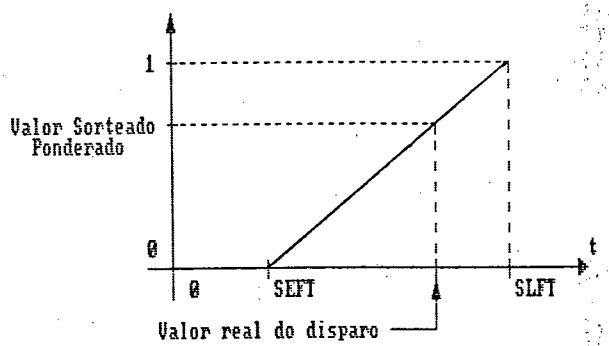

Fig. 3.13: Ajuste do valor aleatório

3.6.4. Ponderação de Conflitos

Em certas circunstâncias é interessante poder forçar uma transição a disparar mais freqüentemente que outras em conflito com ela, fixando probabilidades de disparo entre as transições em conflito.

Denomina-se grupo de conflito o conjunto de todas as transições que possuem os mesmos lugares de entrada (arcos de mesmo peso), com intervalos de disparo iguais e definidos sobre o mesmo ponto ($SEFT_k = SLFT_k$). Assim, todas as transições de um

mesmo grupo de conflito são sensibilizadas simultaneamente, com o mesmo instante de disparo.

Quando uma transição de um grupo de conflito é a próxima a disparar (primeira da fila de sensibilizadas), o mecanismo de evolução da rede escolhe aleatoriamente, respeitando as probabilidades fixadas, uma das transições de seu grupo, que troca de lugar na fila com a primeira, e a dispara.

3.7. Conclusão

Este capítulo apresentou o Sistema ARP, destacando sua construção modular e integrada e descrevendo detalhadamente cada uma das ferramentas que oferece para análise e simulação de Redes de Petri.

Esse sistema foi desenvolvido para equipamentos compatíveis IBM-PC, sob sistema operacional DOS, e seu código fonte em Pascal possui cerca de 11.000 linhas. Devido às limitações do equipamento, são aceitas no ARP redes com até 150 lugares e 150 transições.

O próximo capítulo buscará demonstrar a utilidade do ARP, através da aplicação de suas ferramentas as estudo de problemas de Automação Industrial e Sistemas Distribuídos.

4. Utilização das Ferramentas do Sistema ARP

Neste capítulo serão apresentados alguns exemplos da utilização do ambiente ARP, com o objetivo de verificar as potencialidades de suas ferramentas para análise do comportamento e do desempenho de sistemas modelados por Redes de Petri.

A escolha dos exemplos visou a cobertura das várias ferramentas integradas no ambiente ARP. Em cada aplicação pretende-se mostrar a utilização de uma ou mais ferramentas. Desta forma, para cada exemplo é apresentado o problema de análise a ser resolvido, o estudo realizado em termos de modelagem e análise do sistema e os resultados deste.

Os exemplos escolhidos são ligados a problemas reais dentro das áreas de interesse do LCMI-EEL: Automação Industrial e Sistemas Informáticos Distribuídos. Entretanto a aplicação das ferramentas do ambiente não se restringe a estas áreas, abrangendo outros problemas que possam ser modelados via Redes de Petri.

4.1. Uso do Ambiente ARP na Verificação do Comportamento: Aplicação ao Protocolo de Comunicação MMS

O exemplo proposto visa apresentar o uso do Ambiente ARP para a verificação do comportamento da parte do protocolo MMS que trata da execução ou cancelamento de um serviço, no caso dos serviços confirmados.

Do uso da ferramenta de verificação sobre o modelo RdP deste protocolo concluir-se-á a respeito da adequação da especificação da parte analisada do protocolo MMS em relação ao comportamento desejado descrito na norma [ISO 9506], bem como de eventuais modificações deste.

4.1.1. Descrição da Aplicação

A Especificação de Mensagens da Manufatura (MMS, [ISO 9506]), define um protocolo para a camada aplicação do padrão MAP, que permite a troca de mensagens entre equipamentos programáveis heterogêneos em ambientes industriais.

A execução de um serviço MMS baseia-se no prévio estabelecimento de uma conexão entre a entidade que requisita o mesmo (requisitor) e a que responde (respondedor). Neste exemplo estudar-se-á a interação entre estas entidades durante a execução de um serviço MMS confirmado.

Uma conexão entre dois usuários MMS, tratando um serviço confirmado, pode ser vista da seguinte forma:

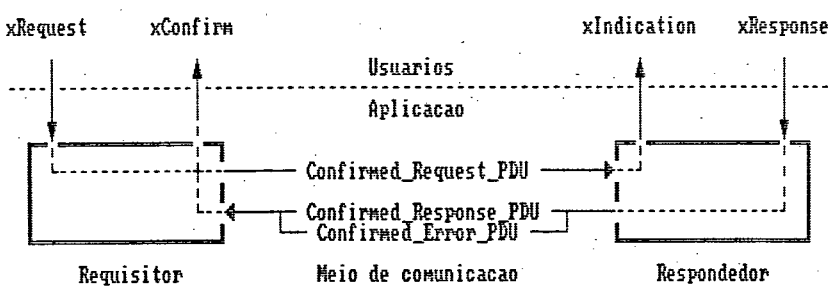

Fig. 4.1: Serviço MMS confirmado

As primitivas do serviço MMS confirmado são:

x_Request : pedido de execução do serviço X.

x_Confirm : indicação de aceite (*x_Confirm*+) ou recusa (*x_Confirm*-) do serviço X.

x_Indication : indicação de um pedido de serviço X.

x_Response : aceite (*x_Response*+) ou recusa (*x_Response*-) de execução do serviço X.

As PDUs ("Protocol Data Units") trocadas entre entidades do protocolo significam:

Confirmed_Request : pedido de serviço confirmado.

Confirmed_Response : indicação de aceite.

Confirmed_Error : indicação de recusa.

O cancelamento de um pedido de serviço confirmado comporta-se do mesmo modo, através das primitivas *Cancel_Request*, *Cancel_Confirm+* e *Cancel_Confirm-*, e das PDUs *Cancel_Request_PDU*, *Cancel_Response_PDU* e *Cancel_Error_PDU*:

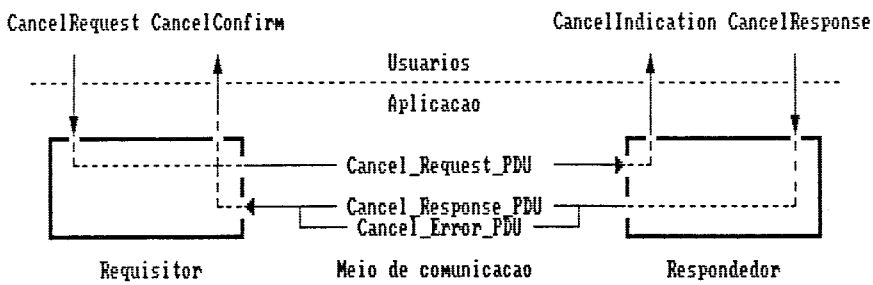

Fig. 4.2: Cancelamento de pedido de serviço MMS confirmado

Será aqui estudado o comportamento no caso de pedido de cancelamento por parte do requisitor. Para tal será construído um modelo do protocolo em RdP, a partir dos modelos das entidades e dos meios de comunicação.

A descrição do protocolo, incluindo as máquinas de estados das entidades, é encontrada na norma [ISO 9506].

4.1.2. Modelagem das Entidades MMS em Redes de Petri

A construção do modelo de um protocolo de comunicação segue uma metodologia bem definida. Seguindo o princípio de estruturação de protocolos em camadas, definido

pelo modelo de referência da ISO, cada camada (N) fornece um serviço (N) à camada (N+1) através de funções realizadas em seu interior usando serviços da camada inferior (N-1) [Courtiat 87].

A fig. 4.3 representa a forma geral de uma camada (N) de um protocolo:

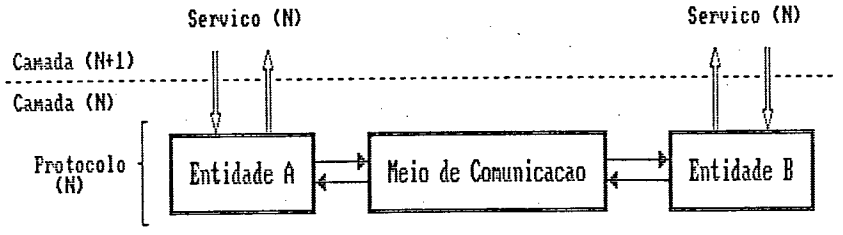

Fig. 4.3: Modelo de protocolo em camadas

A partir do modelo em RdP de cada entidade e do meio de comunicação, construir-se-á o modelo global do protocolo no caso do serviço MMS confirmado.

a. Requisitor do Serviço

O modelo do requisitor do serviço, baseado na máquina de estados indicada na norma [ISO 9506], é apresentado na fig. 4.4. A cada transição do modelo são associadas indicações de envio (!) e recepção (?) de mensagens (primitivas de serviço ou PDUs):

s_1: ? *x_Request* / ! *Confirmed_Request_PDU*

s_2: ? *Confirmed_Response_PDU* / ! *x_Confirm+*

s_3: ? *Confirmed_Error_PDU* / ! *x_Confirm-*

s_4: ? *Cancel_Request* / ! *Cancel_Request_PDU*

s_5: ? *Cancel_Error_PDU* / ! *Cancel_Confirm-*

s_6: ? *Cancel_Response_PDU & Confirmed_Error_PDU /*

! Cancel_Confirm+ & x_Confirm-

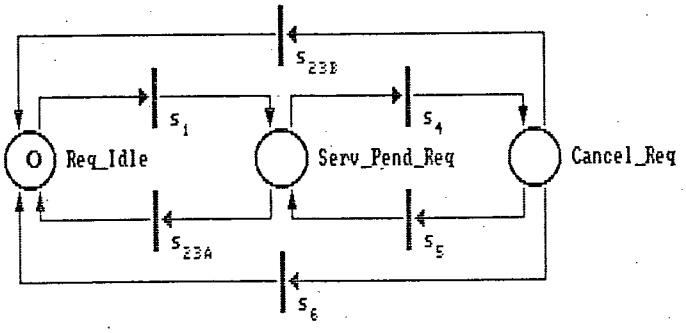

Fig. 4.4: Modelo do requisitor

Para simplicidade do modelo, os eventos s_2 e s_3 foram reunidas em um único evento s_{23}, cuja ocorrência é modelada pelas transições s_{23A} e s_{23B}.

b. Respondedor do Serviço

O modelo do respondedor é apresentado na fig. 4.5, cujas transições correspondem aos seguintes eventos:

t_1: ? *Confirmed_Request_PDU* / ! *x_Indication*

t_2: ? *x_Response+* / ! *Confirmed_Response_PDU*

t_3: ? *x_Response-* / ! *Confirmed_Error_PDU*

t_4: ? *Cancel_Request_PDU* / ! *Cancel_Indication*

t_5: ? *Cancel_Response-* / ! *Cancel_Error_PDU*

t_6: ? *Cancel_Response+ & x_Response-* /

! Cancel_Response_PDU & Confirmed_Error_PDU

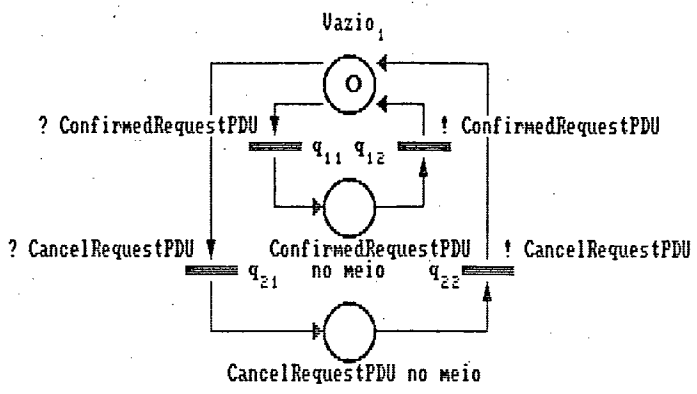

Fig. 4.6: Meio no sentido requisitor-respondedor

No sentido respondedor-requisitor fluem as PDUs *Confirmed_Response_PDU*, *Confirmed_Error_PDU*, *Cancel_Response_PDU* e *Cancel_Error_PDU*. As duas primeiras serão consideradas uma só, devido às uniões de eventos s_{23} e t_{23}, nos modelos do requisitor e do respondedor:

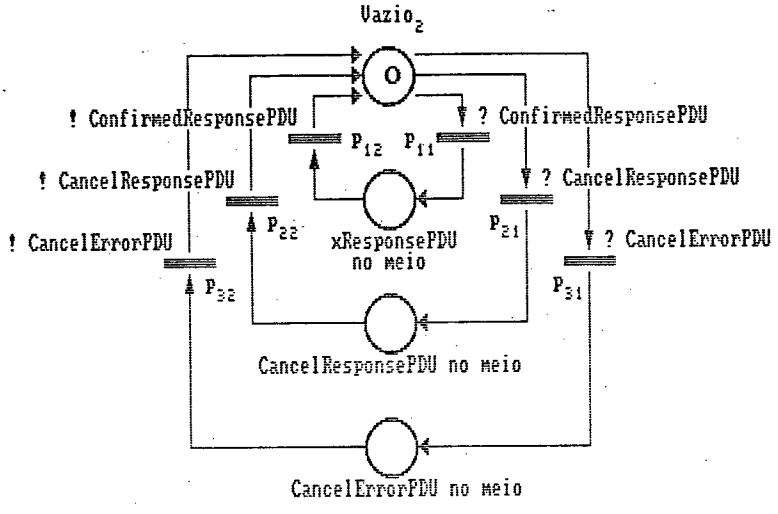

Fig. 4.7: Meio no sentido respondedor-requisitor

O modelo global é construído por fusão das transições de emissão e recepção das PDUs, entre as entidades e os meios de comunicação (a transição p_{12} é duplicada, devido à existência das transições s_{23A} e s_{23B}). É efetuada a fusão entre as seguintes transições:

Requisitor		Respondedor	
s_1	$+ q_{11}$	t_1	$+ q_{12}$
s_{23A}	$+ p_{12}$	t_{23}	$+ p_{11}$
s_{23B}	$+ p_{12}$	t_4	$+ q_{22}$
s_4	$+ q_{21}$	t_5	$+ p_{31}$
s_5	$+ p_{32}$	t_6	$+ p_{21}$
s_6	$+ p_{22}$		

4.1.3. Análise do Modelo Global

A análise do modelo global obtido para a parte do protocolo MMS que trata dos serviços confirmados busca comprovar sua correta construção e a equivalência de seu comportamento com as especificações constantes na norma, que guiaram a modelagem. Esta verificação garantiria a correção do modelo da implementação.

a. Validação do Modelo Global Obtido

Analisando o modelo obtido através do módulo de análise via construção do grafo de acessibilidade do ARP constata-se um bloqueio, observado no grafo de estados obtido:

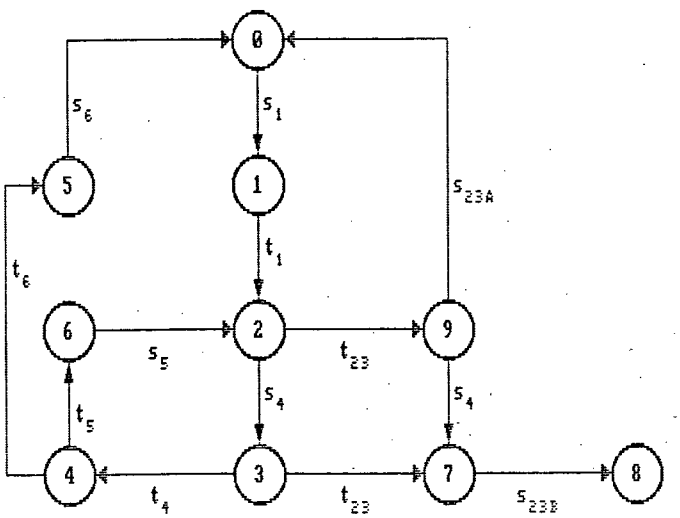

Fig. 4.8: Grafo de estados do modelo global

A marcação na qual ocorre o bloqueio é M_8: {Req_Idle, Rsp_Idle, $Vazio_2$, CancelRequestPDU}, que pode ser alcançada pela seqüência de disparos {s_1, t_1, s_4, t_{23}, s_{23B}}. O pedido de cancelamento *Cancel_Request_PDU*, não recebido pelo respondedor (esta situação é geralmente denominada "recepção não especificada" [Courtiat 87]), impede o requisitor de utilizar o meio de comunicação para um novo pedido de serviço, bloqueando o sistema [Pinho 88].

O ítem 6.4 da norma [ISO 9506] indica que "*na recepção de um Cancel_Request_PDU referente a um pedido desconhecido, o respondedor deve enviar uma Cancel_Error_PDU ao emissor do pedido de cancelamento. O usuário do serviço não é informado sobre o ocorrido*". Portanto deveria ser incluído à especificação e ao modelo do respondedor um novo evento possível t_7:

t_7: ? *CancelRequest_PDU* / ! *Cancel_Error_PDU*

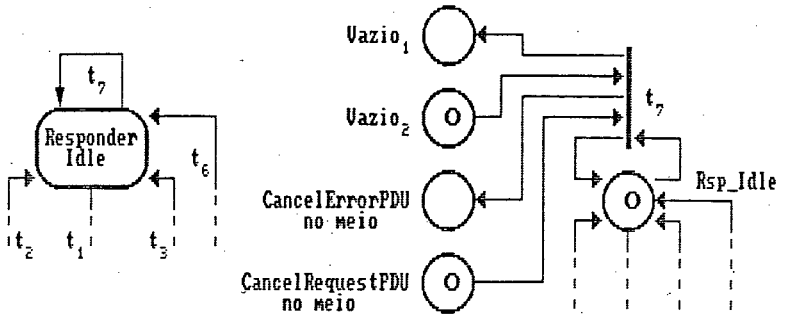

Fig. 4.9: Alteração no modelo do respondedor

Caso o comportamento do requisitor não seja redefinido, o mesmo problema ocorrerá pela recepção não especificada de *Cancel_Error_PDU*. Segundo o ítem 6.3.11 de [ISO 9506]: "*se após o pedido de cancelamento uma Confirmed_Error_PDU ou uma Confirmed_Response_PDU for recebida, o cancelamento é considerado como tendo falhado e uma Cancel_Error_PDU deve ser recebida*". Desta forma, a especificação e o modelo do requisitor devem incluir um novo evento s_7:

s_7: ? *Cancel_Error_PDU*

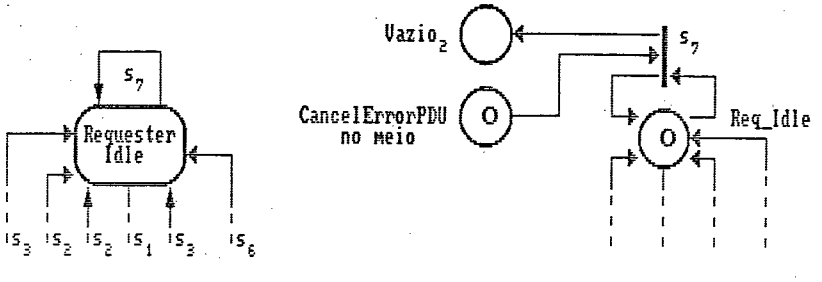

Fig. 4.10: Alteração no modelo do requisitor

Analisando-se o modelo global modificado do protocolo, através do ARP, observa-se que possui as propriedades básicas, (limitação, vivacidade e reiniciação) estando portanto eliminado o bloqueio anteriormente observado.

b. Verificação do Modelo Global

A observação das propriedades básicas não implica na correção do comportamento do modelo frente às especificações de referência. Esta correção pode ser verificada através do módulo de verificação de equivalência de linguagem do ARP.

Para a verificação do comportamento do requisitor, são considerados visíveis somente os eventos relacionados às primitivas do serviço que fornece (s_1 a s_6), no modelo global modificado. Obtém-se então o autômato de estados que representa seu comportamento e verifica-se sua equivalência com a especificação.

O autômato de estados obtido pelo módulo de verificação de equivalência de linguagem do ARP para o requisitor é indicado na fig. 4.11, e coincide com o apresentado na norma [ISO 9506]:

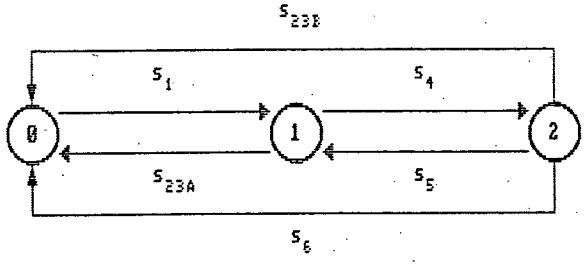

Fig. 4.11: Autômato do requisitor

Para o respondedor (eventos visíveis: t_1 a t_6), o ARP obtém o seguinte autômato de estados, também coincindindo com o desejado:

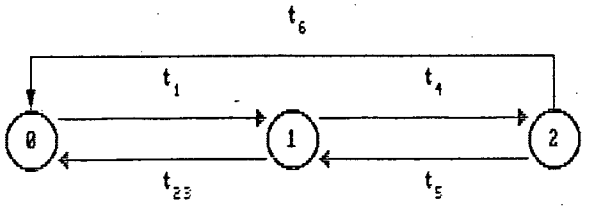

Fig. 4.12: Autômato do respondedor

Da verificação acima efetuada pode-se concluir que o modelo global do protocolo (modificado) executa corretamente o serviço conforme especificado. As modificações efetuadas no modelo global para sua correção foram baseadas no texto da norma [ISO 9506], o que indica inconsistências entre seu texto e as máquinas de estados que apresenta, usadas para a construção dos modelos iniciais das entidades.

4.1.4. Conclusões

Este exemplo buscou mostrar a utilização de ferramentas do ARP na verificação de uma parte do protocolo MMS, que trata dos serviços confirmados.

Demonstrou-se que a verificação de equivalência de linguagem é uma ferramenta importante, pois permite destacar o comportamento de cada entidade do comportamento global do modelo e compará-lo com a especificação. Desta forma garante-se a correção do modelo da implementação obtido a partir da descrição fornecida na norma.

Do exemplo pode-se também destacar uma metodologia para a abordagem da verificação de protocolos, que consiste das seguintes etapas:

- Modelagem de cada entidade do protocolo (incluindo os meios de comunicação), validando os modelos obtidos via grafo de acessibilidade e verificando a

conformidade dos grafos de estados gerados para cada entidade com o respectiva especificação.

- Construção, a partir dos modelos das entidades e do meio, de um modelo global do protocolo, utilizando por exemplo a fusão de transições como meio de interação entre os modelos.
- Validação do modelo global, detectando comportamentos anômalos, e modificações nos modelos das entidades, baseadas na análise do modelo global.
- Verificação do comportamento descrito pelo modelo global que representa o protocolo, do ponto de vista de sua conformidade com a especificação do mesmo serviço (visto da camada superior).

4.2. Uso do Ambiente ARP na Avaliação de Desempenho: Aplicação ao Serviço de Troca de Mensagens com Reconhecimento do Protocolo de Instrumentação Fabril FIP

Este exemplo busca demonstrar a potencialidade da ferramenta de avaliação de desempenho do ambiente ARP. Para tal será medida a eficiência do serviço de troca de mensagens com reconhecimento do protocolo de instrumentação fabril FIP [FIP 87], em um meio de comunicação com perdas.

Será estudada a eficiência do serviço provido pelo protocolo para vários níveis diferentes de perdas no meio, considerando também as diferentes disposições relativas entre as diversas estações do sistema.

4.2.1. Descrição da Aplicação

A comunicação de dados em controle digital de processos, nos níveis inferiores de uma planta de fabricação, evolui na direção do uso de redes locais interconectando os equipamentos, como sensores, atuadores, etc, através de barramentos seriais de alta velocidade (Field-Bus).

O protocolo FIP (Factory Instrumentation Protocol) é uma das propostas de Field-bus, hoje sendo analisadas pelos órgãos de normalização internacionais (ISA, IEC), para poder se tornar um padrão. Ele se baseia na difusão de dados entre estações produtoras e consumidoras destes, via rede de comunicação [FIP 87].

Existem no FIP três tipos de estações: o árbitro (único), que controla o acesso ao barramento das demais estações; a produtora, que difunde seus dados a pedido do árbitro e a consumidora, que recebe os dados difundidos.

Este exemplo visa avaliar a eficiência do serviço SDA (troca de mensagem com reconhecimento) do FIP considerando o meio de comunicação como tendo perdas físicas. A eficiência será medida em função do percentual de perdas efetivas de mensagem no meio e do número máximo de retransmissões permitidas pelo árbitro de barramento.

Serão apresentadas ao final curvas estimando as perdas médias finais (pedidos não atendidos) em função das perdas físicas e do número de retransmissões. Este estudo será realizado para os vários casos correspondentes às diferentes disposições relativas entre as estações (árbitro, produtor e consumidor).

a. Descrição do Serviço SDA do FIP

No FIP cada estação se comunica com as demais buscando valores de variáveis oriundas dos dispositivos monitorados ou controlados. Essa comunicação é coordenada pelo árbitro de barramento, que possui o seguinte ciclo global de funcionamento (macro-ciclo):

- Atualização cíclica (ou periódica) de variáveis.
- Atualização de variáveis requisitadas, de forma acíclica (ou aperiódica).
- Troca de mensagens requisitadas.

Cada etapa do macro-ciclo possui uma janela de tempo respectiva, com duração definida na configuração do sistema:

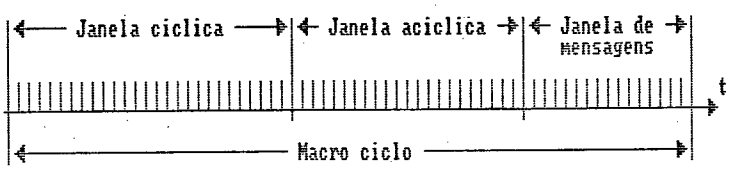

Fig. 4.13: Macro-ciclo do FIP

O serviço SDA (troca de mensagem com reconhecimento) inicia-se na janela de troca cíclica de dados, com um pedido de envio de mensagem pelo produtor da mesma. Este pedido é armazenado pelo árbitro e atendido posteriormente na janela de mensagens (fase MA) pelos seguintes passos:

- Envio do quadro *ID_MSG* pelo árbitro, identificando o produtor da mensagem.
- Recepção do *ID_MSG* pelo produtor e todos os consumidores possíveis.
- Envio do quadro *RP_MSG_ACK* pelo produtor, com a mensagem e seu destino.
- Recepção de *RP_MSG_ACK* pelo árbitro e pelos consumidores.
- Envio do quadro *RP_ACK* (reconhecimento) pelo consumidor destino.
- Recepção de *RP_ACK* pelo árbitro e pelo produtor da mensagem.

Caso o reconhecimento (*RP_ACK*) não chegue ao árbitro, por um motivo qualquer, este retransmite o mesmo *ID_MSG* para repetir todas as etapas acima, na tentativa de

assegurar a troca de mensagem. O número máximo de retransmissões n_{max} não é fixado na norma, devendo ser escolhido pelo projetista.

O comportamento das estações (árbitro, consumidor, produtor) é descrito pelos autômatos de estado apresentados na norma [FIP 87] e na fig 4.14:

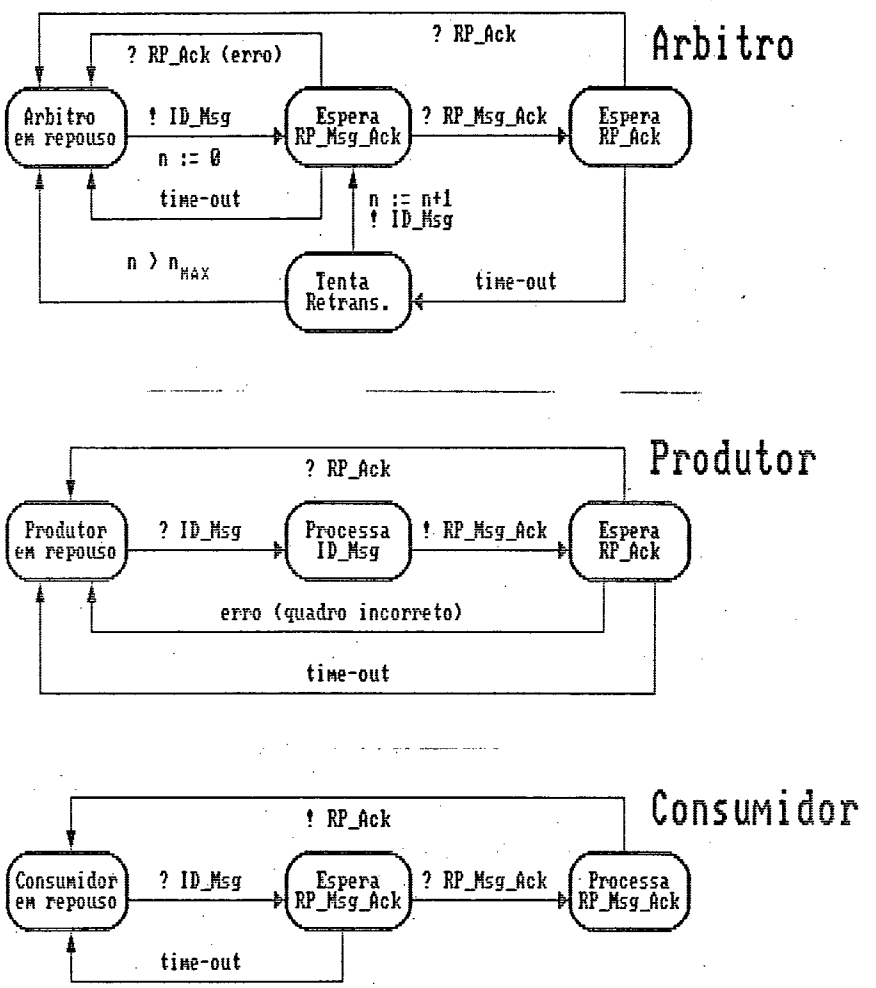

Fig. 4.14: Comportamento das estações FIP

Podem ocorrer três disposições distintas entre as estações no barramento, caso seja considerado somente um produtor e um consumidor, que é o caso deste exemplo:

Fig. 4.15: Disposições entre as estações do FIP

b. Definição dos Time-Outs

Abaixo definem-se os time-outs relevantes a este estudo. T_{XY} é o atraso de propagação entre as estações X e Y, e T_{ret} o tempo de resposta de cada estação.

T_{0t}: permite ao árbitro a detecção da não recepção de uma resposta:

$$T_{0t} > T_{AP} + T_{ret} + T_{PA} \Rightarrow T_{0t} > T_{ret} + 2 * T_{AP}$$

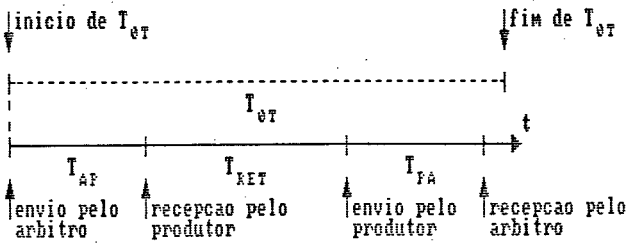

Fig. 4.16: Cálculo de T_{0t}

T_1: permite à estação produtora-consumidora a detecção da não recepção de uma resposta:

$$T_1 + T_{AC} > T_{AP} + T_{ret} + T_{PC} \Rightarrow T_1 > T_{ret} + T_{AP} + T_{PC} - T_{AC}$$

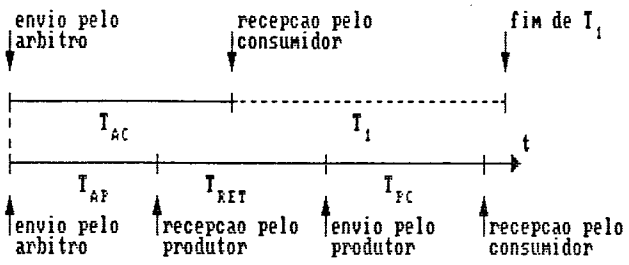

Fig. 4.17: Cálculo de T_1

Para este exemplo, arbitra-se T_{ret} = 15 uS e os seguintes atrasos de propagação, para cada disposição, obtendo-se os respectivos time-outs (valores em uS):

APC: T_{AP}= 2 T_{PC}= 3 T_{AC}= 5 -> T_{0t}> 19 T_1> 15

ACP: T_{AC}= 2 T_{PC}= 3 T_{AP}= 5 -> T_{0t}> 25 T_1> 21

PAC: T_{AP}= 2 T_{AC}= 3 T_{PC}= 5 -> T_{0t}> 19 T_1> 19

Adotar-se-á T_{0t}: 30 uS e T_1: 25 uS, para satisfazer as condições das três disposições.

4.2.2. Modelagem da Fase MA do Serviço SDA

Serão modeladas isoladamente as estações participantes e o meio de comunicação que as interliga. Considerar-se-á somente o atendimento a um pedido de mensagem, para simplificação do modelo e da avaliação.

Ao final será construído um modelo global para cada disposição entre estações, pela interligação de forma adequada entre os modelos das estações e os meios de comunicação.

a. Árbitro de Barramento

O modelo proposto para o árbitro, a ser utilizado neste estudo, inclui o mecanismo normal de transmissão e recepção de quadros, um mecanismo de retransmissão, time-outs e detecção de recepção não especificada (erro grave). O valor "n_{max}" define o número máximo de retransmissões permitidas. O estado "final" indica o fim das transmissões e retransmissões possíveis dentro do mesmo *ID_MSG*:

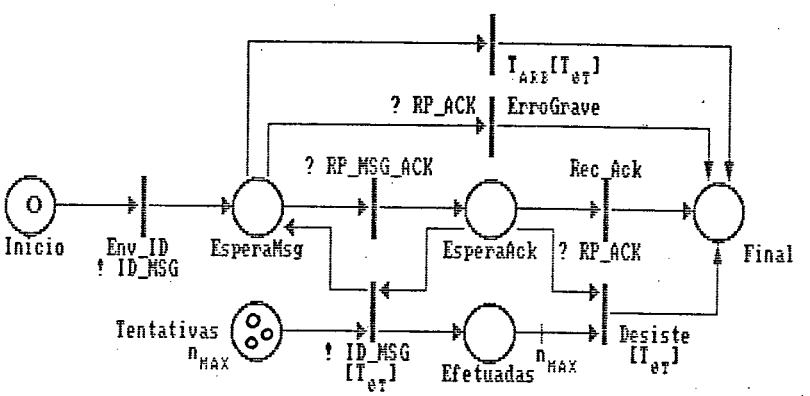

Fig. 4.18: Modelo do árbitro em RdPT

b. Produtor / Consumidor

Os modelos do produtor e do consumidor são apresentados nas figs. 4.19 e 4.20, respectivamente:

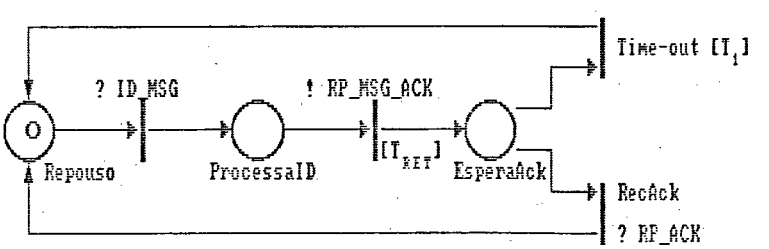

Fig. 4.19: Modelo do produtor em RdPT

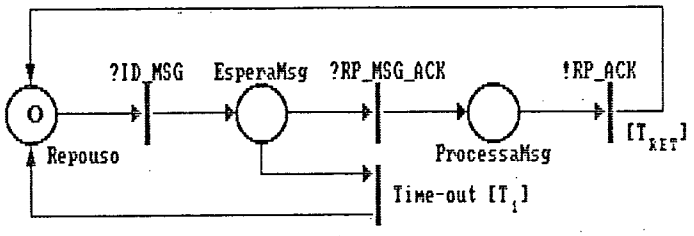

Fig. 4.20: Modelo do consumidor em RdPT

c. Meio de Comunicação

O meio é o responsável pela interação entre as estações do sistema, no qual circulam os quadros que definem o serviço.

Como serão respeitados os time-outs definidos para cada estação, e será modelada somente o atendimento a um único pedido de mensagem, não há necessidade de representar o controle de fluxo e a detecção de colisão nesta modelagem dos meios de comunicação. Somente será modelada a perda de quadro, de interesse para este exemplo:

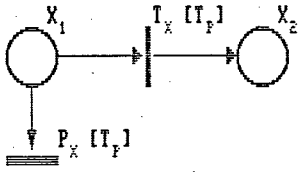

Fig. 4.21: Modelo do meio de comunicação

T_X representa a transferência de informação, e P_X sua perda. O modelo acima deve ser repetido para cada quadro circulante entre as estações. O acoplamento entre o meio e as estações é efetuado da seguinte forma:

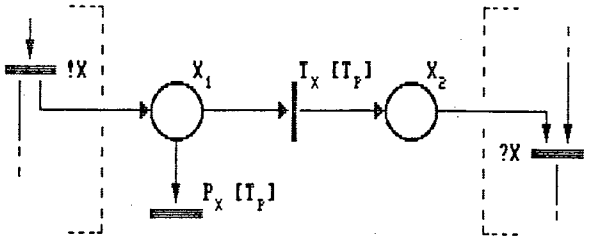

Fig. 4.22: Acoplamento entre estações

4.2.3. Avaliação de Desempenho do Serviço

Para a consideração dos percentuais de perdas no meio de comunicação, atribuir-se-á a cada conflito P_X-T_X uma probabilidade de disparo (P_f) para P_X variando entre 0 e 10%, em relação a T_X. O número máximo de retransmissões (n_{max}) será considerado entre 0 e 2.

Como evento inicial de cada avaliação será definida a transição *EnvID* do árbitro, e como destino único a transição *RecAck* do produtor, indicando o sucesso da execução do serviço. Ciclos de avaliação improdutivos (não alcançando este destino) serão considerados execuções fracassadas.

As avaliações serão efetuadas medindo o percentual de ciclos improdutivos (perdas médias finais, P_{med}) para cada combinação P_f versus n_{max} e para cada disposição relativa entre estações.

As curvas resultantes das avaliações estão apresentadas nos gráficos a seguir (figs 4.23 a 4.26). O último gráfico (fig 4.26) apresenta uma média dos valores obtidos para cada disposição entre estações, indicando o desempenho médio do protocolo. Para cada avaliação (ponto do gráfico) foram realizados cerca de 1000 ciclos de avaliação.

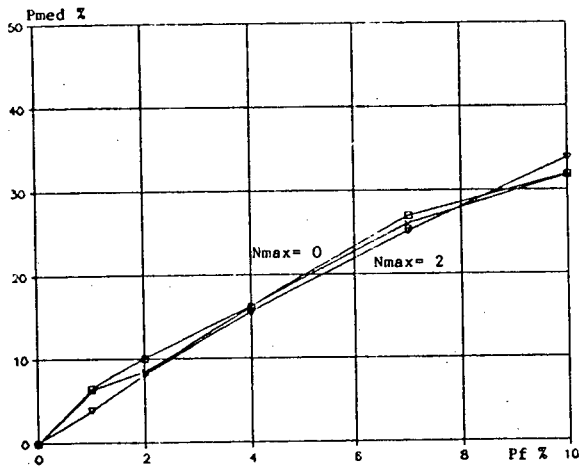

Fig. 4.23: Disposição ACP

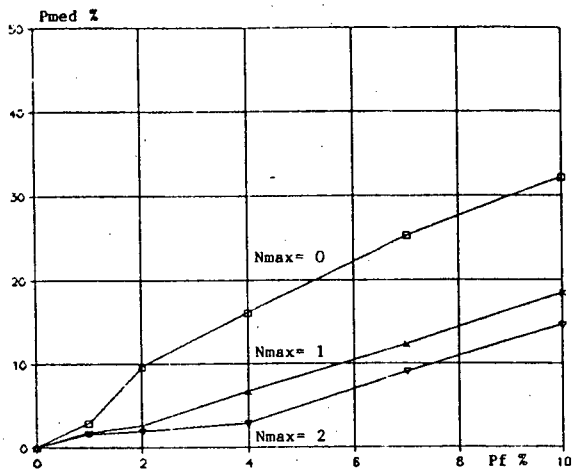

Fig. 4.24: Disposição APC

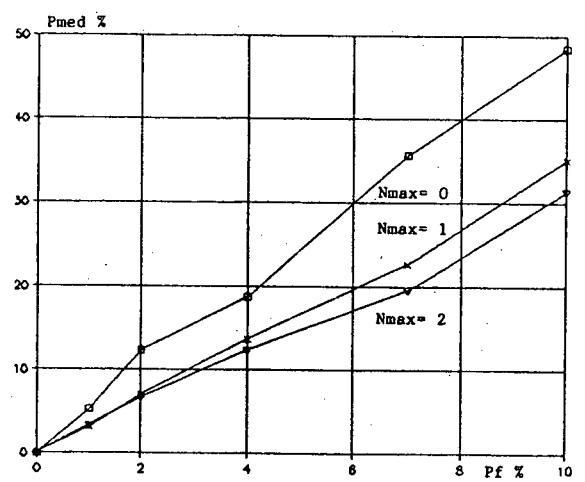

Fig. 4.25: Disposição PAC

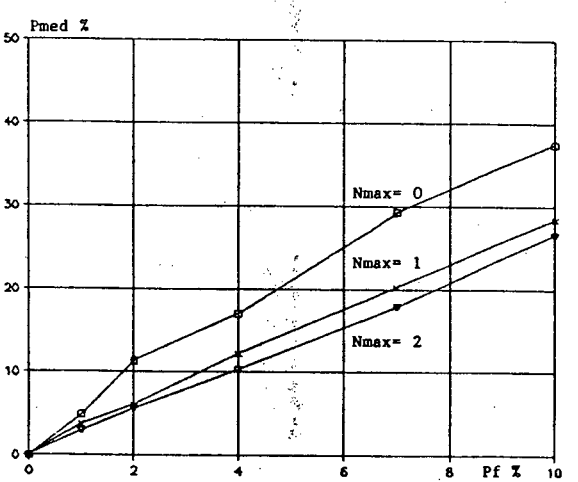

Fig. 4.24: Valores Médios

4.2.4. Conclusões

Em todas as curvas obtidas para o exemplo estudado observa-se um percentual de perdas finais P_{med} maior que o de perdas físicas P_f, mesmo utilizando-se de retransmissões. Isto se deve em parte ao grande número de quadros que circulam para a execução do serviço.

Ao contrário das demais disposições, observa-se uma baixa eficiência do mecanismo de retransmissão na disposição ACP (fig. 4.23). Sua maior eficiência é observada na disposição APC (fig. 4.24), que é a menos afetada pelas perdas físicas do meio. Esse comportamento deve-se aos trajetos a serem realizados pelas diversas PDUs do protocolo.

Um estudo mais profundo sobre o desempenho do FIP (e de outras propostas ao padrão Field-Bus), também utilizando o Sistema ARP, pode ser encontrado em [Aguiar 89].

O módulo de avaliação de desempenho do ARP mostra-se útil no levantamento de curvas de desempenho, que podem ser de difícil obtenção analítica, conforme a complexidade do sistema modelado.

É importante a correta fixação dos estados ou eventos que irão delimitar a avaliação (origem e destinos), que devem identificar completamente o comportamento em estudo. Para a determinação de tais estados ou eventos pode ser utilizada a análise via grafo de acessibilidade ou a simulação.

4.3. Uso do Ambiente ARP na Análise Estrutural de Redes de Petri: Aplicação à Otimização de Sistemas de Manufatura Flexível

Este exemplo visa demonstrar a capacidade do Ambiente ARP para a análise estrutural de Redes de Petri. Para tal será estudado o problema de otimização de um sistema flexível de manufatura com seqüências de fabricação pré-determinadas.

A otimização consistirá na maximização da velocidade de processamento do sistema, com minimização do estoque sob processo. A análise estrutural de Redes de Petri será empregada na validação do modelo obtido para o sistema e na determinação dos ciclos de funcionamento do mesmo, informação necessária ao método de otimização proposto.

4.3.1. Descrição da Aplicação

Um sistema de manufatura flexível (FMS) consiste de um conjunto de máquinas que processam, transportam e medem peças de diferentes tipos, como robôs, máquinas CNC, esteiras, etc.

Em geral, em um FMS as peças são processadas seguindo uma seqüência pré-determinada. Cada peça tem etapas de processamento em várias máquinas, que podem receber distintas peças para processar. Um FMS é dito de alimentação periódica (SMAP) quando uma seqüência fixa de peças o alimenta, ciclicamente [Cury 90].

O objetivo deste exemplo é otimizar o comportamento em regime permanente de um FMS distribuindo pallets no mesmo, forçando a plena utilização de uma máquina "gargalo" (aquela que limita a velocidade de produção, por ser mais lenta ou mais atarefada). O método de otimização aqui descrito é apresentado com detalhes em [Hillion 89].

Satisfazendo-se essa condição obtém-se a máxima taxa de produtividade do sistema. Deve-se considerar ainda a redução do número de pallets envolvidos no processo, minimizando o estoque em processamento.

4.3.2. Modelagem de um SMAP

O sistema a ser estudado (extraído de [Cury 90]) é formado por 2 máquinas (m_1,m_2) que processam 3 tipos de peças (p_1,p_2,p_3). Cada peça e seu processo de fabricação (seqüência de máquinas a utilizar, com respectivos tempos de processamento) definem uma tarefa:

$$p_1: (m_1/3, m_2/5) \quad p_2: (m_2/6, m_1/4) \quad p_3: (m_1/7, m_2/3)$$

A seqüência de entrada de peças no sistema é (p_1,p_2,p_3) e a ordem de seu processamento pelas máquinas é m_1: (p_1,p_2,p_3) e m_2: (p_1,p_2,p_3).

Modela-se o processo de fabricação de uma peça pelo circuito dado na fig. 4.27. O modelo considera que, ao término do processamento de uma peça, o respectivo pallet é imediatamente liberado para o processamento de outra peça.

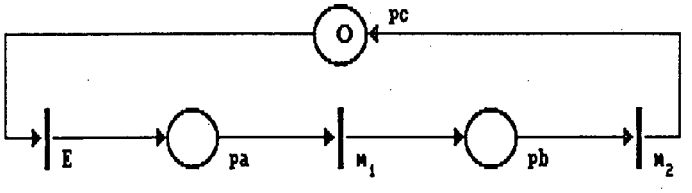

Fig. 4.27: Processo de fabricação de uma peça

As transições m_i indicam o processamento da peça pela máquina i; a transição E indica a entrada de uma nova peça no sistema; os lugares pa, pb e pc indicam a situação da peça no sistema:

pa: peça antes de ser processada.

pb: peça após primeiro processamento.

pc: pallet pronto para novo processamento (peça pronta).

Para simplificar a representação, cada transição m_i do modelo acima abstrai uma estrutura como a definida na fig. 4.28, que representa o processamento de uma peça por vez em cada máquina, com duração T:

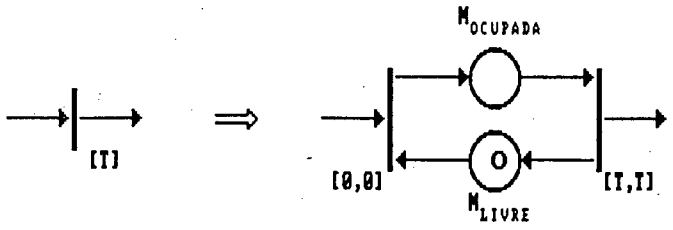

Fig. 4.28: Representação interna de uma máquina

Interligando os processos referentes às peças através dos circuitos das máquinas obtém-se o modelo do sistema acima descrito, na fig. 4.29:

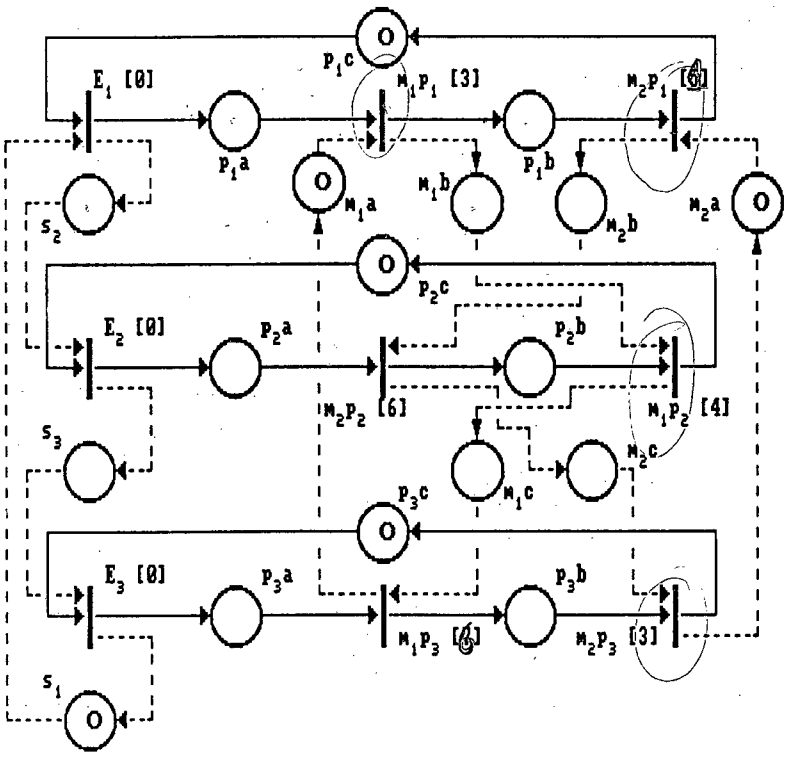

Fig. 4.29: Modelo global do sistema

No modelo da fig. 4.29, uma transição $m_i p_j$ indica a peça p_j sendo processada pela máquina m_i. As transições E_i indicam o lançamento periódico e seqüenciado de peças. Os

traços contínuos representam os processos de produção de cada peça, e os pontilhados o seqüenciamento de comando das máquinas ou de alimentação de peças.

Os lugares $m_i a$, $m_i b$, $m_i c$ e s_i indicam o seqüenciamento de operação das máquinas e de alimentação de peças.

4.3.3. Análise do Modelo Obtido

A análise do modelo visa inicialmente provar suas propriedades básicas a partir da análise estrutural, pois a marcação inicial do mesmo é o objeto da otimização, estando portanto indefinida.

O processo de otimização proposto a seguir está calcado sobre o estudo das máquinas de estados elementares que compõe a estrutura da rede.

a. Cálculo dos Invariantes

Através do módulo de análise estrutural do ARP, verifica-se que o modelo do sistema possui somente um invariante de transição, contendo todas as transições da rede (portanto a rede é potencialmente viva), e que esse invariante é um grafo de eventos.

São obtidos também 30 invariantes de lugar, todos eles sendo máquinas de estados. Todos os lugares da rede pertencem a algum invariante, indicando a limitação da rede.

A lista dos invariantes de lugar obtidos é dada a seguir, classificados de acordo com sua interpretação física:

Invariantes de comando das máquinas:

1: $\{m_2 a, m_2 b, m_2 c\}$

2: $\{m_1 a, m_1 b, m_1 c\}$

Invariantes das peças:

3: {p_1a, p_1b, p_1c}

4: {p_2a, p_2b, p_2c}

5: {p_3a, p_3b, p_3c}

Invariante de alimentação de peças:

6: {s_1, s_2, s_3}

Invariantes mistos (combinações dos anteriores):

7: {m_1a, m_1c, m_2b, p_1b, p_2b}

8: {m_1a, m_1b, m_2c, p_2a, p_2c, p_3a, p_3c}

9: {m_1a, m_1b, p_2c, p_3a, s_3}

10: {m_1a, m_1c, p_1b, p_1c, p_2a, p_2b, s_2}

11: {m_1a, m_2b, m_2c, p_1b, p_3a, p_3c}

12: {m_1a, m_2b, p_1b, p_2b, p_2c, p_3a, s_3}

13: {m_1a, m_2c, p_1b, p_1c, p_2a, p_3a, p_3c, s_2}

14: {m_1a, p_1b, p_1c, p_3a, s_2, s_3}

15: {m_1b, m_1c, p_1a, p_3b, p_3c, s_1}

16: {m_1b, m_1c, m_2a, p_1a, p_1c, p_3b}

17: {m_1b, m_2a, m_2c, p_1a, p_1c, p_2a, p_2c}

18: {m_1b, m_2a, p_1a, p_1c, p_2c, p_3a, p_3b, s_3}

19: {m_1b, m_2c, p_1a, p_2a, p_2c, p_3c, s_1}

20: {m_1b, p_1a, p_2c, s_1, s_3}

21: {m_1c, m_2a, m_2b, p_2b, p_3b}

22: {m_1c, m_2a, p_1c, p_2a, p_2b, p_3b, s_2}

23: {m_1c, m_2b, p_1a, p_1b, p_2b, p_3b, p_3c, s_1}

24: {m_1c, p_2a, p_2b, p_3b, p_3c, s_1, s_2}

25: {m_2a, m_2b, p_2b, p_2c, p_3a, p_3b, s_3}

26: {m_2a, m_2c, p_1c, p_2a, s_2}

27: $\{m_2a, p_1c, p_3a, p_3b, s_2, s_3\}$

28: $\{m_2b, m_2c, p_1a, p_1b, p_3c, s_1\}$

29: $\{m_2b, p_1a, p_1b, p_2b, p_2c, s_1, s_3\}$

30: $\{m_2c, p_2a, p_3c, s_1, s_2\}$

b. Otimização do Sistema

O processo de otimização empregado neste exemplo é proposto em [Hillion 89], e baseia-se no estudo das durações (tempos de ciclo) associadas aos circuitos (máquinas de estados) da rede que modela o sistema. Essas durações são função da marcação do circuito e das durações das transições que o compõe.

O circuito com o máximo tempo de ciclo é denominado "circuito crítico", e deve corresponder ao da máquina gargalo, para a máxima taxa de produtividade em regime permanente. O processo de otimização consiste portanto em distribuir fichas nos circuitos da rede (na mínima quantidade possível) para tornar seus tempos de ciclo inferiores ao da máquina gargalo.

Para tal é construído um programa linear cujas restrições são formadas a partir dos circuitos obtidos, e cuja função a minimizar é o número total de peças em processamento. Aplicando-se tal método ao exemplo em estudo é obtida a seguinte marcação:

$$M: \{m_1a, m_2c, p_1c, p_2b, p_3b, s_1\}$$

O número de peças em processamento é o mínimo: 3 (p_1c, p_2b, p_3b). O tempo de ciclo do sistema também corresponde ao mínimo (14, tempo de ciclo de ambas as máquinas), e pode ser constatado no grafo de estados acessíveis obtido pelo ARP, para essa marcação inicial:

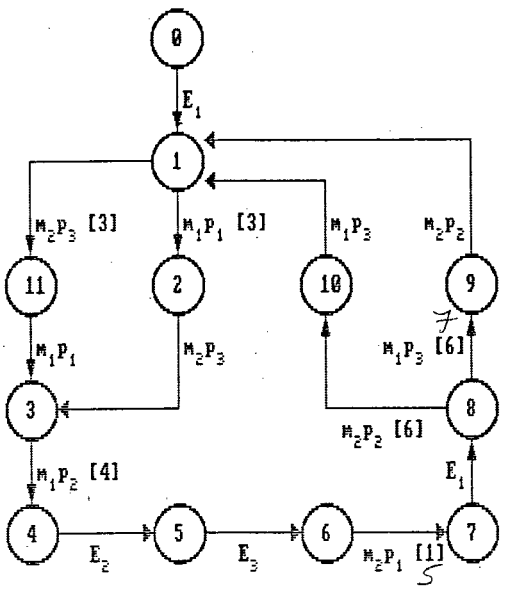

Fig. 4.30: Grafo de estados do sistema

É interessante adotar como marcação inicial do sistema não a inicialmente obtida, mas uma de suas sucessoras, por não apresentarem o transitório de inicio de operação.

4.3.4. Conclusões

Apresenta-se neste exemplo a utilização do Ambiente ARP na determinação de circuitos de um Sistema Flexível de Manufatura, como parte de um processo de otimização do funcionamento do mesmo. Também é importante ressaltar a construção do grafo de acessibilidade, na validação da marcação inicial obtida e na minimização do transitório de inicio de operação do sistema.

Embora este exemplo esteja voltado a um assunto bastante específico, deve-se ressaltar a importância da análise estutural como auxílio à validação de Redes de Petri, bem como para o estudo de características específicas de seu comportamento, como ciclos de funcionamento e componentes conservativas.

4.4. Conclusão

Este capítulo buscou demonstrar a aplicabilidade das ferramentas do Sistema ARP ao estudo de problemas de real interesse, nas áreas de Automação Industrial e Sistemas Informáticos Distribuídos.

O desenvolvimento de cada exemplo buscou delinear uma metodologia de estudo de problemas usando Redes de Petri, desde a modelagem do objeto de estudo até a interpretação dos resultados obtidos pelas diversas ferramentas do ARP.

Deve-se observar que a utilização do ambiente ARP não se restringe às áreas acima citadas, visto que os modelos em estudo são tratados de forma genérica.

A integração existente entre as ferramentas do ambiente ARP permite a utilização conjunta de várias ferramentas em um mesmo problema, de um modo simples. Isto foi efetuado em todos os exemplos estudados.

5. Conclusão e Perspectivas

Neste trabalho procurou-se conceber e implementar um ambiente orientado para a modelagem, análise do comportamento e do desempenho e simulação de sistemas modelados por Redes de Petri ordinária, com Temporização e com Temporização Extendida.

Inicialmente o modelo Rede de Petri foi formalmente descrito, bem como aquelas extensões a serem utilizadas no ambiente. Discutiu-se a metodologia de concepção de sistemas utilizando Redes de Petri.

Apresentou-se o Ambiente ARP, destacando para cada ferramenta implementada seus objetivos, implementações e resultados. Além disso, buscou-se através de exemplos demonstrar a potencialidade e praticidade de uso de cada ferramenta.

Como principais perspectivas a curto e médio prazo para o presente trabalho, visando sanar deficiências ou aumentar o poder e flexibilidade do sistema, podem ser indicadas as seguintes:

- Utilização de uma linguagem de descrição de Redes de Petri que permita explorar o conceito de modularidade na modelagem de sistemas, abordado no cap. 2. Tal linguagem tem características similares à descrita em [Cantú 90].

- Introdução das relações G_{jk} entre transições t_j e t_k sensibilizadas em paralelo, na enumeração de classes de estados de Redes de Petri com Temporização.

- Introdução, no módulo de verificação, da determinação de novas classes de equivalência (forte e observacional).

- Utilização do ARP como parte de um ambiente mais complexo, como o descrito em [Farines 89], que permita tratar problemas mais complexos, descritos em Redes de Petri de alto nível (RdP a Objeto). A técnica de projeção [Cantú 90] permite a extração de Redes de Petri ordinárias da RdP a Objeto em estudo, que podem então ser analisadas no ARP. Outrossim, alguns módulos do ARP são independentes do modelo inicial, como o de verificação de equivalência, e podem ser empregadas sem modificações em outros ambientes.

- Migração do sistema para um equipamento mais poderoso, que permitirá agilizar a análise de sistemas complexos.

Anexo1: Linguagem de Descrição de Redes de Petri

A linguagem de descrição usada no ARP possui uma sintaxe semelhante à do Pascal, permitindo a identificação de lugares e transições por nomes quaisquer. Os caracteres aceitos são:

A..Z a..z 0..9 _ ! ? + - # @ ^ %

A estrutura de uma descrição de rede é a seguinte:

REDE *nome_da_rede* ;

declaração de constantes

declaração dos nodos (lugares e transições)

declaração da estrutura (arcos)

FIM.

Podem ser definidas constantes da seguinte forma:

CONST

Num_Vagas = 7 ;

Num_Envios = 10 ;

Time_Out = 45 ;

A declaração **NODOS** é dividida em duas partes: a declaração de transições e a de lugares (máximo de 150 nodos de cada tipo). A declaração de lugares faz-se da seguinte forma (o valor entre parênteses indica a marcação inicial dos lugares declarados, c/ default 0 e máximo 254):

Robot_Parado,Torno_Operando : **LUGAR** ;

Robot_Ligado,Torno_em_Espera : **LUGAR** (1) ;

Pecas_Esperando : **LUGAR** (Num_Vagas) ;

A declaração de transições é de forma similar. No caso de redes com temporização segundo o modelo de Merlin, podem ser definidos o "Static Earliest Fire Time" (SEFT) e o "Static Latest Fire Time" (SLFT), com default em [0,0]. Também pode ser definida a curva de densidade de probabilidade associada a esse intervalo, por default uniforme:

Robot_Solta_Peca : **TRANSICAO** ;

Inicia_Usinagem : **TRANSICAO** [5,Time_Out] ;

Chegada_Peca : **TRANSICAO** [0,5] **EXPON** (10) ;

Robot_Pega_Peca : **TRANSICAO** [2,8] **NORMAL** (5) ;

Nos intervalos são aceitos inteiros entre 0 e 32000, podendo ser constantes pré-definidas. As curvas de densidade de probabilidade (usadas pelo módulo de avaliação de desempenho) aceitas são:

- Uniforme, por default.
- Normal, com média no centro do intervalo e forma definida pela razão k entre o valor da curva no centro e nos extremos do intervalo ($1 \leq k \leq 10000$).
- Exponencial, com forma definida pela razão k entre o valor da curva no início e no final do intervalo ($0.0001 \leq k \leq 10000$).

Observa-se que a definição das curvas não depende de valores absolutos mas apenas da razão entre os valores máximo e mínimo.

Através da declaração **ESTRUTURA** são descritos os arcos que ligam as transições aos lugares e vice-versa. Sua forma geral é:

Transição: (*lugares de entrada*) , (*lugares de saida*) ;

Essa estrutura deve ser repetida para cada transição da rede. Os nomes dos lugares são separados por vírgulas. No caso de arco com peso não unitário temos:

Transicao: (... ,*peso* * *Lugar*, ...) , (...) ;

onde o peso pode ser um valor numérico ou uma constante pré-definida, entre 0 e 254.

O exemplo a seguir (fig.A1.1) ilustra a linguagem. A rede modela um sistema composto de um torno e um robô que o alimenta, carregando e descarregando peças de um armazém:

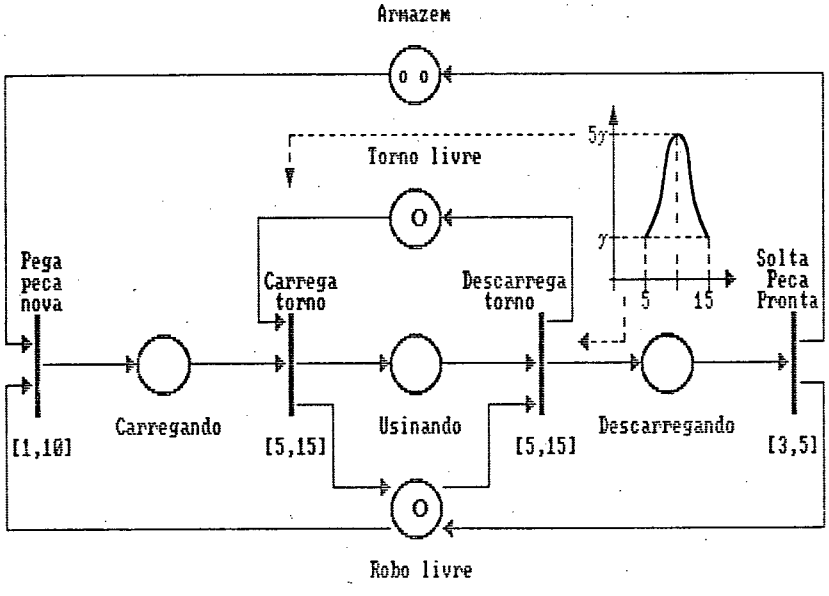

Fig. A1.1: Rede de Petri

A descrição da rede da fig A1.1 através da linguagem de descrição de redes acima apresentada, é indicada na listagem a seguir:

Rede Exemplo_da_Linguagem ;

Const

Pecas_Entrada = 2 ; { peças na entrada da máquina }

Nodos

{ armazem de pecas }

Armazem : **Lugar** (Pecas_Entrada) ;

{ nodos do robo que alimenta o torno }

Robo_Livre : **Lugar** (1) ;

Robo_Carregando,

Robo_Descarregando : **Lugar** ;

Pega_Peca_Nova : **Transicao** [1,10] ;

Solta_Peca_Pronta : **Transicao** [3,5] ;

Carrega_Torno,

Descarrega_Torno : **Transicao** [5,15] **Normal** (5) ;

{ nodos do torno }

Usinando : **Lugar** ;

Torno_Livre : **Lugar** (1) ;

Estrutura { controle do robo }

Pega_Peca_Nova : (Armazem,Robo_Livre), (Robo_Carregando) ;

Carrega_Torno : (Robo_Carregando,Torno_Livre), (Usinando,Robo_Livre) ;

Descarrega_Torno : (Usinando,Robo_Livre),
(Robo_Descarregando,Torno_Livre) ;

Solta_Peca_Pronta : (Robo_Descarregando), (Armazem,Robo_Livre) ;

Fim.

Bibliografia

[Agerwala 79] T. Agerwala: "**Putting Petri Nets to Work**". Computer, dec. 1979 pags 85-94. IEEE, Inc.

[Aguiar 89] M.C. Aguiar: "**Análise Comparativa de Desempenho da Camada Enlace de Dados do ProfiBus e do FIP, Propostas Candidatas ao Padrão FieldBus**". Dissertação de Mestrado, EEL-UFSC, 1989.

[Aho 73] A. Aho, J. Hopcroft, J. Ullman: "**The Design and Analysis of Computer Algorithms**". Addison-Wesley, 1973.

[Aho 86] A. Aho, R. Sethi, J. Ullman: "**Compilers: Principles, Techniques and Tools**". Addison-Wesley, 1986.

[Ayache 85] J.M. Ayache, J.P. Courtiat, M.Diaz, G.Juanole: "**Utilisation de Réseaux de Petri pour la modélisation et la validation des Protocoles**". Technique et Science Informatiques, vol. 4, nº 1, 1985.

[Azema] P. Azema, J.C. Lloret, G. Papapanagiotakis, F. Vernadat: "**ESTELLE Validation and PROLOG Interpreted Petri Nets**". The Formal Description Technique Estelle, Results of the ESPRIT/SEDOS Project. North-Holland, 1989.

[Azema 85] P. Azema, F. Vernadat: "**Requirement Analysis for Communication Protocols**".

[Berthelot 82] G. Berthelot, R. Terrat: "**Petri Nets Theory for the Correctness of Protocols**". IEEE Transactions on Communications, vol. Com-30, nº 12, December 1982.

[Berthomieu 79] B. Berthomieu: "**Analyse Structurelle des Réseaux de Petri: Méthodes et Outils**". Thèse de Doctorat. Université Paul Sabatier de Toulouse, France, 1979.

[Billington] J. Billington: "**Protocol Engineering and Nets**". ...

[Billington 88] J. Billington, G.R. Wheeler, M.C. Wilbur-Ham: "**Protean: A High-Level Petri Net Tool for the Specification and Verification on Communications Protocols**". IEEE Transactions of Software Engineering, march 1988, vol.14, n.3.

[Bolognesi 87a] T. Bolognesi, E. Brinksma: "**Introduction to the ISO Specification Language LOTOS**". Computer Network and ISDN Systems 14, North-Holland, 1987.

[Bolognesi 87b] T. Bolognesi, S.A. Smolka: "**Fundamental Results for the Verification of Observational Equivalence: a Survey**", VII Conference on Protocol Specification, Testing and Verification, pgs. 165-179, may 1987.

[Bougé 88] L. Bougé: "**Sémantique du Parallélisme: un Tour d'Horizon**", rapport LIENS nº 88-6, École Normale Supérieure, Paris, France, juil. 1988.

[Brams 83] G.W. Brams: "**Réseaux de Petri : Théorie et Pratique (tomes 1 et 2)**". Ed. Masson 1983, France.

[Brauer 86] W. Brauer, W. Reisig, G. Rozenberg: "**Advances in Petri Nets 1986, Part I e II - Proceedings of an Advanced Course**", Ed. Springer-Verlag - Lectures Notes in Computer Science, No.255, Sept 1986.

[Cantú 90] E. Cantú: "**Uma Abordagem para a Representação, Simulação e Implementação de Sistemas Baseada na Rede de Petri a Objetos**". Dissertação de Mestrado DEEL-UFSC, março 1990.

[Colom 89] J.M. Colom, M. Silva: "**Convex Geometry and Semi-Flows in P/T Nets. A Comparative Study of Algorithms for Computation of Minimal P-Semi-Flows**". 10th International Conference on Application and Theory of Petri Nets, Bonn - Germany, june 1989.

[Courtiat 84] J.P. Courtiat, J.M. Ayache, B. Algayres: "**Petri Nets are Good for Protocols**". Computer Communications Review, 14, nº 2, 1984.

[Courtiat 87] J.P. Courtiat: "**Contribution à la Description Formèlle de Protocoles**". Thèse de Doctorat d'État, Université Paul Sabatier de Toulouse, France, 1987.

[Courvoisier 80] M. Courvoisier, R. Valette: "**Systèmes de Commande en Temps Réel. Description, Analyse et Realisation**". Éditions SCM, 1980. Paris, France.

[Cury 90] J.E. Cury: **"Lançamento e Escalonamento de Tarefas em Sistemas de Manufatura com Alimentação Periódica"**. Submetido ao 8º Congresso da Sociedade Brasileira de Automática, Belém-PA, 1990.

[Diaz] M. Diaz: **"Aplying Petri Net Based Models in the Design of Systems"**. ...

[Diaz 82] M. Diaz: **"Modeling and Analysis of Communication and Cooperation Protocols Using Petri Nets Based Models"**, Proceedings of IFIP Protocol Specification, Testing and Verification, may 1982.

[Diaz 86] M. Diaz: **"Petri Net Based Models in the Specification and Verification of Protocols"**, Lecture Notes in Computer Science 254: Advance in Petri Nets, Part I, Ed. Springer-Verlag, sept. 1986.

[Dufau 84] J. Dufau: **"Un Outil pour la Verification des Protocoles Décrits par Réseaux de Petri"**. Thèse de Doctorat, Université Paul Sabatier de Toulouse, France, 1984.

[Esteban 85] P. Esteban:**"Sur la Recherche d'Algorithmes Simplifiés d'Analyse des Réseaux de Petri"**. Thèse de Doctorat. LAAS Toulouse, France, 1985.

[Esteban 86] P. Esteban, R. Valette, M. Courvoisier: **"Simplified Algorithms for Petri Nets Analysis"**, Proceedings of IECON'86.

[Farines 89] J.M. Farines, E. Cantú, H. Garnousset, C.A. Maziero: **"ARP: uma Ferramenta para o Desenvolvimento de Software em Aplicações Distribuídas"**. Seminário Franco-Brasileiro em Sistemas Informáticos Distribuídos, Fpolis-SC, 1989.

[FIP 87] **"FIP - a Standard Proposal for FieldBus"**. Club FIP ENSEM/Nancy, France, 1987.

[Freedman 88] P. Freedman, A. Malowany: **"The Analysis and Optimization of Repetition within Robot Workcell Sequencing Problems"**. 1988 IEEE International Conference on Robotics and Automation, Philadelphia - Pennsylvania - USA.

[Freedman 89] P. Freedman, A. Malowany: "**Petri Nets and the Modelling of Repetitive Events**". Research Report TR-CIM-88-5, march 88, McGill Research Centre for Intelligent Machines, McGill University, Montréal, Québec, Canada.

[Hillion 89] H.P. Hillion, J.M. Proth: "**Performance Evaluation of Job-Shop Systems Using Timed Event-Graphs**". IEEE Transactions on Automatic Control, vol AC-34, n^{o} 1, 1989.

[Hopcroft 79] J. Hopcroft, J. Ullman: "**Introduction to Automata Theory, Languages and Computation**". Addison-Wesley, 1979.

[Huber 89] P. Huber, K. Jensen, R. Shapiro: "**Hierarchies in Coloured Petri Nets**". 10th International Conference on Application and Theory of Petri Nets, Bonn - Germany, june 1989.

[ISO 9074] "**Estelle, a Formal Description Technique Based on an Extended State Transition Model**", ISO-DIS 9074, jun. 1987.

[ISO 8807] "**LOTOS - a Formal Description Technique Based on the Temporal Ordering of Observational Behaviour**", ISO-DIS 8807, jul. 1987.

[ISO 9506] ISO/DIS Draft International Standard 9506-2, 11/88, Manufacturing Message Specification (MMS).

[Jard] C. Jard: "**Introduction a la Vérification des Algorithmes Distribués**". ...

[Jensen 85] K. Jensen: "**An Introduction to High Level Petri Nets**". DAIMI PB-197, Computer Science Department, Aarhus University, Denmark, 1985.

[Jensen 86] K. Jensen: "**Computer Tools for Construction, Modification and Analysis of Petri Nets**". GMD Advanced Course on Petri Nets, Bad Honnef, sept. 1986.

[Juanole 89] G. Juanole, J.L. Roux: "**On the Pertinence of the Extended Time Petri Net Model for Analyzing Communication Activities**". 1989 ...

[Lloret 88] J.C. Lloret, P. Azema and F. Vernadat: "**Réseaux PrT Labellés: Structuration et Composition**", rapport n^{o} 88350 LAAS/CNRS, Toulouse, France, nov. 1988.

[Menasche 82] M. Menasche: "**Analyse des Réseaux de Petri Temporisés et Aplication aux Systèmes Distribués**". Thèse de Doctorat. Université Paul Sabatier de Toulouse, France, 1982.

[Menasche 83] M. Menasche, B. Berthomieu: "**Time Petri Nets for Analyzing and Verifying Time Dependent Communication Protocols**". Protocol Specification, Testing and Verification IFIP, North-Holland, 1983.

[Menasche 85] M. Menasche: "**Parede: an Automated Tool for the Analysis of Time(d) Petri Nets**". 1985 ...

[Merlin 76] P.M. Merlin, D.J. Farber: "**Recoverability of Communications Protocols**". IEEE Trans. on Communications, sept. 1976.

[Novali 86] J. Novali: "**Modeles d'Observation pour les Architectures Multicouches de Protocoles de Communication**". Thèse de Doctorat, Institut National des Sciences Apliquées de Toulouse, France, 1986.

[Peterson 81] J.L. Peterson: "**Petri Net Theory and the Modelling of Systems**". Prentice-Hall Editions, 1981.

[Pinho 88] A.N. Pinho: "**Um Estudo da Especificação de Mensagem da Manufatura (MMS) do Protocolo de Comunicação para Ambientes Industriais MAP**". Dissertação de Mestrado, EEL-UFSC, 1988.

[Pradin 79] B. Chezalviel-Pradin: "**Un Outil Graphique Interactif pour la Validation des Systèmes à Evolution Parallèle Décrits par Réseaux de Petri (Ogive)**". Thèse de Doctorat, Université Paul Sabatier, Toulouse, France, 1979.

[Roux 85] J.L. Roux: "**Modelisation et Analyse des Systémes Distribués par les Réseaux de Petri Temporels**". Thèse de Doctorat. Institut National des Sciences Apliquées de Toulouse, France, 1985.

[Roux 87] J.L. Roux, G. Juanole: "**Functional and Performance Analysis Using Extended Petri Nets**", International Workshop on Petri Nets and Performance Models, aug 1987.

[Schreiber 74] T.R. Schreiber: "**Simulation using GPSS**", Ed. Wiley, 1974.

[Sibertin 85] C. Sibertin-Blanc: "**High-level Petri Nets with Data Structures**", 6th European Workshop on Applications and Theory of Petri Nets, Helsinki, Finland, jun. 1985.

[Souissi 89] Y. Souissi, G. Memmi: "**Composition of Nets via a Communication Medium**". 10th International Conference on Application and Theory of Petri Nets, Bonn - Germany, june 1989.

[Tazza 88] M. Tazza: "**Análise Quantitativa de Sistemas**". III Escola Brasileiro-Argentina de Informática, Curitiba-PR, 1988.

[Valette 86] R. Valette: "**Nets in Production Systems**", Lecture Notes in Computer Science 255: Advance in Petri Nets, Part II, Ed. Springer-Verlag, pgs. 191-217, sept. 1986.

[Vissers 88] C.A. Vissers: "**FDTs for Distributed Systems**". Research into Networks and Distributed Applications, North-Holland, 1988.

[Wang 89a] F. Wang, K. Gildea, A. Rubenstein: "**A Note on the Confirmed Services in MMS**". ...

[Wang 89b] F. Wang, K. Gildea, A. Rubenstein: "**A Coloured Petri Net Model for Connection Management Services in MMS**". ...