FÁBIO DA COSTA SOUZA

**DESENVOLVIMENTO DE METODOLOGIA DE APLICAÇÃO DE
REDES DE PETRI PARA AUTOMAÇÃO DE SISTEMAS
INDUSTRIAIS COM CONTROLADORES LÓGICOS
PROGRAMÁVEIS (CLP)**

SÃO PAULO

2006

FÁBIO DA COSTA SOUZA

**DESENVOLVIMENTO DE METODOLOGIA DE APLICAÇÃO DE
REDES DE PETRI PARA AUTOMAÇÃO DE SISTEMAS
INDUSTRIAIS COM CONTROLADORES LÓGICOS
PROGRAMÁVEIS (CLP)**

Dissertação apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção de título
de Mestre em Engenharia Elétrica.

Área de Concentração: Energia e Automação (PEA)

Orientador: Prof. Dr. Sergio Luiz Pereira

SÃO PAULO

2006

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 21 de novembro de 2006.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Souza, Fábio da Costa

Desenvolvimento de metodologia de aplicação de redes de Petri para automação de sistemas industriais com controladores lógicos programáveis / F. da C. Souza. -- ed.rev. -- São Paulo, 2006.

146 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Energia e Automação Elétricas.

1.Redes de Petri 2.Controladores programáveis 3.Automação industrial I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Energia e Automação Elétricas II.t.

A Deus, que me iluminou e esteve comigo no decorrer deste percurso, me capacitando a continuar.

À Fernanda, minha esposa, com amor, admiração e gratidão por sua compreensão, carinho, presença e incansável apoio ao longo do período de elaboração deste trabalho.

À Nathália, minha filha, que este trabalho seja minha demonstração de que devemos sempre buscar nossos objetivos.

Aos meus pais que me deram a sabedoria necessária para estar sempre em busca dos meus desafios.

AGRADECIMENTOS

Ao Prof. Dr. Sergio Luiz Pereira, pela atenção e apoio durante o processo de definição e orientação desta dissertação de mestrado.

Ao convênio Rockwell e Escola Politécnica da USP, por ter colocado à disposição a área experimental e o laboratório, para elaboração do estudo de caso.

Ao departamento de energia e automação (PEA) da USP pela disponibilização de toda infra-estrutura e pessoal que ajudaram na realização deste trabalho.

À Banca examinadora de qualificação pelas orientações e sugestões que contribuíram para o desenvolvimento desta dissertação.

A todos que, direta ou indiretamente, colaboraram com a execução deste trabalho.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE GRÁFICOS

LISTAS DE ABREVIATURAS E SIGLAS

LISTA DE SÍMBOLOS MATEMÁTICOS

GLOSSÁRIO

RESUMO

ABSTRACT

1	INTRODUÇÃO	21
1.1	OBJETIVO	22
1.2	MOTIVAÇÃO	22
2	TEORIA GERAL DAS REDES DE PETRI PARA APLICAÇÃO NA ANÁLISE DE SISTEMAS DE AUTOMAÇÃO	26
2.1	INTRODUÇÃO	26
2.2	INTRODUÇÃO DA TEORIA DAS REDES DE PETRI	26
2.3	DEFINIÇÕES DE REDES DE PETRI	27
2.4	SÍMBOLOS USUAIS PARA REPRESENTAÇÃO GRÁFICA DAS RdP	29
2.5	EXECUÇÃO DAS REDES DE PETRI	31
2.6	PRINCIPAIS PROPRIEDADES DAS REDES DE PETRI	33
2.7	TÉCNICAS DE ANÁLISE DAS REDES DE PETRI	34
2.7.1	ANÁLISE PELAS ÁRVORES DE ALCANÇABILIDADE E DE COBERTURA	35
2.7.1.1	LIMITAÇÕES DA ÁRVORE DE COBERTURA NA ANÁLISE DAS REDES DE PETRI.	40
2.7.2	ANÁLISE DAS RdP PELA MATRIZ DE INCIDÊNCIA E EQUAÇÕES DE ESTADOS.	40
2.7.3	ANÁLISE DE INVARIANTES.	47
2.7.4	ANÁLISE DAS RdP POR SIMULAÇÃO COMPUTACIONAL	53
2.7.4.1	FERRAMENTA DE SIMULAÇÃO VISUAL OBJECJ NET ++ (VON)	53

3	PROGRAMAÇÃO E UTILIZAÇÃO DE CONTROLADORES LÓGICOS PROGRAMÁVEIS-----	57
3.1	CONTROLADOR LÓGICO PROGRAMÁVEL (CLP)-----	57
3.1.1	INTRODUÇÃO-----	57
3.1.2	ARQUITETURA BÁSICA DO CLP-----	58
3.1.3	OPERAÇÃO BÁSICA DO CLP-----	62
3.2	LINGUAGEM DE PROGRAMAÇÃO (LADDER) -----	64
3.2.1	FUNDAMENTOS DE PROGRAMAÇÃO EM LADDER-----	65
3.2.1.1	INSTRUÇÕES BÁSICAS EM LINGUAGENS LADDER-----	67
3.2.2	IMPLEMENTAÇÃO DA LÓGICA DE CONTROLE -----	69
4	METODOLOGIA DE APLICAÇÃO DE REDES DE PETRI PARA AUTOMAÇÃO DE SISTEMAS INDUSTRIAIS COM CLP – MARPASI. -----	71
4.1	ENGENHARIA DE REQUISITOS EM SISTEMAS AUTOMATIZADOS. -----	71
4.2	DESENVOLVIMENTO DA MARPASI -----	73
4.2.1	ETAPA 1: DEFINIÇÃO DO SISTEMA-----	76
4.2.1.1	DIFICULDADES NA ELICITAÇÃO DE REQUISITOS -----	79
4.2.2	ETAPA 2: ANÁLISE, ESPECIFICAÇÃO E VALIDAÇÃO DE REQUISITOS. -----	81
4.2.2.1	ANÁLISE DE REQUISITOS -----	81
4.2.2.2	ESPECIFICAÇÃO DE REQUISITOS-----	83
4.2.2.3	VALIDAÇÃO DE REQUISITOS -----	84
4.2.3	ETAPA 3: MODELAGEM DO SISTEMA DE AUTOMAÇÃO POR MEIO DE REDES DE PETRI-----	86
4.2.4	ETAPA 4: ANÁLISE, VALIDAÇÃO E VERIFICAÇÃO DO SISTEMA POR MEIO DAS RdPs 87	
4.2.5	ETAPA 5: CONVERSÃO DO MODELO DO SISTEMA EM REDES DE PETRI PARA LINGUAGEM LADDER -----	88
4.2.6	ETAPA 6: IMPLEMENTAÇÃO E TESTE DO SISTEMA -----	91
5	IMPLEMENTAÇÃO E ANÁLISE DE APLICAÇÃO DA MARPASI - “ESTUDO DE CASO” -----	92
5.1	INTRODUÇÃO -----	92
5.2	ESTUDO DE CASO – AUTOMAÇÃO DOS TANQUES DE RECEBIMENTO DE SEBO DA INDÚSTRIA ESCOLHIDA -----	93
5.2.1	APLICAÇÃO DA “MARPASI” -----	96
5.2.1.1	ETAPA 1: DEFINIÇÃO DA OPERACIONALIDADE DO SISTEMA DE AUTOMAÇÃO -----	96
5.2.1.2	ETAPA 2: ANÁLISE, ESPECIFICAÇÃO E VALIDAÇÃO DOS REQUISITOS -----	100
5.2.1.3	ETAPA 3: MODELAGEM DO SISTEMA DE AUTOMAÇÃO POR MEIO DE REDES DE PETRI-----	104
5.2.1.4	ETAPA 4: ANÁLISE, VALIDAÇÃO E VERIFICAÇÃO DO SISTEMA POR MEIO DAS RDPs-----	111
5.2.1.5	ETAPA 5: CONVERSÃO DO MODELO DO SISTEMA EM REDES DE PETRI PARA LINGUAGEM LADDER -----	129

5.3	COMPARAÇÃO DO PROCESSO DE PROGRAMAÇÃO TRADICIONAL DE AUTOMAÇÃO DO SISTEMA ESTUDADO COM O PROCESSO DE PROGRAMAÇÃO DO SISTEMA COM A MARPASI. -----	138
6	CONCLUSÕES-----	142
6.1	SUGESTÕES DE COMPLEMENTAÇÃO A ESTE TRABALHO-----	143
7	REFERÊNCIAS BIBLIOGRÁFICAS-----	145

LISTA DE FIGURAS

Figura 2.1 - Pré-sets e Pós-sets	28
Figura 2.2 - Representação de Posição	29
Figura 2.3 - Representação de Transição.....	29
Figura 2.4 - Representação de Arco	30
Figura 2.5 - Representação de Marca ou Ficha	30
Figura 2.6 - Exemplo de uma Rede de Petri.....	30
Figura 2.7 - RdP com Múltiplos Arcos	31
Figura 2.8 – Exemplo de execução de uma RdP.	33
Figura 2.9 - RdP de Sistema de automação de máquinas operatrizes	36
Figura 2.10 - Árvore de alcançabilidade da RdP da figura 2.9	37
Figura 2.11 - A árvore de cobertura da RdP da figura 2.10	39
Figura 2.12 - Exemplo de Invariantes de Lugar	49
Figura 2.13 - Exemplo de Invariante de Transição	51
Figura 2.14 - Tela Principal do VON	54
Figura 2.15 - Menu de Elementos de RdP do VON	54
Figura 2.16 - Janela de Propriedades de Lugares(posição) do VON.....	55
Figura 2.17 - Janela de Propriedades de Transições do VON	55
Figura 2.18 - Janela de Propriedades de Arcos do VON.....	56
Figura 3.1 - Aplicação geral do controlador lógico programável.....	58
Figura 3.2 - Estrutura básica de um CLP (Fonte: SILVEIRA, SANTOS, 1999).....	58
Figura 3.3 - Estrutura básica da CPU (Fonte: SILVEIRA, SANTOS, 1999).	59
Figura 3.4 - Elementos de entrada e saída discretos (Fonte: SILVEIRA, SANTOS, 1999).61	

Figura 3.5 – Módulos de entrada e saída analógica.....	61
Figura 3.6 - Elementos de entrada e saída analógicos (Fonte: SILVEIRA, SANTOS, 1999).	61
Figura 3.7 - Circuito convencional - contatos elétricos.....	62
Figura 3.8 - Ligação elétrica por meio de CLP.	63
Figura 3.9 - Linguagem de Programação Ladder da Figura 3.8.....	63
Figura 3.10 - Componentes da programação em linguagem Ladder.....	66
Figura 3.11 - Organização do raciocínio na solução de problemas de lógica.	69
Figura 4.1 - Modelo espiral do processo de Engenharia de Requisitos.....	72
Figura 4.2 - Entradas e Saídas do processo de engenharia de requisitos [31]	73
Figura 4.3 - Etapas da metodologia MARPASI	75
Figura 4.4 - Etapa 1: Definição do Sistema.....	79
Figura 4.5 - Espiral representando o processo de interação entre elicitação e análise de requisitos.....	82
Figura 4.6 - O processo de Validação dos Requisitos	85
Figura 4.7 - Etapa 2: Análise, Especificação e Validação de requisitos	86
Figura 4.8 - Etapa 3: Modelagem do sistema de Automação por meio das Redes de Petri	87
Figura 4.9 - Etapa 4: Análise, Validação e Verificação do Sistema por meio das RdP	88
Figura 4.10 - Linguagem Ladder da função $Y = ((X0 + X1 + X2).(X4.X5)) + X3$	90
Figura 4.11 - Rede de Petri da função $Y = ((X0 + X1 + X2).(X4.X5)) + X3$	90
Figura 4.12 - Etapa 5: Conversão RdP para Linguagem Ladder.....	91
Figura 5.1 - Esquema ilustrativo dos Tanques de Recebimento de Sebo a ser automatizado	93
Figura 5.2 - Arquitetura de hardware do sistema de automação dos tanques	97

Figura 5.3 - Ciclo do Processo de Automatização dos Tanques de Recebimento de Sebo	100
Figura 5.4 - Fluxograma da Sequência 1 - Início/Repouso	101
Figura 5.5 - Fluxograma da sequência 2 - Limpeza da Tubulação	101
Figura 5.6 - Fluxograma da sequência 3 - Descarga dos Tanques	102
Figura 5.7 - Fluxograma da sequência 4 - Início da Parada	103
Figura 5.8 - Fluxograma da sequência 5 - Limpeza para Parada	103
Figura 5.9 - Rede de Petri da sequência 1 - Início/Repouso	107
Figura 5.10 - Rede de Petri da sequência 2 - Limpeza da Tubulação	108
Figura 5.11 - Rede de Petri da sequência 3 - Descarga dos Tanques	109
Figura 5.12 - Rede de Petri da sequência 4 - Início da Parada	110
Figura 5.13 - Rede de Petri da sequência 5 - Limpeza para Parada	111
Figura 5.14 - Árvore de Alcançabilidade da sequência 1 - Início/Parada	113
Figura 5.15 - Árvore de Alcançabilidade da sequência 2 –Limpeza da Tubulação com marcação M1	115
Figura 5.16 - Árvore de Alcançabilidade da sequência 2 –Limpeza da Tubulação com marcação M5	116
Figura 5.17 - Árvore de Alcançabilidade da sequência 2 –Limpeza da Tubulação com marcação M9	117
Figura 5.18 - Árvore de Alcançabilidade da sequência 2 –Limpeza da Tubulação com marcação M13	118
Figura 5.19 - Árvore de Alcançabilidade da sequência 3 - Descarga dos Tanques - Marcação M1	120
Figura 5.20 - Árvore de Alcançabilidade da sequência 3 – Descarta dos Tanques – Marcação M2	122

Figura 5.21 - Árvore de Alcançabilidade da seqüência 3– Descarga dos Tanques –	
Marcação M3	124
Figura 5.22 - Árvore de Alcançabilidade da seqüência 3– Descarga dos Tanques –	
Marcação M4	126
Figura 5.23 - Diagrama Ladder do Sistema modelado em Redes de Petri	131
Figura 5.24 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação) .	132
Figura 5.25 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação) .	133
Figura 5.26 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação) .	134
Figura 5.27 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação) .	135
Figura 5.28 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação) .	136
Figura 5.29 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação) .	137
Figura 5.30 - Diagrama Ladder do Sistema modelado em Redes de Petri	138

LISTA DE TABELAS

Tabela 2-1 - Principais Propriedades das Redes de Petri	33
Tabela 2-2 - Propriedades x Método de Análise das Redes de Petri	52
Tabela 3-1 - Principais Instruções da Linguagem Ladder	68
Tabela 3-2 - Implementação da lógica booleana/De Morgan em linguagem Ladder.	70
Tabela 4-1 - Resumo de algumas Técnicas de Elicitação de requisitos	78
Tabela 4-2 - Modelagem de redes de Petri e linguagem Ladder.	89
Tabela 4-3 - Resumo da Metodologia MARPASI.	91
Tabela 5-1 - Identificação das referências citadas no esquema ilustrativo da figura 5.1	94
Tabela 5-2 - Descrição das Posições (condições) das Redes de Petri	106
Tabela 5-3 - Marcações iniciais possíveis da RdP das figuras 5.9 a 5.13	112
Tabela 5-4 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.14.....	113
Tabela 5-5 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.15.....	115
Tabela 5-6 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.16.....	116
Tabela 5-7 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.17.....	117
Tabela 5-8 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.18.....	118
Tabela 5-9 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.19.....	121
Tabela 5-10 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.20.....	123
Tabela 5-11 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.21.....	125
Tabela 5-12 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.22.....	127
Tabela 5-13 - Lista de Entradas (I) do CLP.....	129
Tabela 5-14 - Lista de Saídas (O) do CLP.....	130
Tabela 5-15 - Comparação do Tamanho do Programa Ladder sem e com a MARPASI...	139

LISTA DE GRÁFICOS

Gráfico 1.1 - Custo: Hardware x Software - CLP	24
Gráfico 1.2 - Principais características buscadas em um CLP no momento da aquisição ...	25
Gráfico 3.1 - Linguagem de Programação em uso (Fonte: Control Engeneering-12/05). ...	65

LISTA DE ABREVIATURAS E SIGLAS E TERMOS

ABNT: Associação Brasileira de Normas Técnicas

AI: *Analog Input*

AO: *Analog Output*

CLP: Controlador Lógico Programável

IEC: *International Electrotechnical Committee*

ISA: *The Instrumentation, Systems, and Automation Society*

ISO: *International Organization for Standardization*

I/O: *Input/Output*

MARPASI: *Metodologia de Aplicação das Redes de Petri em Automação de Sistemas Industriais*

NEMA: *Nation Electric Manufacturs Association*

RdP: Redes de Petri

SVC: Sistema de variáveis contínuas

SED: Sistemas a eventos discretos

LISTA DE SÍMBOLOS MATEMÁTICOS

$\cup$: união de conjuntos

$\cap$: intersecção de conjuntos

$\in$: pertence

$\forall$: para todo

$\emptyset$: conjunto vazio

$|$: tal que

$\subset$: está contido

X^T : transposta da matriz X

GLOSSÁRIO

Livelocks: ciclo de disparos de transições que faz com que se repitam sempre os mesmos estados, sem possibilidade de mudança de direção.

Deadlocks: quando duas ou mais transições ficam em impasse, ou seja, uma dependendo da outra para ser disparada, fazendo com que não ocorram mudanças de estados.

Elicitação: O termo elicitación vem da palavra inglesa *elicitation*, que significa descobrir, desvendar algo obscuro. Embora o termo elicitación não exista no vocabulário da Língua Portuguesa, ele vem sendo aceito e utilizado na comunidade de Engenharia de Requisitos e na Engenharia de Inteligência Artificial para aplicações e desenvolvimento de sistemas especialistas como uma “tradução” do termo *elicitation*.

Grafo: Um grafo é um conjunto de pontos, chamados vértices (nodos ou nós), conectados por linhas, chamadas de arestas (ou arcos). Dependendo da aplicação, arestas podem ou não ter direção, pode ser permitido ou não arestas ligarem um vértice a ele próprio e vértices e/ou arestas podem ter um peso (numérico) associado. Se as arestas têm uma direção associada (indicada por uma seta na representação gráfica) temos um grafo direcionado, ou dígrafo.

Variável de Estado: A variável de estado de uma RdP de n posições é o vetor $\mathbf{m}$ definido pela marcação da RdP; $\mathbf{m} = [m(p_1) \ m(p_2) \ m(p_3) \dots \ m(p_n)]$.

Vetor de Estado: O vetor de estado $\mathbf{M}$ de uma RdP representa o estado da RdP, no qual o primeiro valor do vetor é quantidade de fichas (tokens) na primeira posição da rede, o segundo é a quantidade de fichas na segunda posição, e assim por diante. Tal representação descreve completamente o estado de uma rede de Petri.

RESUMO

Devido às necessidades do mundo moderno, os sistemas de automação têm aumentado sua complexidade, fazendo com que sejam desenvolvidas ferramentas de engenharia cada vez mais poderosas para modelá-los e analisá-los. Em sistemas de automação industrial, os Controladores Lógicos Programáveis (CLPs) têm sido amplamente empregados. Os CLPs são geralmente programados por meio da linguagem de programação Ladder, uma das cinco linguagens definidas pela IEC 61131-3. Entretanto, apesar da linguagem de programação Ladder ser flexível e de fácil aprendizado por parte dos usuários, ela apresenta limitações quanto à: detecção de erros no algoritmo de controle do sistema de automação; torna as modificações muito trabalhosas e não possibilita a simulação, análise de performance e análise operacional do sistema.

Este trabalho de pesquisa apresenta o desenvolvimento e os testes da metodologia denominada MARPASI - Metodologia de Aplicação das Redes de Petri em Automação de Sistemas Industriais. Como o desenvolvimento da MARPASI foi efetuado baseado na teoria de Redes de Petri, este trabalho também apresenta uma revisão bibliográfica sobre o tema de aplicação de Redes de Petri para a programação de CLP. A MARPASI possibilita analisar um sistema de automação por meio das Redes de Petri e na geração da linguagem de programação Ladder. Portanto, o emprego da MARPASI contribui para a otimização do processo de engenharia de automação e também para a programação mais eficiente de CLPs.

Palavras-chave: Redes de Petri. Controladores Programáveis. Automação Industrial.

ABSTRACT

Due to the needs of the modern world, the systems of automation have increased their complexity, forcing the development of ever more powerful engineering tools to shape and analyse them. In systems of industrial automation, the Programmable Logic Controllers (PLCs), have been widely applied. The PLCs are generally programmed through the use of the Ladder programming language, one of the five languages defined by the IEC 61131-3. Unfortunately, while the Ladder programming language is flexible, and easily learned by its users, it evinces limitations concerning: error detection in the control algorithm of the automation system: makes modifications very laborious and does not allow any simulation, performance analysis and operational systems analysis.

This study presents the development and tests of the methodology denominated as MARPASI – Methodology of the application of Petri Net in the automation of industrial systems. Since the development of MARPASI was made based on Petri Net Theory, this study also presents a bibliographic review about the theme of the application of Petri Net for the programming of PLC. MARPASI makes it possible to analyse a system of automation through Petri Net and in the generation of the Ladder programming language. Because of these, the utilization of MARPASI contributes for capacities, the optimization of automation engineering processes, and also for a more efficient programming of the PLCs.

Key-words: Petri Net. Programmable Controllers. Industrial Automation.

1 INTRODUÇÃO

Atualmente existem inúmeros processos automatizados. Tais processos vêm se tornando cada vez mais complexos em função das necessidades do mundo moderno e, portanto, demandam nas fases de projeto e de implementação ferramentas de engenharia cada vez mais poderosas para modelá-los e analisá-los da maneira mais eficiente possível. Os Controladores Lógicos Programáveis – CLP têm sido amplamente utilizados em sistemas de automação industrial. Os CLPs trabalham em redes de automação interligados a sistemas supervisórios, sensores, atuadores e, geralmente, são programados em linguagem Ladder. Assim sendo, os Controladores Lógicos Programáveis - CLPs são equipamentos de grande relevância em processos automatizados.

Sistemas de automação complexos exigem alto nível de controle de eventos discretos e baixo nível de implementação em hardware [01]. Programas escritos em lógica Ladder são amplamente empregados em CLPs. A desvantagem da linguagem de programação Ladder é que os programas são difíceis de corrigir e de modificar tornando o seu desenvolvimento e manutenção menos eficientes do que o desejado. De fato, os programas escritos em lógica Ladder atingem dimensões tais, que tornam difíceis a detecção de erros e de análise operacional do sistema.

Recentemente, motivado pela necessidade de mais confiabilidade e métodos modulares de programação de CLPs, algumas pesquisas têm sido feitas considerando metodologias alternativas de projeto de sistemas de controle. Nestas incluem-se as Redes de Petri. Contudo, as pesquisas envolvendo Redes de Petri ainda não têm sido amplamente empregadas pela indústria de automação.

A teoria das Redes de Petri foi criada em 1962 por Carl Adam Petri com objetivo de permitir a modelagem de sistemas a eventos discretos. As Redes de Petri podem ser aplicadas como ferramenta auxiliar de programação para eliminação das desvantagens da programação Ladder.

1.1 OBJETIVO

O objetivo deste trabalho é mostrar a teoria das redes de Petri como ferramenta de auxílio na programação dos Controladores Lógicos Programáveis – CLPs, apontando o caminho de como fazer análise (verificação e validação) de algoritmos de controle, usando a linguagem de programação Ladder para implementação em hardware.

As redes de Petri podem ser utilizadas como ferramenta de modelagem, verificação e validação de sistemas de controle de eventos discretos. Assim sendo, as Redes de Petri podem auxiliar na geração do código a ser implementado no .

1.2 MOTIVAÇÃO

Os sistemas controlados por CLPs são extremamente variados, com aplicações desde sistemas de manufaturas, controle de processos químicos, distribuição de energia, entre outros. Portanto, há necessidade de uma ferramenta de auxílio na programação de CLPs.

As Redes de Petri podem ser aplicadas na: especificação, modelagem, análise, avaliação de desempenho e de implementação de sistemas de automação. As mesmas possibilitam a identificação de *livelocks* e *deadlocks*.

A Rede de Petri é, portanto, um formalismo que permite a modelagem de sistemas dinâmicos discretos com grande poder de expressividade, permitindo representar com facilidade todas as relações de causalidade entre processos em situações de: sequencialidade, conflito, concorrência e sincronização. O interesse pelas redes de Petri tem aumentado, nos últimos anos, devido às possibilidades representadas pelas redes de alto nível, que permitem modelar sistemas com maior complexidade, por gerarem modelos mais compactos e pelas extensões que incorporam parâmetros temporais.

Outro ponto importante é a qualidade do programa a ser implementado no CLP. Em muitos casos, são necessários mais de um programador para o desenvolvimento do programa de CLP e a experiência mostra que o risco devido a erros de programação cresce de acordo com o número de programadores envolvidos e, conseqüentemente, com o tamanho do programa. A experiência mostra também, que as novas plantas industriais apresentam problemas após a instalação. Algumas falhas podem interromper a produção e, no pior caso, também podem causar danos aos equipamentos e acidentes com vítimas humanas.

A grande maioria dos programas de controle é escrita utilizando pacotes de softwares proprietários dos fabricantes de produtos de controle. Muitos destes pacotes não dispõem de recursos adequados para trabalhar com módulos, para reutilização de código e da documentação.

Os softwares para aplicações de controle são adaptados para características específicas de uma única planta industrial. Muitos usuários consideram difícil absorver o custo total de desenvolvimento de programa de controle. Um usuário sem experiência no desenvolvimento de programa de controle consegue apresentar apenas uma pobre descrição funcional para o desenvolvedor. Em muitos casos, isto leva ao desenvolvimento de um software que atende parcialmente aos requisitos do usuário. Mesmo pequenas modificações e melhorias implicam em custos para implementação, especialmente na fase final de desenvolvimento [05].

O crescente aumento da velocidade de desenvolvimento de hardware tem levado a uma constante queda dos preços. Com a crescente melhoria da relação custo-desempenho do hardware, o custo total da instalação é cada vez mais determinado pelo tempo para desenvolvimento do programa de controle, conforme gráfico 1.1.

Gráfico 1.1 - Custo: Hardware x Software - CLP

Uma das principais características buscadas em um CLP no momento da aquisição é o software de programação universal, conforme gráfico 1.2, isto é, a capacidade do CLP de usar software de programação universal para diversas finalidades. Isto mostra que existe uma preocupação com software e conseqüentemente isto exige que seja buscada cada vez mais uma ferramenta que auxilie na programação do CLP.

Gráfico 1.2 - Principais características buscadas em um CLP no momento da aquisição

Fonte: Control Engineering December 1, 2005

2 TEORIA GERAL DAS REDES DE PETRI PARA APLICAÇÃO NA ANÁLISE DE SISTEMAS DE AUTOMAÇÃO

2.1 INTRODUÇÃO

Esse capítulo apresenta um breve resumo da teoria geral das Redes de Petri com o objetivo de mostrar a sua aplicação para análise de desempenho e operacional de sistemas modeláveis pelas mesmas.

2.2 INTRODUÇÃO DA TEORIA DAS REDES DE PETRI

O conceito de Redes de Petri foi introduzido por Carl Adam Petri , em sua tese de doutoramento intitulada *Kommunikation mit Automaten* (Comunicação com Autômato), em 1962, na faculdade de Matemática e Física da Universidade de Darmstadt, na então Alemanha Ocidental. Nos anos seguintes, à medida em que seu trabalho foi sendo introduzido em outros países, esse conceito difundiu-se e vem se aperfeiçoando através da promoção de “workshops”, conferências internacionais e publicações de novos trabalhos.

Redes de Petri são uma ferramenta gráfica e matemática de modelagem (descrição/especificação) que podem ser aplicadas em diversos tipos de sistemas, apresentando um bom nível de abstração em comparação com outros modelos gráficos. Usando-se RdP , pode-se modelar sistemas paralelos, concorrentes , assíncronos e não-determinísticos.

Sendo um modelo do tipo condição-evento, cada evento possui pré-condições que vão permitir sua ocorrência e pós-condições decorrentes desta, que são, por sua vez, pré-condições de outros eventos posteriores.

Áreas como redes de computadores e protocolos de comunicação, sistemas operacionais, programação paralela, bancos de dados distribuídos e sistemas flexíveis de manufatura, enfim, qualquer área em que: sequencialidade, conflito, concorrência e sincronização seja um fator preponderante, são passíveis de ter vantajosamente aplicadas as Redes de Petri. Outra característica importante é que ela permite a análise da estrutura e do comportamento dinâmico do sistema modelado.

2.3 DEFINIÇÕES DE REDES DE PETRI

Uma Rede de Petri (RdP) é uma quintupla $(\mathbf{P}, \mathbf{T}, \mathbf{F}, \mathbf{W}, \mathbf{M}_0)$ [09] em que:

- $\mathbf{P} = \{p_1, \dots, p_n\}$ é um conjunto finito de posições ou lugares;
- $\mathbf{T} = \{t_1, \dots, t_m\}$ é um conjunto finito de transições;
- $\mathbf{F} \subset (\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P})$ é um conjunto finito de arcos contidos no conjunto $(\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P})$, em que $(\mathbf{P} \times \mathbf{T})$ representa o conjunto dos arcos orientados de p_i para t_j também designados por (p_i, t_j) , e $(\mathbf{T} \times \mathbf{P})$ representa o conjunto dos arcos orientados de t_j para p_i ou (t_j, p_i) ;
- $\mathbf{W} = \{1, 2, 3, \dots\}$ é a função que atribui um peso w (um número inteiro) a cada arco;
- $\mathbf{M}_0$ é m vetor cuja i -ésima coordenada define o número de marcas (tokens) na posição p_i , no início da evolução da rede, isto é, a marcação inicial da rede.

Os conjuntos $\mathbf{T}$ e $\mathbf{P}$ são disjuntos, isto é, $\mathbf{T} \cap \mathbf{P} = \emptyset$.

$n = |\mathbf{P}|$ é a cardinalidade do conjunto $\mathbf{P}$, o número de posições da RdP.

$m = |\mathbf{T}|$ é o número de transições da RdP.

Em certos autores, o conjunto dos arcos $\mathbf{F}$ é substituído por dois conjuntos de funções: o das funções de saída $\mathbf{O}$, que definem mapeamentos (arcos orientados) das transições a subconjuntos de posições; o das funções de entrada $\mathbf{I}$, que definem o mapeamento das posições para subconjuntos de transições.

Outro conceito importante em Redes de Petri são os pré-sets e pós-sets. O símbolo de pré-set de transição é $*t$ e o pós-set de transição é t^* .

Pré-set de t : $= *t := \{\forall p_i \in P \mid F \in (p_i, t)\}$, ou seja, o pré-set de t , $*t$ é o conjunto das posições em P a partir das quais existe arco para a transição t ;

Pós-set de t : $= t^* := \{\forall p_i \in P \mid F \in (t, p_i)\}$, ou seja, o pós-set de t , t^* é o conjunto das posições em P a partir das quais existe arco oriundo da transição t ;

O símbolo de pré-set de posição é $*p$ e o pós-set de posição é p^* .

Pré-set de p : $= *p := \{\forall t_j \in T \mid F \in (t_j, p)\}$

Pós-set de p : $= p^* := \{\forall t_j \in T \mid F \in (p, t_j)\}$

A Figura 2.1.a e 2.1.b ilustram os exemplos de pré-set e pós-set.

Figura 2.1 - Pré-sets e Pós-sets

As RdP são formadas por dois tipos de componentes: um ativo denominado Transição e outro passivo denominado Posição. As posições correspondem às variáveis de estado e as transições às ações (eventos) realizadas pelo sistema. Os arcos que interligam posições às transições correspondem à relação entre as condições verdadeiras, que em um dado momento possibilitam a execução das ações, enquanto os arcos que interligam transições às posições representam a relação entre as ações e as condições que se tornam verdadeiras com a execução das ações.

2.4 SÍMBOLOS USUAIS PARA REPRESENTAÇÃO GRÁFICA DAS RdP

POSIÇÃO (figura 2.2): modela condição, estado ou recurso que devem ser certificados para os eventos acontecerem.:

Figura 2.2 - Representação de Posição

TRANSIÇÃO (figura 2.3): modela um evento, tal como o início de uma operação correspondente aos eventos que caracterizam as mudanças de estado do sistema:

Figura 2.3 - Representação de Transição

ARCO ORIENTADO (figura 2.4): interliga uma posição a uma transição ou vice-versa, encadeando condições e eventos:

Figura 2.4 - Representação de Arco

MARCA OU FICHA (figura 2.5): representa um recurso disponível ao agente; o posicionamento dessas fichas em algumas posições da rede constitui a marcação. A evolução da marcação permite modelar o comportamento dinâmico do sistema:

Figura 2.5 - Representação de Marca ou Ficha

A figura 2.6 exemplifica a simbologia gráfica de uma RdP:

Figura 2.6 - Exemplo de uma Rede de Petri

$$P=\{p_1, p_2\} \quad T=\{t_1\} \quad A=\{(p_1, t_1), (t_1, p_2)\}$$

$$W: w(p_1, t_1) = 2, w(t_1, p_2) = 1$$

$M_0 = (1 \ 0)$ pois $m(p_1)=1$, $m(p_2)=0$ no início.

Os vértices de um grafo associados a uma RdP podem ser interligados por múltiplos arcos. Por exemplo, um lugar pode ser conectado a uma transição por meio de diversos arcos ou vice-versa. Por conveniência, pode-se substituir os múltiplos arcos por um único arco valorado, onde o numeral associado ao arco corresponde ao número de arcos que interligam os vértices (figura 2.7).

Figura 2.7 - RdP com Múltiplos Arcos

2.5 EXECUÇÃO DAS REDES DE PETRI

Chama-se de execução da RdP a movimentação das marcas pela rede de acordo com certas regras. As movimentações das marcas são efetuadas em duas fases: habilitação e disparo de transição.

Uma transição $t_j \in T$ numa RdP está **habilitada** por uma marcação m se ocorre que, para todo $p_i \in {}^*t$, $m(p_i) \geq w(p_i, t_j)$, isto é, uma transição t_j pertencente ao conjunto de transição T está **habilitada** se as marcações de todas as posições p_i pertencentes ao conjunto de pré-set de t (*t) é maior ou igual ao peso dos arcos de p_i a t_j .

Uma transição é **disparada** por meio de duas operações:

- Remover marcas das posições do pré-set (tantas marcas quanto for o peso do arco correspondente) e
- Depositar em cada uma das posições do pós-set tantas marcas quanto for o peso do arco correspondente.

A figura 2.8.a ilustra as etapas de movimentação das marcas na RdP. A transição t_1 está habilitada a disparar, pois as posições p_1 e p_2 possuem uma marcação cada e o arco que interliga p_1 a t_1 e p_2 a t_1 tem peso 1.

Após o disparo de t_1 , as marcações são retiradas de p_1 e p_2 e uma marcação é adicionada em p_3 , habilitando a transição t_2 (figura 2.8.b).

Após o disparo de t_2 , as marcações são retiradas de p_3 e p_4 e uma marcação é adicionada em p_5 (figura 2.8.c).

Figura 2.8 – Exemplo de execução de uma RdP.

2.6 PRINCIPAIS PROPRIEDADES DAS REDES DE PETRI

As RdP possuem determinadas propriedades que podem servir de ferramentas de auxílio na análise de sistemas modelados pelas mesmas. Este trabalho de pesquisa apresenta uma tabela resumo das principais propriedades das RdP[09,10].

Tabela 2-1 - Principais Propriedades das Redes de Petri

PROPRIEDADE	DESCRIÇÃO
Limitação	Um lugar é k -limitado se o número de marcas neste determinado lugar não exceder a um número k . Uma Rede é k -limitada se todos os lugares foram k-limitados. k é um número inteiro positivo.
Segurança	Uma RdP é segura (safe) se ela é k-limitada com k=1, isto é, se o número de marcas é 0 ou 1 em todas as posições da rede.

Conservação	Uma RdP em que a soma total das marcas permanece constante na sua execução é dita conservativa .
Vivacidade	Uma transição é viva, dado um estado inicial $\mathbf{M}_0$, se e somente se ela é habilitada a partir de algum estado decorrente de $\mathbf{M}_0$. Uma RdP é viva, dado um estado inicial $\mathbf{M}_0$, se e somente se todas as suas transições são vivas.
Impasse (<i>deadlock</i>)	Uma RdP está em impasse(<i>deadlock</i>), quando não há transição habilitada e portanto nenhum estado sucessor.
Alcançabilidade	Verifica se determinado estado $\mathbf{M}$ é alcançável a partir de um dado estado inicial $\mathbf{M}_0$ quando uma ou mais transições forem disparadas.
Reversibilidade	Uma RdP é dita reversível , quando existe o retorno à marcação inicial $\mathbf{M}_0$ ou uma outra marcação qualquer $\mathbf{M}$.

2.7 TÉCNICAS DE ANÁLISE DAS REDES DE PETRI

Entende-se por análise das RdP a verificação das propriedades das Redes de Petri. Basicamente, são dois os métodos matemáticos usuais para verificar propriedades em uma RdP [09,10]:

- a) com base nas árvores de alcançabilidade e cobertura;
- b) com base nas matrizes de incidência e equação de estado.

Um terceiro método extremamente eficiente para verificação das propriedades das Redes de Petri é a simulação computacional.

2.7.1 ANÁLISE PELAS ÁRVORES DE ALCANÇABILIDADE E DE COBERTURA

A árvore de alcançabilidade (*reachability*) é um gráfico do tipo árvore em que os nós são os vetores de estado, alcançados sucessiva e alternativamente pela rede, e os arcos são as correspondentes transições executadas.

Por meio das árvores de alcançabilidade organiza-se o trabalho de enumerar exaustivamente as marcações alcançáveis começando por M_0 . Todas as transições habilitadas são disparadas, gerando M_1 , M_2 e assim por diante. Em determinadas situações pode ocorrer a habilitação simultânea de transições; então, é preciso tomar cada alternativa como uma nova “raiz” ou “ramo” da árvore.

A árvore de alcançabilidade é uma representação gráfica que emprega símbolos retangulares, números e arcos. Os símbolos retangulares representam todos os vetores de estado da RdP.

O número de marcas de cada posição da RdP em cada estado é representado por números dentro dos retângulos. Os arcos representam as transições habilitadas.

A figura 2.9 ilustra um sistema de automação onde dois operadores S1 e S2 operam 3 Máquinas Operatrizes MO1, MO2 e MO3. S1 opera MO1 e MO2 e S2 opera MO1 e MO3. O processamento das peças se inicia por MO1. Considera-se as seguintes condições:

- Início de processamento com M1 em espera ($\underline{MO1\ ESP} = p1$)
- Peça pronta por M1 ($\underline{PP_MO1} = p2$) e a espera de M2 ($\underline{MO2\ ESP} = p3$) ou M3 ($\underline{MO3\ ESP} = p4$)
- Peça pronta = $\underline{PP_MO2}$ e $\underline{MO3} = p5$
- M1 parada = $\underline{MO1_PAR} = p6$
- M2 parada = $\underline{MO2_PAR} = p7$
- M3 parada = $\underline{MO3_PAR} = p9$

- S1 parado = S1_PAR = p8
- S2 parado = S2_PAR = p10
- S1 operando MO1 = S1_OP_MO1 = p11
- S1 operando MO2 = S1_OP_MO2 = p12
- S2 operando MO1= S2_OP_MO1 = p14
- S2 operando MO3= S2_OP_MO3 = p3

Figura 2.9 - RdP de Sistema de automação de máquinas operatrizes

Figura 2.10 - Árvore de alcançabilidade da RdP da figura 2.9

A figura 2.10 mostra, do primeiro ao nono nível, a árvore de alcançabilidade da RdP da figura 2.9, com uma marca inicial na posição p_1 . Está habilitada inicialmente a transição t_1 que resulta no estado $(0,0,0,0,0,0,0,0,0,0,1,0,0,0)$, habilitando a transição t_2 . No terceiro nível têm-se duas “raízes”, pois, a transição t_3 e t_{12} encontram-se habilitadas a partir do estado $(0,1,0,0,0,1,0,1,0,0,0,0,0,0)$.

Uma inconveniência da árvore de alcançabilidade é a ocorrência de um número ilimitado de nós (e níveis). Um objetivo, então, é reduzir uma árvore de alcançabilidade infinita a uma representação do mesmo tipo, porém finita. O processo de redução de uma árvore de alcançabilidade infinita a uma árvore finita é desenvolvido a partir de duas observações:

- Nós que são idênticos a nós anteriores não precisam ser prosseguidos, porque suas consequências já estão registradas na árvore;

b) Em um circuito fechado na RdP, onde, a cada execução completa aumenta o número de marcas de uma dada posição, esse número vai certamente a infinito. Para fins de analisar propriedades como vivacidade, limitação, etc., não é preciso continuar indefinidamente a árvore; para isso, será usado o símbolo ω , chamado pseudo-infinito, que representa “um número arbitrariamente grande”. Para qualquer constante a , define-se:

- i. $a < \omega$, para qualquer número inteiro a ;
- ii. $\omega \leq \omega$;
- iii. $\omega \pm a = \omega$, para qualquer número inteiro a .

A árvore de alcançabilidade, reduzida por meio do pseudo-infinito ω , é chamada de **árvore de cobertura**. Para construção sistemática da árvore de cobertura adota-se a seguinte nomenclatura:

- **Raiz:** nó ou estado inicial.
- **Terminal:** nó ou estado a partir do qual não há transições habilitadas.
- **Duplicado:** um nó idêntico a outro já alcançado.
- **Dominante:** o nó M domina o nó M'' quando $m(p_i) \geq m''(p_i)$, para todos os valores de i , e ainda $m(p_i) > m''(p_i)$, para pelo menos um valor de i .

As etapas para construção da Árvore de Cobertura conforme referência [09] são:

- a) Inicia-se criando a raiz da árvore, o nó com o vetor de estado inicial M_0 , chamando-o de “novo”.
- b) Enquanto houver nó “novo” seleciona-se um deles e chama-se o de M .
 - b.1. Verifica-se cada transição de saída de M , se não há transição habilitada, M é um nó terminal da árvore, assinala-o com T.
 - b.2. Se uma transição é habilitada, executa-se-a, resultado M' como novo estado, chama-se-o de “novo”

- b.2.1. Se existe um nó M'' no caminho da raiz até M' , tal que o nó M' domine M'' , altera-se o valor da coordenada $m'(p_i)$ do vetor M' para ω , em toda posição p_i onde $m'(p_i) > m''(p_i)$.
- b.2.2. Se a coordenada $m(p_i)$ do vetor M vale ω , então $m(p_i) = \omega$, pois, $\omega \pm a = \omega$.
- b.2.3. Se M' é igual a um estado qualquer já existente na RdP, chama-se-o de “duplicado” e assinala-se-o com a letra “D”.
- c) Retorna-se ao passo 2, para outros nós “novos”.
- d) Quando todos os nós “novos” forem terminais (T) ou Duplicados(D), a árvore está completa.

A figura 2.11 mostra a árvore de cobertura da RdP da figura 2.9.

Figura 2.11 - A árvore de cobertura da RdP da figura 2.10

2.7.1.1 LIMITAÇÕES DA ÁRVORE DE COBERTURA NA ANÁLISE DAS REDES DE PETRI.

A árvore de cobertura pode ser usada para solucionar os problemas de segurança, limitação e conservação, mas em geral não pode ser usada para solucionar os problemas de alcançabilidade e vivacidade ou para determinar quais seqüências de disparo são possíveis. Esses problemas são limitados pela existência do símbolo ω que esconde informações importantes sobre a RdP[10].

Como as seqüências de disparo não são conhecidas, a análise da vivacidade não é possível, pois conforme a seqüência de disparos possíveis pode haver uma seqüência em que a rede não é viva.

É importante salientar também que na mesma estrutura de uma RdP, para uma nova marcação inicial, tem que construir uma nova árvore, pois as propriedades podem ser alteradas.

2.7.2 ANÁLISE DAS RdP PELA MATRIZ DE INCIDÊNCIA E EQUAÇÕES DE ESTADOS.

Tem-se que:

- a) a execução de uma transição t_j numa RdP de n posições e m transições $(P, T, F, W, \mathbf{M}_0)$ ocorre se e somente se a marcação $m(p_i) \geq \text{peso do arco } w(p_i, t_j)$ para $\forall p_i \in \text{pré-set de } t_j$.
- b) a marcação $\mathbf{M}$ de cada posição p_i é alterada para $\mathbf{M}'$ pela execução de t_j ; algebricamente, pode-se escrever para $\forall p_i \in P$:

$m'(p_i) = m(p_i) - w(p_i, t_j)$ se $p_i \in \text{pré-set de } t_j$, isto é, a nova marcação na posição p_i , $m'(p_i)$, é igual à marcação atual, $m(p_i)$, menos o peso do arco de p_i para t_j , se p_i pertence ao pré-set de t_j .
$m'(p_i) = m(p_i) + w(t_j, p_i)$ se $p_i \in \text{pós-set de } t_j$ isto é, a nova marcação na posição p_i , $m'(p_i)$, é igual à marcação atual, $m(p_i)$, mais o peso do arco de t_j para p_i , se p_i pertence ao pós-set de t_j .
$m'(p_i) = m(p_i)$ nos outros casos, $\forall p_i \in p_i$, em que: <ul style="list-style-type: none"> - $w(p_i, t_j)$ são os pesos dos arcos de p_i a t_j em que p_i são as posições do pré-set de t_j, e - $w(t_j, p_i)$ são os pesos dos arcos de t_j a p_i em que p_i são as posições do pós-set de t_j.

A matriz de incidência A , de uma Rede de Petri, de m transições e n posições, é a matriz m linhas e n colunas, de números inteiros[09]:

$$A = a_{ij}^+ - a_{ij}^- \quad (1)$$

Onde $a_{ij}^+ = w(t_j, p_i)$, para $p_i \in \text{pós-set de } t_j$ e $a_{ij}^- = w(p_i, t_j)$ para $p_i \in \text{pré-set de } t_j$

Os a_{ij} são positivos ou negativos, conforme o arco associado saia de, ou entre em, t_j . Os $w(t_j, p_i)$ e $w(p_i, t_j)$ são sempre positivos; convencionou-se chamá-los de a_{ij}^+ (peso dos arcos de saída de t_j), e de a_{ij}^- (peso dos arcos de entrada em t_j).

Chama-se matriz de incidência de entrada a matriz $A^- = (a_{ij}^-)$ e matriz de incidência de saída $A^+ = (a_{ij}^+)$. Consequentemente, a matriz de incidência A

$$A = (A^+) - (A^-) \quad (2)$$

A matriz de incidência $A = (A+) - (A-)$

$$A = \begin{array}{c|cccccccccccccccc|c} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} & p_{12} & p_{13} & p_{14} & \\ \hline & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & t_1 \\ & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & t_2 \\ & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & t_3 \\ & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & t_4 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & t_5 \\ & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_6 \\ & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & t_7 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & t_8 \\ & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & t_9 \\ & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & t_{10} \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & t_{11} \\ & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & t_{12} \end{array}$$

Por meio da Matriz de incidência é possível verificar se determinada RdP é conservativa, se determinada marcação $\mathbf{M}$ é alcançável e qual a seqüência de disparo que leva a uma determinada marcação[09,10].

O vetor disparo da transição t_j é um vetor $\mathbf{u}_j$, m linhas x 1 coluna, de elementos todos nulos, exceto o j -ésimo, que vale 1.

Em termos de a_{ij} e de $\mathbf{u}_j$, a execução da transição t_j leva a marcação $m(p_i)$ à $m'(p_i)$ por meio da equação:

$$m'(p_i) = m(p_i) + a_{ij} \cdot 1$$

e leva o vetor de estado $\mathbf{M}$ ao $\mathbf{M}'$, por meio da equação de estado:

$$\mathbf{M}' = \mathbf{M} + \mathbf{u}_j^T \cdot \mathbf{A} \quad (3)$$

em que $\mathbf{M}$ e $\mathbf{M}'$ são respectivamente o estado inicial e estado que se deseja alcançar, respectivamente, e possuem 1 linha por n colunas. $\mathbf{A}$ é a matriz de incidência e

possui m linhas e n colunas e $\mathbf{u}_j^T$ é o vetor contagem de disparo, transposto, e possui 1 linha por n colunas.

O vetor de contagem de disparos $\mathbf{u}_j$, relacionado a uma seqüência de disparos de transições, é um vetor em que o j -ésimo elemento conta quantas vezes a transição j foi disparada.

Algebricamente, o vetor de contagem $\mathbf{u}_j$ é a soma dos vetores de disparos $\mathbf{u}_i$ ocorridos, sem conter informação sobre a ordem dos disparos (a qual está presente nas árvores de alcançabilidade). Por exemplo, $\mathbf{u} = (2 \ 3 \ 1)^T$ pode ser o vetor de contagem de qualquer seqüência de disparos de transições contendo 2 disparos de t_1 , 3 de t_2 , e 1 de t_3 ; por exemplo:

$$\mathbf{u} = \mathbf{u}_1 + \mathbf{u}_1 + \mathbf{u}_2 + \mathbf{u}_2 + \mathbf{u}_2 + \mathbf{u}_3, \text{ ou } \mathbf{u}_1 + \mathbf{u}_2 + \mathbf{u}_1 + \mathbf{u}_2 + \mathbf{u}_2 + \mathbf{u}_3, \text{ ou } \dots\dots$$

Observa-se que um único vetor $\mathbf{u}_j$ pode ser originado por várias seqüências de disparo distintas, algumas delas eventualmente inviáveis por falta de habilitação.

Para exemplificar o uso da equação de estado, considere a RdP da figura 2.9 que tem a seguinte matriz de incidência A :

$$A = \begin{array}{c|cccccccccccccc|c} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} & p_{12} & p_{13} & p_{14} & \\ \hline & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & t_1 \\ & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & t_2 \\ & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & t_3 \\ & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & t_4 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & t_5 \\ & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_6 \\ & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & t_7 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & t_8 \\ & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & t_9 \\ & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & t_{10} \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & t_{11} \\ & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & t_{12} \end{array}$$

Com o estado inicial $\mathbf{M}_0 = \mathbf{M} = (1,0,0,0,0,0,0,0,0,0,0,0,0,0)$, tem-se a transição t_1 habilitada que leva a nova marcação $\mathbf{M}'$:

$$\mathbf{M}' = \mathbf{M} + \mathbf{u}_j^T \mathbf{A}$$

$$\mathbf{M}' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{M}' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{M}' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Para determinar se o estado $\mathbf{M}' = (0,1,1,0,0,1,0,0,0,0,0,0,0,0)$ é alcançável a partir do estado inicial $\mathbf{M} = (1,0,0,0,0,0,0,0,0,0,0,0,0,0)$, obtém-se:

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \mathbf{u}_j^T \cdot \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = [-u_1+u_{10}, u_2+u_{10}, u_3-u_4, -u_9+u_{10}, u_6+u_7, u_2-u_{12}, u_5-u_6, u_2-u_3, -u_7+u_8, -u_{11}+u_{12}, u_1-u_2, u_4-u_5, -u_8+u_9, -u_{10}+u_{11}]$$

que leva às seguintes equações:

$$\begin{aligned} (1) \quad & -1 = -u_1 + u_{10} \\ (2) \quad & 1 = u_2 + u_{10} \\ (3) \quad & 1 = u_3 - u_4 \quad \text{=====>} \quad u_3 = 1 + u_4 \\ (4) \quad & 0 = -u_9 + u_{10} \quad \text{=====>} \quad u_9 = u_{10} \\ (5) \quad & 0 = u_6 + u_7 \quad \text{=====>} \quad u_6 = -u_7 \\ (6) \quad & 1 = u_2 - u_{12} \quad \text{=====>} \quad u_2 = 1 - u_{12} \\ (7) \quad & 0 = u_5 - u_6 \quad \text{=====>} \quad u_5 = u_6 \\ (8) \quad & 0 = u_2 - u_3, \quad \text{=====>} \quad u_2 = u_3 \\ (9) \quad & 0 = -u_7 + u_8 \quad \text{=====>} \quad u_7 = u_8 \\ (10) \quad & 0 = -u_{11} + u_{12} \quad \text{=====>} \quad u_{11} = u_{12} \\ (11) \quad & 0 = u_1 - u_2 \quad \text{=====>} \quad u_1 = u_2 \\ (12) \quad & 0 = u_4 - u_5 \quad \text{=====>} \quad u_4 = u_5 \\ (13) \quad & 0 = -u_8 + u_9 \quad \text{=====>} \quad u_8 = u_9 \\ (14) \quad & 0 = -u_{10} + u_{11} \quad \text{=====>} \quad u_{10} = u_{11} \end{aligned}$$

somando equação (1) e (2), tem-se

$$\begin{aligned} (1) \quad & -1 = -u_1 + u_{10} \\ (1) \quad & \underline{1 = u_1 + u_{10}} \\ & 0 = 0 + 2u_{10} \quad \text{----->} \quad u_{10} = 0 \end{aligned}$$

Se $u_{10}=0$, então: $u_{12}=u_{11}=u_{10}=u_9=u_8=u_7=0$

Se $u_{12}=0$, então: $u_2 = 1 - 0 \quad \text{----->} \quad \mathbf{u_2=1}$

Se $u_2=1$, então: $\mathbf{u_1=u_2=u_3=1}$

Se $u_3=1$, então: $u_3 = 1 + u_4 \quad \text{=====>} \quad u_4 = u_5 = u_6$

Portanto, $\mathbf{u} = (1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)^T$ que corresponde à sequência de disparo

t_1, t_2, t_3

2.7.3 ANÁLISE DE INVARIANTES.

Os invariantes em uma RdP representam os componentes conservativos e repetitivos da rede. Há conjuntos de lugares e de transições da rede cujo comportamento não se altera durante o seu funcionamento. A identificação e interpretação de cada um destes conjuntos é importante, pois, eles refletem certas propriedades da rede que podem ser de interesse para a análise do sistema modelado.

a) Invariante de Lugar (P-Invariante):

Os componentes conservativos da rede são representados em seus invariantes de lugar, ou seja, são conjuntos de lugares da rede nos quais a soma das marcas é constante durante todo o seu funcionamento.

Para identificar componentes conservativos da rede deve-se partir da equação de estado das RdP:

$$\mathbf{M}' = \mathbf{M} + \mathbf{u}_j^T \mathbf{A}$$

Onde, $\mathbf{M}'$ representa a marcação que se deseja alcançar, $\mathbf{M}$ representa a marcação inicial da rede, $\mathbf{u}_j^T$ representa o vetor contagem de disparos e $\mathbf{A}$ a matriz de incidência da rede.

A situação na qual todos os lugares da rede formam um único componente conservativo é muito improvável na prática. Por outro lado, encontrar sub-conjuntos de lugares nos quais o total de marcas são se altere já é bem mais comum. Para encontrar tais sub-conjuntos deve-se isolar os lugares correspondentes durante a análise. Para isto, deve-se multiplicar a equação de estado por um vetor de posições ponderadas, de n por 1 elementos, denominado $\mathbf{W}$, o qual servirá para produzir o isolamento desejado. Para introduzir o vetor $\mathbf{W}$, sem alterar a equação de estado, deve-se multiplicar todos os seus termos por $\mathbf{W}$, isto é:

$$\mathbf{M}' \cdot \mathbf{W} = \mathbf{M} \cdot \mathbf{W} + \mathbf{u}_j^T \mathbf{A} \cdot \mathbf{W}$$

A solução desta nova equação será obtida ao se igualar a zero seus termos. Para que a rede tenha componentes conservativos a igualdade $\mathbf{M}' \cdot \mathbf{W} = \mathbf{M} \cdot \mathbf{W}$ deve ser verdadeira. O vetor $\mathbf{u}_j^T$ será diferente de 0, já que é a representada do comportamento dinâmico da rede. Sendo assim, pode-se encontrar os componentes conservativos (invariantes de lugar) da rede resolvendo-se a equação:

$$\mathbf{W} \cdot \mathbf{A}^T = \mathbf{0} \quad (4)$$

Para ilustrar a eq.(04), tem-se a RdP da figura 2.12, que tem 4 transições e 5 posições, cuja matriz de incidência é dada por:

$$\mathbf{A} = \begin{bmatrix} -1 & 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 1 \\ 1 & 0 & 0 & -1 & -1 \end{bmatrix}$$

O vetor $\mathbf{W}$ é dado por: $\mathbf{W} = [p_1 \ p_2 \ p_3 \ p_4 \ p_5]$, aplicando a equação 04, tem-se:

$$\begin{bmatrix} p_1 & p_2 & p_3 & p_4 & p_5 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} = 0$$

Como resultado tem-se o seguinte sistema de equações:

$$\begin{array}{ll} -p_1 + p_2 + p_3 = 0 & \rightarrow p_1 = p_2 + p_3 \\ -p_2 + p_4 = 0 & \rightarrow p_2 = p_4 \\ -p_3 + p_5 = 0 & \rightarrow p_3 = p_5 \\ p_1 - p_4 - p_5 = 0 & \rightarrow p_1 = p_4 + p_5 \end{array}$$

Este sistema tem inúmeras soluções possíveis, dependendo da marcação inicial. Como $\mathbf{W}$ é um vetor ponderação, serão utilizados os valores zero e um, resultando nos seguintes componentes conservativos.

Para $p1=0$ e $p3=1$, tem-se $W = [1 \ 0 \ 1 \ 0 \ 1]$, indicando que o conjunto $[p1 \ p3 \ p5]$ é um invariante de lugar da rede da figura 2.12.

Invertendo-se os valores, isto é, para $p1=1$ e $p3=0$, tem-se $W = [1 \ 1 \ 0 \ 1 \ 0]$, indicando o conjunto $[p1 \ p2 \ p4]$ como outro invariante de lugar da rede da figura 2.12.

Figura 2.12 - Exemplo de Invariantes de Lugar

b) Invariante de Transição (T-Invariante):

Os componentes repetitivos são representados em seus invariantes de transição, isto é, são conjuntos de transições da rede que ao serem disparadas em determinada sequência retornam à marcação de partida.

Para identificar os componentes repetitivos da uma Rede de Petri utiliza-se um procedimento análogo ao que é usado para identificar os componentes conservativos, ou seja, para encontrar tais sub-conjuntos deve-se isolar as transições correspondentes durante a análise. Sendo assim, deve-se multiplicar a equação de estado por um vetor de posições ponderadas, de $1 \times n$ elementos, denominado $\mathbf{Y}$, o qual servirá para produzir o isolamento desejado. Para introduzir o vetor $\mathbf{Y}$, sem alterar a equação de estado, deve-se multiplicar todos os seus termos por $\mathbf{Y}$, isto é:

$$\mathbf{M}' \cdot \mathbf{Y} = \mathbf{M} \cdot \mathbf{Y} + \mathbf{u}_j^T \mathbf{A} \cdot \mathbf{Y}$$

A solução desta nova equação será obtida ao se igualar a zero seus termos. Para que a rede tenha componentes repetitivos a igualdade $\mathbf{M}' \cdot \mathbf{Y} = \mathbf{M} \cdot \mathbf{Y}$ deve ser verdadeira. O vetor $\mathbf{u}_j^T$ será diferente de 0, já que é a representada do comportamento dinâmico da rede. Sendo assim, pode-se encontrar os componentes repetitivos (invariantes de transição) da rede resolvendo-se a equação:

$$\mathbf{Y}^T \cdot \mathbf{A} = 0 \quad (5)$$

Para ilustrar a equação (05), tem-se a RdP da figura 2.13, que tem 6 transições e 7, cuja matriz de incidência é dada por:

$$\mathbf{A} = \begin{bmatrix} -1 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

O vetor Y é dado por: $\mathbf{Y} = [t1 \ t2 \ t3 \ t4 \ t5 \ t6]^T$

$$\begin{bmatrix} t1 & t2 & t3 & t4 & t5 & t6 \end{bmatrix} \cdot \begin{bmatrix} -1 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix} = 0$$

Como resultado tem se o seguinte sistema de equações:

$$\begin{aligned}
 t_1 + t_3 &= 0 \\
 t_1 - t_2 = 0 &\text{ -----} \rightarrow t_1 = t_2 = t_3 \\
 t_2 - t_3 = 0 &\text{ -----} \rightarrow t_2 = t_3 \\
 -t_1 + t_2 - t_4 + t_5 &= 0 \\
 -t_4 + t_6 = 0 &\text{ -----} \rightarrow t_4 = t_6 = t_5 \\
 t_4 - t_5 = 0 &\text{ -----} \rightarrow t_4 = t_5 \\
 t_5 - t_6 = 0 &\text{ -----} \rightarrow t_5 = t_6
 \end{aligned}$$

Resumindo tem-se:

$$\begin{aligned}
 t_1 &= t_2 = t_3 \\
 t_4 &= t_5 = t_6
 \end{aligned}$$

Este sistema também tem inúmeras soluções possíveis, dependendo da marcação inicial. Como $\mathbf{Y}$ é um vetor ponderação, serão utilizados os valores zero e um, resultando nos seguintes componentes conservativos:

Para $t_1=1$ e $t_4=0$, tem-se $\mathbf{Y} = [1 \ 1 \ 1 \ 0 \ 0 \ 0]$, indicando que o conjunto $[t_1 \ t_2 \ t_3]$ é um invariante de transição da rede da figura 2.13.

Invertendo-se os valores, isto é, para $t_1=0$ e $t_4=1$, tem-se $\mathbf{Y} = [0 \ 0 \ 0 \ 1 \ 1 \ 1]$, indicando o conjunto $[t_4 \ t_5 \ t_6]$ como outro invariante de transição da rede da figura 2.13.

Figura 2.13 - Exemplo de Invariante de Transição

A tabela 2-2 apresenta um resumo das propriedades das Redes de Petri com as técnicas de análise das RdP com as respectivas propriedades.

Tabela 2-2 - Propriedades x Método de Análise das Redes de Petri

PROPRIEDADE	MÉTODO DE ANÁLISE	
	ÁRVORE DE ALCANÇABILIDADE/COBERTURA	MATRIZ DE INCIDÊNCIA/INVARIANTES
LIMITAÇÃO	<p>UMA RdP É LIMITADA SE E SOMENTE SE O SÍMBOLO ω NÃO APARECE EM SUA ÁRVORE DE COBERTURA.</p> <p>SE ω NÃO APARECE, PELO SIMPLES EXAME DE TODOS OS NÓS DA ÁRVORE, A MAIOR COORDENADA DE TODOS OS ESTADOS FORNECE O k DA k-LIMITAÇÃO.</p>	<p>UMA RdP É LIMITADA SE PARA TODO $p \in P$, $p \in W$, SENDO W UM INVARIANTE DE LUGAR.</p>
SEGURANÇA	<p>UMA RdP É SEGURA SE O SÍMBOLO ω NÃO APARECE EM SUA ÁRVORE DE COBERTURA E A MAIOR COORDENADA DE TODOS OS ESTADOS FORNECE O $k=1$</p>	NÃO É POSSÍVEL DETERMINAR
CONSERVAÇÃO	<p>PARA QUE UMA RdP SEJA CONSERVATIVA A SOMA PONDERADA DAS COORDENADAS DOS VETORES X EM TODOS OS NÓS SEJA CONSTANTE, ISTO É: CHAMANDO DE w_i, $i=1$ a n, OS ELEMENTOS NÃO NULOS DO VETOR PESO, ASSOCIADOS A n POSIÇÕES, PODE-SE ESCREVER, PARA CADA NÓ DA REDE:</p> $w_1 \cdot x_1 + w_2 \cdot x_2 + \dots + w_n \cdot x_n = k$ <p>onde k é uma constante</p>	<p>PARA QUE UMA RdP SEJA CONSERVATIVA DEVE SATISFAZER A SEGUINTE CONDIÇÃO:</p> $W \cdot A = 0$ <p>onde w é o vetor peso, associado a n posições e A é matriz de incidência da RdP.</p>
VIVACIDADE	<p>UMA RdP É VIVA SE TODAS AS SUAS TRANSIÇÕES FOREM HABILITADAS AO MENOS UMA VEZ, PARA UMA DADA MARCAÇÃO INICIAL (1)</p>	<p>UMA RdP É VIVA SE FOR LIMITADA E HOUVER INVARIANTES DE TRANSIÇÃO</p>
IMPASSE (DEADLOCK)	<p>QUANDO UM NÓ DA ÁRVORE NÃO HABILITAR MAIS NENHUMA TRANSIÇÃO, ISTO INDICA QUE APÓS n DISPAROS DE TRANSIÇÕES O SISTEMA NÃO EVOLUI MAIS, ENTRANDO EM DEADLOCK</p>	NÃO É POSSÍVEL DETERMINAR
ALCANÇABILIDADE	<p>PARA QUE ESTADO M SEJA ALCANÇÁVEL, É NECESSÁRIO APARECER EM TODOS OS RAMOS DA ÁRVORE DE ALCANÇABILIDADE (1)</p>	<p>PARA QUE ESTADO M SEJA ALCANÇÁVEL, É NECESSÁRIO QUE EXISTA A SOLUÇÃO u, DE ELEMENTOS INTEIROS NÃO NEGATIVOS, PARA A EQUAÇÃO:</p> $M' = M + u^T \cdot A.$
REVERSIBILIDADE	<p>UMA RdP É REINICIALIZÁVEL (REVERSÍVEL) SE NÃO EXISTIR NÓS TERMINAIS NA ÁRVORE E TODOS OS NÓS DUPLICADOS FOREM IGUAIS AO NÓ RAIZ (1)</p>	<p>UMA RdP É REINICIALIZÁVEL (REVERSÍVEL) SE HOUVER INVARIANTES DE TRANSIÇÃO</p>
(1) CONDIÇÃO NECESSÁRIA, MAS NÃO SUFICIENTE, POIS ATRAVÉS DA ÁRVORE DE ALCANÇABILIDADE NÃO SE SABE QUAL A SEQUÊNCIA DE DISPARO.		

Fonte: MURATA,1989;PETERSON,1981

2.7.4 ANÁLISE DAS RdP POR SIMULAÇÃO COMPUTACIONAL

Uma outra técnica de modelagem e verificação das propriedades das Redes de Petri é a simulação por meio de recursos computacionais.

Entre as vantagens da simulação computacional podem-se citar as seguintes:

- a) Efeitos das animações e gráficos ajudam no aprendizado e entendimento do sistema simulado.
- b) Permite manter um maior controle sobre o experimento, o que muitas vezes não é possível no sistema real.
- c) Permite estudar o sistema durante o longo período de tempo simulado.

Na URL <http://www.informatik.uni-hamburg.de/TGI/PetriNets/>, acessado em 05/06/2006, que é um site sobre Conferências Internacionais e Teoria das Redes de Petri, encontra-se uma lista completa de vários simuladores de Redes de Petri disponíveis comercialmente e gratuitamente. Um dos simuladores para Rede de Petri mais empregados atualmente é o simulador *Visual Object Net*

2.7.4.1 FERRAMENTA DE SIMULAÇÃO VISUAL OBJECJ NET ++ (VON)

O Visual Object Net ++ é um software de simulação baseado em Redes de Petri (discretas e contínuas) desenvolvido por Rainer Drath, na *Ilmenau University of Technology*, Alemanha, Departamento Controle Automático e Sistemas de Engenharia, acessado em 05/06/2006 na URL: http://www.systemtechnik.tu-ilmenau.de/~drath/visual_e.htm

É um software livre e de fácil utilização, possui um menu com botões básicos similares ao dos produtos Microsoft Windows e um menu específico com componentes particulares das Redes de Petri. A figura 2.12 ilustra uma tela do mesmo como exemplo.

Figura 2.14 - Tela Principal do VON

Figura 2.15 - Menu de Elementos de RdP do VON

Ao mesmo tempo em que um elemento gráfico é inserido na janela de edição, um elemento correspondente a ele é inserido na janela de propriedades – através desta janela o usuário pode alterar as propriedades dos objetos.

Figura 2.16 - Janela de Propriedades de Lugares(posição) do VON

A figura 2.14 ilustra a janela de propriedades de lugares, onde: *variable* é o nome do lugar(posição), *startvalue* é o número de marcas iniciais e *value* é a capacidade do lugar(posição).

Figura 2.17 - Janela de Propriedades de Transições do VON

A figura 2.15 ilustra a janela de propriedades de transições, onde: *name* é o nome da transição, *weight* é o peso dos arcos e *firingtime* é o tempo de disparo.

Figura 2.18 - Janela de Propriedades de Arcos do VON

A figura 2.16 ilustra a janela de propriedades de arcos, onde: *weight* é o peso dos arcos e *actual value* é similar a *weight*.

3 PROGRAMAÇÃO E UTILIZAÇÃO DE CONTROLADORES LÓGICOS PROGRAMÁVEIS

3.1 CONTROLADOR LÓGICO PROGRAMÁVEL (CLP)

3.1.1 INTRODUÇÃO

De acordo com a Associação Brasileira de Normas Técnicas (ABNT), o Controlador Lógico Programável (CLP) é um equipamento eletrônico digital com hardware e software compatíveis com aplicações industriais. A *National Electrical Manufacturers Association* (NEMA), de acordo com a *International Electrotechnical Commission* (IEC), define CLP como: “suporte eletrônico digital para armazenar instruções de funções específicas, como de lógica, seqüencialização, contagem e aritméticas, todas dedicadas ao controle de máquinas e processos”.

O Controlador Lógico Programável pode, também, ser definido como um dispositivo de estado sólido, com memória programável para armazenamento de instruções para controle lógico programável e pode executar funções equivalentes as de um painel de relés ou de um sistema de controle lógico, também realizando operações lógicas e aritméticas, manipulação de dados e comunicação em rede, sendo utilizado no controle de sistemas automatizados.

A figura 3.1 apresenta uma aplicação geral de um CLP, onde as informações de entrada, fornecidas pelos dispositivos de entrada, são analisadas, as decisões são tomadas, os comandos ou acionamentos são enviados as saída, por meio dos dispositivos de saída que atuam no sistema automatizado.

Figura 3.1 - Aplicação geral do controlador lógico programável.

3.1.2 ARQUITETURA BÁSICA DO CLP

A figura 3.2 ilustra a estrutura básica de um CLP. Esta compreende a CPU, módulo de entrada e saída e fonte de alimentação.

Figura 3.2 - Estrutura básica de um CLP (Fonte: SILVEIRA, SANTOS, 1999).

A seguir, as principais características desses elementos são apresentadas.

- a) CPU (Unidade Central de Processamento). A figura 3.3 compreende a estrutura básica da CPU que é composta pelo processador, sistema de memória e barramento de dados, controle e endereço.

Figura 3.3 - Estrutura básica da CPU (Fonte: SILVEIRA, SANTOS, 1999).

a1) Processador

Os processadores podem ser, por exemplo, do tipo microprocessador/controlador convencional 80286, 80386, 8051, ou até mesmo um processador dedicado, um DSP (Processador Digital de Sinais). Existem CPUs que possuem processamento paralelo, no qual dois ou mais processadores executam o programa de aplicação, comparando os resultados obtidos após cada ciclo de varredura.

O processador é responsável pelo gerenciamento total do sistema, controlando os barramentos de endereços, de dados e de controle, interpreta e executa as instruções do programa de aplicação, controla a comunicação com dispositivos externos e verifica a integridade de todo o sistema realizando relatórios ou diagnósticos do sistema operacional.

a2) Sistema de memória

O sistema de memória da CPU é composto por: (i) memória do sistema de operação, onde é armazenado o programa de execução desenvolvido pelo fabricante, e que determina como o sistema deve operar, incluindo a execução dos programas do

usuário, controle de serviços periféricos, atualização dos módulos de entrada/saída etc. (ii) memória de aplicação ou Memória de usuário, onde o programa desenvolvido pelo usuário chamado de programa de aplicação é armazenado.

Juntamente com o programa de aplicação, são armazenados os dados do sistema em uma tabela para realização dos controles dos módulos de entrada/saída utilizados. Cada ponto de entrada/saída conectado aos módulos tem um endereço específico na tabela de dados, o qual é acessado pelo programa de aplicação.

b) Módulos de Entrada/Saída

Estes módulos realizam a comunicação entre a CPU e os dispositivos externos por meio das entradas e saídas dos módulos, garantindo isolamento e proteção à CPU. Os módulos de entrada recebem os sinais dos dispositivos tais como sensores, chaves e transdutores e convertem esses sinais em níveis adequados para serem processados pela CPU. Os módulos de saída enviam os sinais de controle aos dispositivos externos tais como motores atuadores e sinalizadores. Esses sinais são resultantes da lógica de controle, pela execução do programa de aplicação, ou podem ser forçados pelo usuário, independente da lógica de controle.

Os módulos são classificados como Discretos (Digitais) ou Analógicos. Os módulos discretos de entrada/saída são utilizados em sistemas sequenciais e na maioria das aplicações com CLPs. A figura 3.4 mostra alguns elementos de entrada/saída digital.

Os módulos analógicos tratam os sinais analógicos que são utilizados pelos sistemas contínuos. Os módulos analógicos de entrada convertem sinais analógicos provenientes dos dispositivos de entrada (transdutor, conversor, termopar) em sinais digitais por meio de conversores analógicos/digitais, disponibilizando-os adequadamente ao barramento da CPU. Os módulos analógicos de saída convertem sinais digitais, disponíveis no barramento da CPU, em sinais analógicos por meio de conversor digital/analógico, enviando-os aos dispositivos de saída (driver,

amplificador). A figura 3.5 ilustra o funcionamento dos módulos analógicos de entrada e de saída. Na figura 3.6 são mostrados alguns elementos de entrada e saída analógicos.

Figura 3.4 - Elementos de entrada e saída discretos (Fonte: SILVEIRA, SANTOS, 1999).

**Figura 3.5 – Módulos de entrada e saída analógica
(Fonte: SILVEIRA, P. R., SANTOS W. E, 1999).**

Figura 3.6 - Elementos de entrada e saída analógicos (Fonte: SILVEIRA, SANTOS, 1999).

c) Fonte de Alimentação

Dispositivo responsável pela tensão de alimentação fornecida à CPU e aos Módulos (circuitos) de entrada/saída. Em alguns casos proporciona saída auxiliar de baixa corrente.

3.1.3 OPERAÇÃO BÁSICA DO CLP

A CPU controla todas as ações do CLP executando a leitura das condições e estados dos dispositivos por meio dos módulos de entrada/saída. Essas condições são armazenadas em memória para serem processadas pelo programa de aplicação desenvolvido pelo usuário e armazenado na memória no CLP. O processador atualiza os status dos dispositivos de saída por meio dos módulos de saída, realizando a lógica de controle, para garantir o ciclo de varredura. A programação pode ser feita através de um programador manual ou com software de programação no computador para posterior transferência.

A figura 3.7 representa um circuito convencional, contatos elétricos, e a figura 3.8 representa a ligação dos dispositivos de entrada (botões B0 = X0 e B1 = X1) aos módulos de entrada e os dispositivos de saída (lâmpada sinalizadora L0 = Y0) aos módulos de saída.

Figura 3.7 - Circuito convencional - contatos elétricos.

Figura 3.8 - Ligação elétrica por meio de CLP.

Onde:

B0/B1 = botoeira liga-desliga – contato normalmente fechado (NF)

LO = lâmpada sinalizadora.

F1/F2 – alimentação Elétrica.

A figura 3.9 apresenta um programa de CLP escrito na linguagem de programação Ladder que implementa a mesma lógica do circuito elétrico da figura 3.7. O programa de aplicação determina o acionamento da saída em função das entradas. Qualquer alteração desejada nesta lógica é realizada por meio de alterações no programa, permanecendo as mesmas conexões nos módulos de entrada/saída.

Figura 3.9 - Linguagem de Programação Ladder da Figura 3.8.

3.2 LINGUAGEM DE PROGRAMAÇÃO (LADDER)

A linguagem de programação que será utilizada neste trabalho de pesquisa para implementação nos CLPs é a Ladder. Essa é uma linguagem gráfica baseada em símbolos semelhantes aos contatos de relés e bobinas nos esquemas elétricos. Por sua semelhança com sistemas de controle a relés é a linguagem de programação muito utilizada para programação de CLPs.

A Norma IEC 61131 (inicialmente 1131), de agosto de 1992, sobre Controladores Lógicos Programáveis, apresenta atualmente oito partes (IEC 61131-1 a IEC 61131-8). A terceira parte (IEC 61131-3) aborda as linguagens de programação e define, também, a estrutura de um projeto, os tipos de dados e a organização interna do programa. As cinco linguagens de programação definidas pela IEC 61131-3, são: as Textuais: lista de instruções (*Instruction List-IL*) e texto estruturado (*Structured Text-ST*) e as gráficas: Ladder (Diagramas de Relés-LD), diagrama de blocos de função (*Function Block Diagrama – FBD*) e Linguagem de Diagrama Seqüencial (*Seqüencial Flow Chart – SFC*). O gráfico 3.1 apresenta a porcentagem de emprego de cada uma destas linguagens de programação.

Gráfico 3.1 - Linguagem de Programação em uso (Fonte: Control Engineering-12/05).

3.2.1 FUNDAMENTOS DE PROGRAMAÇÃO EM LADDER

A figura 3.10 ilustra os principais componentes da linguagem de programação Ladder.

Figura 3.10 - Componentes da programação em linguagem Ladder

(Fonte: JUNIOR, PEREIRA, 2006).

A linguagem de programação Ladder possui as seguintes regras de conexão:

- a) As linhas verticais das extremidades à direita e à esquerda chamam-se linhas mestres; na da esquerda são conectadas as instruções de entrada e na da direita são conectadas somente as instruções de saída, que deve ocupar somente a última coluna à direita;
- b) As instruções de entrada e as de saídas são conectadas através de linhas horizontais. As linhas horizontais são interligadas através de linhas verticais e não se permitem várias linhas numa única coluna;

A Linguagem de programação Ladder tem regra para o processamento das instruções, e a instrução de saída será sempre executada se houver continuidade da linha lógica, por exemplo, uma instrução de saída do tipo OTE será sempre

executada caso haja continuidade lógica da linha em que a mesma está conectada e a variável por ela endereçada será colocada em 1.

A quantidade de colunas e linhas, ou instruções, que cada linha pode ter é determinada pelo fabricante do CLP, podendo variar conforme a CPU utilizada. Em geral, este limite não representa uma preocupação ao usuário durante o desenvolvimento do programa de aplicação, pois os softwares de programação indicam se tal quantidade foi ultrapassada durante a compilação do programa de aplicação.

Cada elemento da lógica de controle representa uma instrução da linguagem Ladder, sendo alocada em um endereço específico e consumindo uma quantidade determinada de memória disponível para armazenamento do programa de aplicação, conforme a CPU utilizada.

3.2.1.1 INSTRUÇÕES BÁSICAS EM LINGUAGENS LADDER

As instruções básicas da maioria dos CLPs podem ser agrupadas em sete grupos: lógica de relé ou instrução de bit, temporização e contagem, aritméticas, manipulação de dados, controle de fluxo, transferência de dados e avançadas.

Neste trabalho de pesquisa são indicadas as principais instruções de bit, temporização e contagem.

Uma instrução de bit pode ser de entrada ou de saída. Durante a execução de uma instrução de entrada o estado de um bit em um determinado endereço é examinado. Durante a execução de uma instrução de saída de bit o estado de um bit de um determinado endereço é alterado para 0 ou 1 conforme haja ou não continuidade lógica da linha a que a instrução está relacionada.

Algumas das principais instruções de bit (de entrada e de saída), temporização e contagem são:

Tabela 3-1 - Principais Instruções da Linguagem Ladder

INSTRUÇÃO	TABELA-VERDADE	SÍMBOLO GRÁFICO						
<div>XIC</div> <div>Examinar se energizado</div>	<table><tr><th>BIT</th><th>INSTRUÇÃO</th></tr><tr><td>1</td><td>VERDADEIRA</td></tr><tr><td>0</td><td>FALSA</td></tr></table>	BIT	INSTRUÇÃO	1	VERDADEIRA	0	FALSA	<div>endereço</div> <div> </div> <div>Número do bit</div>
BIT	INSTRUÇÃO							
1	VERDADEIRA							
0	FALSA							
<div>XIO</div> <div>Examinar se desenergizado</div>	<table><tr><th>BIT</th><th>INSTRUÇÃO</th></tr><tr><td>1</td><td>FALSA</td></tr><tr><td>0</td><td>VERDADEIRA</td></tr></table>	BIT	INSTRUÇÃO	1	FALSA	0	VERDADEIRA	<div>endereço</div> <div> </div> <div>Número do bit</div>
BIT	INSTRUÇÃO							
1	FALSA							
0	VERDADEIRA							
<div>OTE</div> <div>Energizar saída</div>	<table><tr><th>Linha de instrução</th><th>INSTRUÇÃO</th></tr><tr><td>0</td><td>FALSA</td></tr><tr><td>1</td><td>VERDADEIRA</td></tr></table> <div>Sendo a instrução verdadeira a saída endereçada será energizada</div>	Linha de instrução	INSTRUÇÃO	0	FALSA	1	VERDADEIRA	<div>endereço</div> <div> </div> <div>Número do bit</div>
Linha de instrução	INSTRUÇÃO							
0	FALSA							
1	VERDADEIRA							
<div>OTL</div> <div>Energizar saída com retenção</div>	<table><tr><th>Linha de instrução</th><th>INSTRUÇÃO</th></tr><tr><td>0</td><td>FALSA</td></tr><tr><td>1</td><td>VERDADEIRA</td></tr></table> <div>Sendo a instrução verdadeira a saída endereçada somente será desabilitada caso a instrução OTU seja acionada.</div>	Linha de instrução	INSTRUÇÃO	0	FALSA	1	VERDADEIRA	<div>endereço</div> <div> </div> <div>Número do bit</div>
Linha de instrução	INSTRUÇÃO							
0	FALSA							
1	VERDADEIRA							
<div>OTU</div> <div>Desabilita uma saída Habilitada por uma instrução OTL.</div>	<table><tr><th>Linha de instrução</th><th>INSTRUÇÃO</th></tr><tr><td>0</td><td>FALSA</td></tr><tr><td>1</td><td>VERDADEIRA</td></tr></table>	Linha de instrução	INSTRUÇÃO	0	FALSA	1	VERDADEIRA	<div>endereço</div> <div> </div> <div>Número do bit</div>
Linha de instrução	INSTRUÇÃO							
0	FALSA							
1	VERDADEIRA							
<div>TON</div> <div>TEMPORIZADOR</div>	<table><tr><th>Linha de instrução</th><th>INSTRUÇÃO</th></tr><tr><td>0</td><td>FALSA</td></tr><tr><td>1</td><td>VERDADEIRA</td></tr></table> <div>Inicia a contagem nos intervalos da base de tempo selecionada quando a condição da linha se torna verdadeira. O bit EN é colocado no estado lógico 1 cada vez que a instrução é acionada. O bit DN é colocado no estado lógico 1 quando o valor ACCUM for igual ao valor PRESET.</div>	Linha de instrução	INSTRUÇÃO	0	FALSA	1	VERDADEIRA	<div> </div>
Linha de instrução	INSTRUÇÃO							
0	FALSA							
1	VERDADEIRA							
<div>CTU</div> <div>CONTADOR CRESCENTE</div>	<table><tr><th>Linha de instrução</th><th>INSTRUÇÃO</th></tr><tr><td>0</td><td>FALSA</td></tr><tr><td>1</td><td>VERDADEIRA</td></tr></table> <div>Incrementa o valor ACCUM a cada transição de falsa para verdadeira da condição lógica da linha em que a instrução está inserida.</div>	Linha de instrução	INSTRUÇÃO	0	FALSA	1	VERDADEIRA	<div> </div>
Linha de instrução	INSTRUÇÃO							
0	FALSA							
1	VERDADEIRA							

(Fonte: JUNIOR, PEREIRA, 2006.)

3.2.2 IMPLANTAÇÃO DA LÓGICA DE CONTROLE

A figura 3.11 apresenta um diagrama de blocos que mostra como resolver em etapas um problema de lógica.

Figura 3.11 - Organização do raciocínio na solução de problemas de lógica.

A análise para a criação de programas no CLP é predominantemente baseada na lógica binária (álgebra de Boole). O sistema binário é um sistema de numeração que consta de dois valores, 0 e 1. Cada dígito de um número representado no sistema binário recebe o nome de bit, de maneira tal que cada bit pode tomar o valor 0 ou 1. A notação empregada neste trabalho de pesquisa é $\bar{A}$ para negar A, “.” e “+” para indicar “e” e “ou”, respectivamente. Os teoremas booleanos são [16]:

1. $\overline{\bar{A}} = A$	5. $A + 1 = 1$	9. $A \cdot \bar{A} = 0$
2. $A \cdot 0 = 0$	6. $A + A = A$	10. $A \cdot B + A \cdot C = A \cdot (B + C)$
3. $A + 0 = A$	7. $A \cdot A = A$	11. $A + \bar{A} \cdot B = A + B$
4. $A \cdot 1 = A$	8. $A + \bar{A} = 1$	

Os teoremas booleanos, juntamente com os teoremas de DE MORGAN permitem minimizar as relações lógicas entre as variáveis. Os teoremas de De Morgan são:

1. $\overline{A \cdot B} = \bar{A} + \bar{B}$
2. $\overline{A + B} = \bar{A} \cdot \bar{B}$

A tabela 3-2 a seguir considera as funções lógicas e suas implementações utilizando a linguagem Ladder.

Tabela 3-2 - Implementação da lógica booleana/De Morgan em linguagem Ladder.

FUNÇÃO	TABELA VERDADE DA LINHA DE PROGRAMAÇÃO	LÓGICA BOOLEANA / DE MORGAN PARA LINHA DE PROGRAMAÇÃO	LINGUAGEM LADDER															
NÃO (NOT)	<table><tr><td>A</td><td>Y</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Y	0	1	1	0	$Y=\bar{A}$
A	Y																	
0	1																	
1	0																	
E (AND)	<table><tr><td>A</td><td>B</td><td>Y</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	0	1	0	0	1	1	1	$Y = A.B$
A	B	Y																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OU (OR)	<table><tr><td>A</td><td>B</td><td>Y</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	1	$Y = A+B$
A	B	Y																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NÃO E (NAND)	<table><tr><td>A</td><td>B</td><td>Y</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	1	1	0	1	1	1	0	$Y = \overline{A.B} = \bar{A}+\bar{B}$
A	B	Y																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NÃO OU (NOR)	<table><tr><td>A</td><td>B</td><td>Y</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	0	$Y = \overline{A+B} = \bar{A}.\bar{B}$
A	B	Y																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
OU EXCLUSIVO (OR EXCLUSIVO)	<table><tr><td>A</td><td>B</td><td>Y</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	0	$Y = A.B+A.\bar{B}$ OU $Y = A\oplus B$
A	B	Y																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
NÃO OU EXCLUSIVO (NOR EXCLUSIVO)	<table><tr><td>A</td><td>B</td><td>Y</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	0	$Y = \overline{A.B+A.\bar{B}}$ $Y = (\bar{A}+B).(A+\bar{B})$
A	B	Y																
0	0	1																
0	1	0																
1	0	0																
1	1	0																

(Fonte: SILVEIRA, SANTOS, 1999)

4 METODOLOGIA DE APLICAÇÃO DE REDES DE PETRI PARA AUTOMAÇÃO DE SISTEMAS INDUSTRIAIS COM CLP – MARPASI.

Neste capítulo, é apresentada a metodologia denominada “MARPA SI” - Metodologia de Aplicação das Redes de Petri para Automação de Sistemas Industriais.

A “MARPA SI” é uma ferramenta de engenharia para auxílio na programação de CLPs.

4.1 ENGENHARIA DE REQUISITOS EM SISTEMAS AUTOMATIZADOS.

O desenvolvimento de um sistema de automação industrial demanda uma especificação concisa e muito acurada no que diz respeito ao comportamento desejado do sistema. É, portanto, essencial que os métodos e a representação das especificações permitam expressar sem ambigüidades o comportamento do sistema, bem como validar e verificar sua consistência.

Embora a análise de requisitos não seja o foco do presente trabalho de pesquisa, é importante para o desenvolvimento de sistemas de automação conhecer como devem ser obtidos e analisados os requisitos necessários para iniciar a modelagem dos mesmos e poder efetuar posteriormente sua análise.

Neste contexto, a engenharia de requisitos é vista como um processo sistemático e disciplinado de levantamento de necessidades e de funcionalidades de um sistema, com a finalidade da especificação precisa de requisitos operacionais de desempenho e de qualidade.

Os requisitos devem ser definidos nas fases iniciais do desenvolvimento de sistemas como sendo uma especificação do que deve ser implementado.

A aplicação deste processo varia entre as organizações, porém a maioria envolve a elicitação, análise, especificação e validação e gerenciamento de requisitos. Elicitação, análise, especificação e validação são atividades que ocorrem nessa sequência. No entanto, são atividades que se relacionam entre si, e podem em determinados momentos assumir seqüências diferentes. A atividade de gerenciamento ocorre em paralelo a essas atividades.

A Engenharia de Requisitos, enquanto um processo de “construção” dos requisitos, é um processo iterativo, em que a seqüência de execução de suas atividades pode ocorrer diversas vezes.

A Figura 4.1 apresenta um modelo espiral do processo de Engenharia de Requisitos fundamentado nas atividades de elicitação, análise, especificação e validação dos requisitos. A elicitação dos requisitos expande a declaração informal dos requisitos, oferecendo subsídios para análise dos requisitos que resultará nos requisitos acordados entre usuários e desenvolvedores. Os requisitos acordados devem ser especificados na etapa de especificação, gerando os esboços da documentação dos requisitos. Os esboços de especificação devem ser validados gerando a especificação de requisitos final, juntamente com o relatório de validação.

Figura 4.1 - Modelo espiral do processo de Engenharia de Requisitos

Fonte: KOTONYA, SOMMERVILLE, 1998

A figura 4.2 ilustra o processo de engenharia de requisitos, destacando as principais entradas e saídas do processo [31].

Figura 4.2 - Entradas e Saídas do processo de engenharia de requisitos [31]

Fonte: KOTONYA, SOMMERVILLE,1998

4.2 DESENVOLVIMENTO DA MARPASI

A MARPASI proposta neste trabalho de pesquisa visa orientar de modo eficaz a programação de CLPs, tendo as Redes de Petri como ferramenta auxiliar de modelagem e análise operacional do sistema, e a linguagem de programação Ladder como ferramenta de implementação em hardware do sistema modelado. Esta metodologia deve, assim, organizar e estruturar as etapas de trabalho de forma racional e sistemática.

A MARPASI é composta de seis etapas, ilustradas na figura 4.3.

Nesta metodologia, as etapas 1 e 2 consistem, na definição concreta do sistema, na análise das necessidades requeridas e na definição do sistema apropriado para a solução do sistema.

A etapa 3 é a elaboração do modelo do sistema por meio das Redes de Petri.

A etapa 4 constitui a análise dos modelos, verificando se estes foram corretamente gerados e validando-os. No contexto da figura 4.3, pode-se considerar esta etapa como parte do processo de verificação de propriedades qualitativas e de avaliação de desempenho.

A etapa 5 consiste na conversão do sistema modelado por meio das Redes de Petri para linguagem de programação Ladder.

A etapa 6 é a implementação do sistema de acordo com as informações geradas nas etapas anteriores. No contexto da figura 4.3, pode-se considerar esta etapa como a fase de execução.

4.2.1 ETAPA 1: DEFINIÇÃO DO SISTEMA

Esta é a etapa de definição do sistema de automação. Nesta etapa identifica-se o porquê do desenvolvimento do sistema.

Esta etapa consiste na elicitación de requisitos, é a primeira atividade a ser desenvolvida. Na elicitación busca-se descobrir os requisitos do sistema, normalmente obscuros, vagos e confusos no início do desenvolvimento de um sistema.

Nessa etapa, usuários e desenvolvedores trabalham em conjunto para definir o problema a ser solucionado, enfocando principalmente os serviços que o sistema deve oferecer. No entanto, é comum os usuários não saberem exatamente o que eles desejam que seja implementado no sistema de automação, o que pode fazer com que os requisitos definidos inicialmente não reflitam as reais necessidades dos usuários.

Assim, esta atividade não envolve apenas perguntar ao usuário o que ele deseja. Ela requer uma análise cuidadosa da organização onde o sistema será implantado, uma análise do domínio da aplicação e uma análise dos processos de negócios onde o sistema será utilizado.

A elicitación de requisitos é uma atividade complexa, principalmente devido ao seu alto grau de incerteza. A elicitación de requisitos, realizada de forma efetiva, deve abordar quatro dimensões:

- a) **Entendimento do domínio da aplicação:** significa conhecer a área onde o sistema é aplicado de uma forma geral. Este entendimento exige conhecimentos gerais sobre a aplicação em questão.
- b) **Entendimento do problema:** significa conhecer os detalhes específicos do problema de um cliente em particular.
- c) **Entendimento do negócio:** normalmente sistemas de automação contribuem de alguma forma com os objetivos e missão da organização onde eles estão inseridos. O entendimento do negócio significa conhecer como o sistema

interage e afeta os negócios da organização, e que tipo de contribuição pode proporcionar.

- d) **Entendimento das necessidades e restrições das pessoas envolvidas no sistema:** para obter-se este tipo de entendimento, é necessário conhecer os processos que o sistema deverá suportar, pois boa parte destes processos são realizados pelas pessoas envolvidas no sistema.

Há diversas técnicas para elicitação dos requisitos [32] e pode-se usar uma ou mais de uma técnica para elicitação de requisitos em sistemas de automação industrial. A tabela 4-1 apresenta um resumo de algumas destas técnicas.

Tabela 4-1 - Resumo de algumas Técnicas de Elicitação de requisitos

TÉCNICA	DESCRIÇÃO	VANTAGEM	DESVANTAGEM
INTROSPECÇÃO	<p>A introspecção é o ato de “imaginar” como deveria ser um sistema para atender a um determinado problema.</p> <p>Ela leva em consideração um ponto de vista particular, por exemplo, o ponto de vista do desenvolvedor do sistema, que procura imaginar que tipo de sistema ele desejaria ter para realizar as tarefas que o usuário real irá executar.</p>	Permite a elaboração de uma concepção de sistema rapidamente.	Está fortemente vinculada a experiência do desenvolvedor em relação ao problema a ser resolvido.
OBSERVAÇÃO	O princípio básico desta técnica é obter informações a partir da observação de tarefas executadas pelo usuário (ou por um grupo de usuários), com um mínimo de interferência da parte do desenvolvedor.	Os requisitos podem ser obtidos sem interferência de comunicação, uma vez que estarão sendo percebidos pelo observador no ambiente do próprio usuário. A observação ocorre no cotidiano do usuário, onde as situações de trabalho (rotinas) que envolvem as pessoas que vão fazer uso do sistema emergem naturalmente.	O período de observação para cobrir as principais rotinas de trabalho do usuário pode ser longo, uma vez que muitas situações importantes podem ocorrer apenas ocasionalmente.
QUESTIONÁRIO	O questionário é um instrumento que pode ser útil na elicitação de requisitos quando se deseja obter informações de um grande número de pessoas. Outro aspecto interessante do uso de questionários, é que eles podem ser preparados de tal forma que suas respostas possam ser facilmente tabuladas, possibilitando uma análise estatística dos dados obtidos.	Pode trazer informações pertinentes ao sistema estudado vinda de um grande número de pessoas envolvidas no mesmo. Facilita uma análise estatística das informações obtidas.	O grau de flexibilidade que o questionário dá para as respostas dos envolvidos muitas vezes é pequeno, tendendo a ser rígido, dificultando a obtenção de informações mais subjetivas. Muitas vezes as pessoas não se sentem à vontade de colocar suas opiniões por escrito.
ENTREVISTA	A entrevista é uma técnica muito comum utilizada em processos de elicitação de requisitos. Existem basicamente dois tipos de entrevista: entrevista fechada e entrevista aberta. Estes dois tipos de entrevista, normalmente, ocorrem de forma individual e privada.	É flexível na obtenção das informações, sendo adequada para se obter informações de caráter subjetivo. Aproxima o engenheiro de requisitos do usuário do sistema, fazendo com que o usuário se sinta participativo no processo de desenvolvimento do sistema.	Pode demandar muito tempo se muitas pessoas precisarem ser entrevistadas, uma vez que a entrevista normalmente ocorre de forma individual, com cada usuário do sistema. A tabulação das informações obtidas a partir das entrevistas, principalmente na de tipo aberto, costuma ser trabalhosa.
ANÁLISE DE PROTOCOLO	A análise de protocolo é uma técnica de elicitação de requisitos que pode ser aplicada a partir da narração de uma tarefa realizada pelo usuário. Enquanto o usuário está executando a tarefa ele vai falando o que está fazendo. Os proponentes desta técnica acreditam que este processo pode ser considerado como uma verbalização direta de processos cognitivos específicos.	Duas formas de comunicar informações sobre o sistema ocorrem simultaneamente, o fazer e o falar o que está fazendo. Requisitos podem ser capturados e ajustados comparando as duas formas de comunicação. O fazer pode completar o falar e vice-versa.	O usuário pode se sentir inibido de falar enquanto faz, podendo também se desconcentrar no que está fazendo enquanto tenta explicar verbalmente.
PROTOTIPAGEM	construção de um protótipo do sistema a ser implementado. O protótipo é frequentemente usado para elicitar e validar requisitos do sistema (31). Um requisito essencial para um protótipo, é que deve ser possível desenvolvê-lo de forma rápida, tal que ele possa ser utilizado durante o processo de Engenharia de Requisitos. O protótipo não é o sistema real, mas apenas uma simulação do sistema que será construído de fato.	O protótipo oferece uma ajuda efetiva na comunicação entre usuário e engenheiro de requisitos, devido a seu forte apelo visual.	Existe o risco de se usar a estrutura do protótipo como base de implementação da primeira versão do sistema.
CASOS DE USO	A utilização de casos de uso (<i>use cases</i>) para a elicitação de requisitos do sistema vem recebendo interesse crescente da comunidade de Engenharia de Requisitos, e sendo bem aceita por muitos metodologistas. Esta abordagem enfatiza a importância da captura dos requisitos do sistema do ponto de vista dos atores que interagem com ele.	Uma vez identificados os casos de uso, os requisitos do sistema estão praticamente definidos. Oferece uma estrutura de organização para o processo de elicitação e análise dos requisitos. Os requisitos podem ser apresentados para o usuário por meio de diagramas de fácil entendimento, o que facilita o processo de comunicação entre usuário e engenheiro de requisitos.	Cada caso de uso precisa ser descrito detalhadamente em separado, uma vez que sua representação pictórica com o diagrama de casos de uso oferece poucas informações sobre os requisitos.

Fonte: MARTINS, 2001.

Basicamente as informações geradas nesta etapa são:

- Objetivo que a organização e usuários esperam que o sistema execute;
- Área de atuação do sistema;
- Limites de atuação do sistema;
- Benefícios esperados pela organização e usuários.

Estas informações são o ponto de partida para efetuar a análise, especificação e validação dos requisitos que será feito na etapa 2. A figura 4.4 mostra de maneira esquemática as entradas e saídas da etapa 1.

Figura 4.4 - Etapa 1: Definição do Sistema

4.2.1.1 DIFICULDADES NA ELICITAÇÃO DE REQUISITOS

Identificar corretamente os requisitos do sistema não é uma tarefa trivial, em parte devido à natureza abstrata do sistema. Para entender melhor as dificuldades enfrentadas na elicitação dos requisitos pode-se analisá-las a partir de dois grandes grupos: dificuldades acidentais e essenciais.

As dificuldades acidentais são aquelas oriundas da falta de controle sobre aquilo que precisa ser construído, dentre as quais destacam-se: pouco esforço despendido no levantamento de informações junto ao usuário, documentação pobre sobre os requisitos obtidos, pouca revisão dos requisitos obtidos, especificações

incorretas dos requisitos e tendência de iniciar logo o processo de desenvolvimento do sistema.

As dificuldades essenciais são aquelas inerentes à elicitación dos requisitos do sistema, dentre as quais destacam-se: dificuldade do usuário em saber efetivamente o que ele quer de um sistema de automação, dificuldade de comunicação entre usuário e desenvolvedor e a natureza mutante dos requisitos.

As dificuldades acidentais podem ser consideradas como mais fáceis de serem superadas. A adoção de um processo sistemático que oriente a elicitación, análise, especificação, validação e gerenciamento dos requisitos tende a solucionar, ou pelo menos minimizar significativamente, os problemas dessa categoria.

No entanto, as dificuldades essenciais são mais difíceis de serem superadas, uma vez que fazem parte da natureza do sistema, que é abstrata, maleável e complexa. A adoção de um processo sistemático para a Engenharia de Requisitos, principalmente no que se refere à especificação, validação e gerenciamento dos requisitos, também poderá ajudar na superação das dificuldades essenciais. Porém, a problemática que naturalmente existe no processo de comunicação e compreensão humana, que está no cerne da elicitación de requisitos, necessitará de uma abordagem que leve em consideração fatores como o contexto em que as pessoas exercem suas atividades e reconhecem os objetos que lhes são pertinentes, o histórico de evolução dessas atividades e seus instrumentos de mediação, e outros aspectos de relevância social e psicológica que afetam os usuários do sistema a ser desenvolvido.

Dessa forma, entendemos que as dificuldades essenciais enfrentadas na elicitación de requisitos não poderão ser resolvidas por uma abordagem puramente tecnológica, uma vez que os aspectos sociais assumem grande importância nessa atividade [31]. A maioria dos sistemas é desenvolvida sem nenhum auxílio das Ciências Sociais (como Psicologia, Sociologia, Antropologia etc.), não abordando de forma sistemática as necessidades do usuário, tanto em nível individual como organizacional. Normalmente as pessoas (usuários do sistema) não sabem o que elas

realmente desejam em termos da funcionalidade a ser embutida no sistema. Isto não significa que as pessoas não têm uma idéia geral do que elas gostariam que o sistema fizesse, mas que no início do processo de desenvolvimento elas não têm uma idéia precisa e detalhada de quais funcionalidades o sistema terá.

4.2.2 ETAPA 2: ANÁLISE, ESPECIFICAÇÃO E VALIDAÇÃO DE REQUISITOS.

Esta etapa tem como finalidade analisar, especificar e validar os requisitos do sistema de automação. Nesta etapa devem ser consideradas as informações geradas na etapa 1.

A primeira parte desta etapa visa analisar os requisitos necessários para desenvolver as ações com vista ao desenvolvimento e implementação do sistema.

A segunda parte desta etapa é a especificação de requisitos, onde todos os dispositivos e equipamentos necessários para a implementação do sistema devem ser especificados. Devem ser especificadas as capacidades e características gerais exigidas para cada um dos dispositivos e equipamentos de modo que os mesmos possam atender os requisitos levantados para o sistema.

A terceira parte desta etapa é a validação dos requisitos, onde o objetivo é verificar e validar os requisitos especificados.

4.2.2.1 ANÁLISE DE REQUISITOS

A atividade de análise de requisitos está muito vinculada à elicitação, uma vez que, na medida em que os requisitos vão se desvendando, algum grau de análise sobre os mesmos é realizada. O objetivo da análise de requisitos é encontrar

possíveis problemas na declaração informal dos requisitos obtida na etapa 1, elicitação de requisitos.

Considera-se que existe um processo de forte interação entre a elicitação e a análise de requisitos, conforme ilustra a figura 4.5.

Figura 4.5 - Espiral representando o processo de interação entre elicitação e análise de requisitos

Fonte: KOTONYA, SOMMERVILLE,1998

A Figura 4.5 procura demonstrar a interação entre as atividades de elicitação e análise de requisitos, adicionando uma tarefa importante dentro da atividade de análise, a negociação.

Enquanto os requisitos são descobertos, alguma análise acaba ocorrendo. Durante a análise é comum encontrar problemas nos requisitos descobertos, tipicamente conflitos e ambigüidades, que devem ser negociados entre desenvolvedores e usuários, de forma a se obter um "acordo" sobre os requisitos obtidos.

Existem três elementos fundamentais sobre os requisitos, que devem ser verificados na análise, auxiliando na identificação de problemas:

- a) **Verificação de Necessidade:** a necessidade de um requisito obtido deve ser analisada. Alguns requisitos podem ser propostos, mas não contribuem com as metas de negócio da organização ou para o problema específico a ser endereçado pelo sistema.
- b) **Verificação de consistência e completitude:** consistência significa que nenhum requisito deve ser contraditório, e completitude significa que nenhum serviço ou restrição que sejam necessários ao sistema tenha sido omitido.
- c) **Verificação de Possibilidade:** os requisitos devem ser verificados no sentido de assegurar que eles são factíveis de serem implementados no sistema a ser desenvolvido, tanto em termos de orçamento como de tempo.

4.2.2.2 ESPECIFICAÇÃO DE REQUISITOS

A etapa de especificação é onde os resultados da elicitação e análise de requisitos serão transformados em documentos que organizam os requisitos do sistema.

Através da especificação dos requisitos é possível se estabelecer um veículo efetivo de comunicação com os usuários do sistema, de forma que o entendimento dos problemas obtidos pelos desenvolvedores possam ser discutidos e elaborados em conjunto com os usuários.

A especificação de requisitos resulta em documentos que organizam os requisitos obtidos. Estes documentos assumem um papel fundamental no desenvolvimento de sistemas. Dentre os benefícios obtidos pelos documentos gerados pode-se citar:

1. O documento de especificação é o veículo básico de comunicação entre desenvolvedores e usuários sobre o que deve ser construído;
2. O documento de especificação registra os resultados da análise do problema (obtido através da elicitação e análise dos requisitos);

3. O documento de especificação define quais propriedades o sistema deve ter e quais são as restrições impostas em seu projeto e implementação;
4. O documento de especificação é a base para estimativas de custo e cronograma;
5. O documento de especificação é a base para o desenvolvimento do plano de teste do sistema;
6. O documento de especificação oferece uma definição padrão de comportamento esperado pelos profissionais envolvidos na manutenção do sistema;
7. O documento de especificação é utilizado para registrar mudanças na engenharia do sistema.

4.2.2.3 VALIDAÇÃO DE REQUISITOS

A etapa de validação procura certificar que os requisitos acordados e especificados representam uma descrição aceitável do sistema a ser construído [31].

A validação dos requisitos envolve tanto usuários como desenvolvedores, que analisam os requisitos buscando identificar possíveis problemas, omissões e ambigüidades. Os principais problemas descobertos durante a validação dos requisitos são:

- a) Não atendimento a padrões de qualidade;
- b) Requisitos descritos de forma ineficiente, levando à ambigüidade;
- c) Erros na modelagem do problema ou sistema;
- d) Requisitos conflitantes não identificados durante a etapa de análise.

Estes problemas devem ser sanados antes que o documento de especificação seja aprovado e utilizado para etapa 3, a modelagem do sistema por meio das Redes de Petri. Alguns problemas são resolvidos com a correção do documento de especificação, no entanto, outros problemas podem levar a uma nova execução da espiral da Engenharia de Requisitos, levando a uma nova elicitação, análise e especificação.

O principal problema enfrentado na validação dos requisitos é que não existe um documento que possa ser usado como base para a validação, uma vez que o próprio documento de especificação, que será validado, é o primeiro parâmetro formal de referência sobre o sistema a ser construído.

A Figura 4.6 representa as entradas e saídas envolvidas no processo de validação dos requisitos.

Figura 4.6 - O processo de Validação dos Requisitos

O documento de especificação deve estar em conformidade com os padrões da organização, representando os requisitos negociados e acordados entre usuários e desenvolvedores. O conhecimento organizacional, embora não seja uma "entrada tangível" no processo de validação, é muito importante, pois está muito ligado à cultura e às estruturas organizacionais onde o sistema será construído.

Como saída, o processo de validação deve gerar uma lista de problemas e ações acordadas. Esta deve relatar todos os problemas identificados no documento de especificação, classificando-os de alguma forma, como por exemplo: ambigüidade, inconsistência, etc. As ações acordadas são ações a serem executadas a partir dos problemas relacionados na lista.

Após a execução das tarefas de análise, especificação e validação de requisitos, é desenvolvida a arquitetura de hardware e software do sistema de automação. Cada dispositivo é identificado de acordo com a função desempenhada dentro do processo.

As informações que devem ser geradas nesta etapa são:

- Diagrama de hardware e software do sistema de automação;
- Definição dos dispositivos de detecção;
- Definição dos dispositivos de atuação;
- Definição dos dispositivos de monitoração;

A Figura 4.7 mostra de maneira esquemática as entradas e saídas da etapa 2.

Figura 4.7 - Etapa 2: Análise, Especificação e Validação de requisitos

4.2.3 ETAPA 3: MODELAGEM DO SISTEMA DE AUTOMAÇÃO POR MEIO DE REDES DE PETRI

Uma vez efetuada a especificação do sistema, etapas 1 e 2, é então elaborada a sua modelagem por meio das Redes de Petri. O modelo gerado pode ser então analisado com a finalidade de atender às especificações geradas na definição do sistema.

Qualquer problema encontrado pela análise do modelo indica uma falha na fase de projeto do sistema. O projeto deve, assim, ser modificado para corrigir as falhas. Esta modificação do projeto deve ser novamente modelada e analisada. Desta forma, o ciclo é repetido até não se encontrar falhas.

A Figura 4.8 mostra de maneira esquemática as entradas e saídas da etapa 3.

Figura 4.8 - Etapa 3: Modelagem do sistema de Automação por meio das Redes de Petri

O modelo do sistema é importante para tratar os problemas de interpretação das informações obtidas dos usuários (as quais, muitas vezes, podem ser subjetivas). O mesmo pode auxiliar na organização das idéias e no conhecimento do sistema e permitir a descrição e compreensão deste ao modelar as inter-relações e funções de seus diferentes componentes para, depois, derivar uma especificação funcional adequada.

A especificação funcional do sistema obtida nesta etapa define os requisitos do sistema. Uma análise destes requisitos pode até mesmo alterar os parâmetros inicialmente adotados para o sistema.

4.2.4 ETAPA 4: ANÁLISE, VALIDAÇÃO E VERIFICAÇÃO DO SISTEMA POR MEIO DAS RdPs

Nesta etapa, analisa-se e valida-se o modelo do sistema com a finalidade de confirmar se os objetivos iniciais das etapas 1 e 2 foram alcançados.

O modelo é analisado de acordo com as técnicas de análise das Redes de Petri.

De acordo com a estrutura da metodologia proposta neste trabalho, no caso de se encontrar algum problema, deve-se voltar à etapa devida para fazer as correções necessárias de forma a se ter um contínuo aprimoramento do modelo. A Figura 4.9 mostra de maneira esquemática as entradas e saídas da etapa 4.

Figura 4.9 - Etapa 4: Análise, Validação e Verificação do Sistema por meio das RdP

4.2.5 ETAPA 5: CONVERSÃO DO MODELO DO SISTEMA EM REDES DE PETRI PARA LINGUAGEM LADDER

Uma vez que a RdP que modela o sistema a ser automatizado é aprovada, passa-se para a etapa de conversão para linguagem de programação Ladder.

Na etapa 5 é empregado um método de conversão de Redes de Petri que modela o sistema em estudo para a Linguagem de programação.

O objetivo principal deste método de conversão é reduzir o esforço de projeto devido à complexidade de problemas de controle [21].

As primeiras quatro linhas da tabela 4-2 mostram em fila como os elementos básicos de redes de Petri são usados para modelar condições, estados, atividades, informações e fluxo de material e recursos e que a linguagem Ladder não possui as representações explícitas correspondentes.

Tabela 4-2 - Modelagem de redes de Petri e linguagem Ladder.

LÓGICA DE CONSTRUÇÃO	REDE DE PETRI EQUIVALENTE	PROGRAMA EM LADDER EQUIVALENTE
Condição ou status do elemento do sistema	Lugar	Representação não explícita
Atividade ou Evento	Transição	Representação não explícita
Fluxo de informação ou material	Arco direcionado	Representação não explícita
Objetos ativos: máquinas, robôs, peças, etc	Lugar com marcação	Representação não explícita
<u>Lógica AND (E)</u> se A=1 e B=1 e C=1 então D=1
<u>Lógica OR (OU)</u> se A=1 ou B=1 ou C=1 então D=1
<u>CONCORRÊNCIA</u> se A=1 e B=1 então C=1 e D=1 e E=1
<u>TEMPORIZAÇÃO</u> se A=1 então após atraso de tempo τ B=1
<u>SINCRONIZAÇÃO</u> se A=1 então após atraso de tempo τ_1 D=1 se B=1 e C=1 então após atraso de tempo τ_2 E=1 se D=1 e E=1 então após atraso de tempo τ_3 F=1

Fonte: VENKATESH, ZHOU, CAUDILL,1995.

Para ilustrar a aplicação da tabela 4-2, tem-se a função lógica: $Y = ((X0 + X1 + X2).(X4.X5)) + X3$ que gera a linguagem Ladder da figura 4.10, e conseqüentemente a rede de Petri da figura 4.11.

Figura 4.10 - Linguagem Ladder da função $Y = ((X0 + X1 + X2).(X4.X5)) + X3$.

Figura 4.11 - Rede de Petri da função $Y = ((X0 + X1 + X2).(X4.X5)) + X3$.

A Figura 4.12 mostra de maneira esquemática as entradas e saídas da etapa 5.

Figura 4.12 - Etapa 5: Conversão RdP para Linguagem Ladder

4.2.6 ETAPA 6: IMPLEMENTAÇÃO E TESTE DO SISTEMA

Na etapa 6 os modelos gerados, validados e verificados são implementados como sistema físico e/ou como software de controle.

A tabela 4-3 ilustra as etapas, as ações de entradas e saídas da metodologia MARPASI.

Tabela 4-3 - Resumo da Metodologia MARPASI.

ETAPA	ENTRADA	SAÍDA	ATIVIDADE	MÉTODO
ETAPA 1 DEFINIÇÃO DO PROBLEMA	Necessidade do usuário	Descrição estruturada do problema	Conhecimento do problema	Entrevistas, solicitações
ETAPA 2 ANÁLISE DO SISTEMA E DEFINIÇÃO DOS REQUISITOS	Descrição estruturada do problema (Sistemas existentes Possíveis)	Estrutura do sistema (Modelo comunicativo)	Conhecimento do sistema	Engenharia de requisitos
ETAPA 3 MODELAGEM DO SISTEMA POR MEIO DAS RdP	Comportamento do sistema (Atividades e elementos)	Modelo formal	Geração de Modelo	Redes de Petri
ETAPA 4 ANÁLISE, VALIDAÇÃO E VERIFICAÇÃO DO SISTEMA	Modelo formal	Modelos para implementação do sistema	Validação, verificação e aprimoramento do modelo	Técnicas de Análise das RdP
ETAPA 5 CONVERSÃO RdP PARA LINGUAGEM LADDER	Modelos para implementação do sistema	Linguagem Ladder do Sistema	Validação, verificação da Linguagem Ladder do Sistema	Petri-Ladder
ETAPA 6 IMPLEMENTAÇÃO E TESTE DO SISTEMA	Projeto Completo do sistema	Aplicação em Sistema Real	Validação e Verificação	Implementação/Teste

5 IMPLEMENTAÇÃO E ANÁLISE DE APLICAÇÃO DA MARPASI - “ESTUDO DE CASO”

5.1 INTRODUÇÃO

Este capítulo apresenta a aplicação e a análise de viabilidade do emprego da MARPASI como ferramenta de engenharia no auxílio à programação de CLP em projetos de automação industrial.

Como estudo de caso foi escolhida parte de uma linha automatizada de uma importante companhia de cosméticos. A automação da linha produtiva da planta escolhida foi desenvolvida dentro do convênio Rockwell Automation & EPUSP no ano de 2005. O sistema de automação implementado nesta linha é relativamente complexo e, portanto o mesmo pode servir satisfatoriamente como estudo de caso para aplicação e validação da MARPASI.

Os principais objetivos desta etapa do trabalho são:

- Aplicação da MARPASI em um caso real de projeto de automação industrial.
- Análise comparativa entre: o desempenho, o tamanho e a modularidade do programa ladder obtido por meio de aplicação da MARPASI e o programa ladder obtido e implementado por meio dos procedimentos tradicionais da engenharia de automação.
- Análise qualitativa do nível de investimento de esforço e de tempo para desenvolver um programa Ladder empregando técnicas tradicionais da engenharia de automação e empregando a MARPASI
- Validação neste nível de pesquisa da MARPASI.

5.2 ESTUDO DE CASO – AUTOMAÇÃO DOS TANQUES DE RECEBIMENTO DE SEBO DA INDÚSTRIA ESCOLHIDA

O processo escolhido para validar a MARPASI foi a planta parcial de produção de cosméticos.

A figura 5.1 ilustra o sistema automatizado que serviu como referência de aplicação da MARPASI.

Figura 5.1 - Esquema ilustrativo dos Tanques de Recebimento de Sebo a ser automatizado

A tabela 5-1 apresenta a lista dos equipamentos incluindo tanques, bombas, filtros com suas respectivas codificações adotadas no projeto.

Tabela 5-1 - Identificação das referências citadas no esquema ilustrativo da figura 5.1

ÍTEM	DESCRIÇÃO
TQ-28	TANQUE DE RECEBIMENTO DE SEBO
TQ-29	TANQUE DE RECEBIMENTO DE SEBO
TQ-30	TANQUE DE RECEBIMENTO DE SEBO
TQ-31	TANQUE DE RECEBIMENTO DE SEBO
V-28	VÁLVULA PARA CONTROLE DO DESCARREGAMENTO DO TANQUE 28 (TQ-28)
V-29	VÁLVULA PARA CONTROLE DO DESCARREGAMENTO DO TANQUE 29 (TQ-29)
V-30	VÁLVULA PARA CONTROLE DO DESCARREGAMENTO DO TANQUE 30 (TQ-30)
V-31	VÁLVULA PARA CONTROLE DO DESCARREGAMENTO DO TANQUE 31 (TQ-31)
V-32	VÁLVULA DE CONTROLE DE VAPOR NA TUBULAÇÃO DE DESCARGA DOS TANQUES
V-33	VÁLVULA DE ENTRADA(SUCÇÃO) DA BOMBA P006
V-34	VÁLVULA DE SAÍDA(RECALQUE) DA BOMBA P006
V-35	VÁLVULA DE ENTRADA(SUCÇÃO) DA BOMBA P001
V-36	VÁLVULA DE SAÍDA(RECALQUE) DA BOMBA P001
V-37	VÁLVULA DE ENTRADA DO FILTRO F001
V-38	VÁLVULA DE ENTRADA DO FILTRO F002
V-39	VÁLVULA DE SAÍDA DO FILTRO F001
V-40	VÁLVULA DE SAÍDA DO FILTRO F002
V-41	VÁLVULA PARA SAÍDA DE VAPOR DA TUBULAÇÃO
P001	BOMBA P001
P006	BOMBA P006
F001	FILTRO F001
F002	FILTRO F002

Esta parte da planta automatizada é composta de quatro tanques de recebimento de sebo denominados TQ-28, 29,30 e 31. A engenharia do processo é quem determina o algoritmo de controle dos reservatórios ou tanques. O enchimento e o esvaziamento são efetuados por meio da abertura e fechamento das válvulas de admissão e de saída. Observe-se que neste trabalho de pesquisa a aplicação da MARPASI o algoritmo de referência de automação é o de esvaziamento dos reservatórios. Assim sendo, a figura 5.1 ilustra exclusivamente as válvulas de descargas dos reservatórios. O sistema possui uma válvula denominada V-32 que controla a entrada de vapor em alta pressão na tubulação de descarga dos tanques. Existem dois filtros denominados F001 e F002 e duas bombas denominadas P001 e P006. Na tubulação de entrada e saída de cada filtro e de cada bomba existem válvulas denominadas V-33,V-34,V-35,V-36,V-37,V-38,V-39 e V-40. O sistema possui também uma válvula denominada V-41 para saída de vapor. A seleção do tanque, da bomba e do filtro que será utilizado é uma opção do operador do sistema ou do sistema supervisor.

O processo automatizado efetua 5 macro seqüências de operação:

a) Seqüência 1: Início/Repouso

O processo inicia com o desligamento das bombas P001 e P006, e o fechamento de todas as válvulas V-28 a V-41.

Este é a parte do algoritmo que implementa a rotina de segurança do início do processo.

b) Seqüência 2: Limpeza das Tubulações

Antes de descarregar os tanques é necessário fazer a limpeza da tubulação de saída dos tanques.

A limpeza das tubulações é efetuada por meio do enchimento da tubulação com vapor em alta pressão. Para iniciar a limpeza da tubulação deve-se abrir a válvula correspondente ao tanque do qual o produto será descarregado, aguardar 44 segundos para abrir a válvula de vapor V-32 e fechar as válvulas V-28, V-29, V-30 e V-31.

c) Seqüência 3: Descarregamento dos Tanques

Após a limpeza da tubulação, o processo de descarregamento dos tanques deve ser iniciado com a abertura da válvula V-41 para saída de vapor da tubulação, seleção do filtro (F001 ou F002) , abertura das válvulas V-37 e V-39 caso for selecionado o Filtro F001, ou V-38 e V-40 caso for selecionado o Filtro F002 e fechamento da V-41 após 50 segundos da mesma ter sido aberta.

Após o fechamento da válvula V-41 deve ser selecionada a bomba (P001 ou P006) e aberta suas respectivas válvulas: V-35 e V-36 para a bomba P001 e válvulas V-33 e V-34 para a bomba P006. Decorridos 5 segundos de abertura das válvulas V-

33 e V-34 ou V-35 e V-36, a válvula de vapor V-32 deve ser fechada e aberta a válvula correspondente ao tanque a ser descarregado.

Depois de confirmado que as válvulas da bomba selecionada estão abertas, a mesma deve ser ligada.

Nesta etapa do processo, caso ocorra um pedido de parada de descarga por parte do operador, o processo deve seguir imediatamente para a sequência 4.

d) Sequência 4: Limpeza da Tubulação para Parada

É aberta a válvula correspondente ao tanque selecionado (V-29, V-30 ou V-31) e desligada as duas bombas P001 e P006.

Após a confirmação de que as válvulas dos tanques estão fechadas, deve-se abrir a válvula de vapor V-32 e fechar as válvulas de entrada e saída das bombas, isto é, V-33, V-34, V-35 e V-36.

e) Sequência 5: Limpeza para Parada

Abrir a válvula V-41 e fechar a V-32. Fechar a válvula V-41 após 50 segundos de sua abertura e retornar à sequência 1.

5.2.1 APLICAÇÃO DA “MARPAISI”

5.2.1.1 ETAPA 1: DEFINIÇÃO DA OPERACIONALIDADE DO SISTEMA DE AUTOMAÇÃO

O objetivo desta etapa é definir a operacionalidade do sistema de automação. A operação automática do sistema dos tanques de recebimento, também neste caso,

consiste de cinco seqüências. A execução do sistema deve seguir em ordem a seqüência de execução. A seguir tem-se a descrição do algoritmo a ser implementado com a identificação de cada uma seqüências.

O hardware para a automação deste sistema consiste de um CLP, cartões de entrada e saída, atuadores nas válvulas e também de um PC onde roda um sistema supervisorio.

A figura 5.2 ilustra a arquitetura de hardware do sistema.

Figura 5.2 - Arquitetura de hardware do sistema de automação dos tanques

a) Seqüência 1: Início/Repouso.

Para início do processo de descarga dos tanques de recebimento deverá ser pressionado o botão iconográfico da tela do sistema supervisorio **“CONFIRMAR DESCARGA”**, que passará o comando para que o CLP execute as seguintes ações:

- Desligar as Bombas P001 e P006;
- Fechar todas as válvulas, isto é, V-28, V-29, V-30, V-31, V-32, V-33, V-34, V-35, V-36, V-37, V-38, V-39, V-40 e V-41.

b) Sequência 2 : Limpeza da Tubulação

O processo de limpeza da linha será iniciado automaticamente após a conclusão da sequência 1 (início/repouso) e deverá ter as seguintes ações:

- Selecionar o tanque do qual o sebo será descarregado (TQ-28, TQ-29, TQ-30 ou TQ-31);
- Abrir a válvula do tanque selecionado (V-28, V-29, V-30 ou V-31), aguardar $t = 44$ segundos para abrir a válvula V-32 (vapor) e fechar as válvulas (V-28, V-29, V-30 e V-31).

c) Sequência 3: Descarga dos Tanques

O processo de descarga dos tanques será iniciado automaticamente após a conclusão da sequência 2 (limpeza da tubulação) e deverá ter as seguintes ações:

- Abrir a válvula V-41;
- Selecionar o Filtro F001 ou o F002;
- Abrir as válvulas V-37 e V-39 se selecionado o filtro F001, ou V-38 e V-40 se o filtro selecionado for F002. Abrir primeiro as válvulas de saídas dos filtros, isto é, V-37 ou V-38, conforme o filtro selecionado.
- Decorridos $t = 50$ segundos, fechar V-41, selecionar a Bomba P001 ou P006, abrir as válvulas V-35 e V-36 se for selecionada a bomba P001, ou V-33 e V-34 se for selecionada a Bomba P006.

- Após 5 segundos de abertura das válvulas V-35 e V-36 ou V-33 e V-34, fechar a válvula V-32, abrir primeiro as válvulas de entrada das bombas, isto é, V-33 ou V-35, conforme a bomba selecionada, e por fim, abrir a válvula do tanque selecionado (V-28, V-29, V-30 ou V-31) e ligar a bomba selecionada (P001 ou P006).

O operador pode solicitar parada através do botão “PARAR DESCARGAR” e o sistema deverá imediatamente seguir à seqüência 4.

d) Seqüência 4 : Início da Parada

- Fechar a válvula do tanque selecionado no passo anterior (V-28, V-29, V-30 ou V-31).
- Desligar a bomba selecionada (P001 ou P006).
- Abrir a válvula V-32 e após $t=15$ segundos fechar as válvulas V-33, V-34, V-35 e V-36.

e) Seqüência 5: Limpeza para Parada

- Abrir as válvulas V-41 e fechar a válvula V-32.
- Fechar a válvula V-41 após 50 segundos da abertura da mesma.

Observação: a qualquer instante o processo do sistema de automação pode ser abortado por meio do operador, que deverá pressionar o botão iconográfico “ABORTAR PARADA”, reiniciando o processo na seqüência 1.

5.2.1.2 ETAPA 2: ANÁLISE, ESPECIFICAÇÃO E VALIDAÇÃO DOS REQUISITOS

5.2.1.2.1 ANÁLISE DOS REQUISITOS

Com o intuito de analisar os requisitos apresentados na Etapa 1 é elaborado um fluxograma de cada um dos passos do processo.

A figura 5.3 ilustra o ciclo do processo.

Figura 5.3 - Ciclo do Processo de Automatização dos Tanques de Recebimento de Sebo

A figura 5.4 ilustra a sequência 1- Início/Repouso.

Figura 5.4 - Fluxograma da Sequência 1 - Início/Repouso

A figura 5.5 ilustra a sequência 2- Limpeza da Tubulação.

Figura 5.5 - Fluxograma da sequência 2 - Limpeza da Tubulação

A figura 5.6 ilustra a sequência 3- Descarga dos Tanques.

Figura 5.6 - Fluxograma da sequência 3 - Descarga dos Tanques

A figura 5.7 ilustra a sequência 4- Início da Parada.

Figura 5.7 - Fluxograma da sequência 4 - Início da Parada

A figura 5.8 ilustra com detalhes a sequência 5- Limpeza para Parada.

Figura 5.8 - Fluxograma da sequência 5 - Limpeza para Parada

A análise dos requisitos permite basicamente concluir que:

- a) Não existe nenhuma mudança significativa no algoritmo do processo manual com o algoritmo de processo automático.
- b) Não existe a supervisão ou atuação proporcional analógica em nenhuma variável de referência controlada. Assim sendo, o processo de controle é exclusivamente um processo de seqüenciamento e intertravamento.
- c) O algoritmo do processo é implementável pela arquitetura de hardware proposta.

5.2.1.3 ETAPA 3: MODELAGEM DO SISTEMA DE AUTOMAÇÃO POR MEIO DE REDES DE PETRI

Esta etapa consiste na modelagem do sistema por meio das Redes de Petri, com base nos requisitos solicitados na etapa 1 e analisados na etapa 2.

O sistema será modelado conforme as seqüências descritas na etapa 1. A tabela 5-1 descreve cada uma das posições (condições) da Rede de Petri para a modelagem do processo. Nas figuras 5.9 a 5.13 as posições de cor amarela referem-se aos comandos manuais do operador.

Neste processo convencionou-se que, quando ocorre qualquer um dos eventos descritos na tabela 5-1, há uma marca presente na posição associada.

As figuras 5.9 a 5.13 ilustram as cinco seqüências da Rede de Petri do sistema. Na figura 5.9 tem-se a seqüência 1 – Início/Repouso do sistema, que para ser iniciado deve ser confirmado por meio do comando do operador “CONFIRMAR DESCARGA”.

Na figura 5.10 tem-se a Rede de Petri da seqüência 2 – Limpeza da Tubulação, que somente será iniciada após a conclusão do passo 1, isto é, terá que ter marcação na posição P1-2.

Na figura 5.11 tem-se a Rede de Petri da seqüência 3 – Descarga dos Tanques, que mais uma vez só será iniciada quando concluída a seqüência 2. Na conclusão desta etapa a posição P2-3 estará marcada, habilitando o passo seguinte.

A figura 5-12 ilustra a Rede de Petri da seqüência 4 – Início da Parada.

A figura 5-13 ilustra a Rede de Petri da seqüência 5 – Limpeza para Parada, que após o fechamento da válvula V-41, retornará ao estado inicial, aguardando um novo ciclo do processo, através do comando externo CONFIRMAR DESCARGA.

Tabela 5-2 - Descrição das Posições (condições) das Redes de Petri

POSIÇÃO	DESCRIÇÃO
CONFIRMAR DESCARGA	CONFIRMAÇÃO DE DESCARREGAMENTO
P001_DESL / P001_LIG	BOMBA P001 DESLIGADA / BOMBA P001 LIGADA
P006_DESL / P006_LIG	BOMBA P006 DESLIGADA / BOMBA P006 LIGADA
V-28_FE / V-28_AB	VÁLVULA V-28 FECHADA / VÁLVULA V-28 ABERTA
V-29_FE / V-29_AB	VÁLVULA V-29 FECHADA / VÁLVULA V-29 ABERTA
V-30_FE / V-30_AB	VÁLVULA V-30 FECHADA / VÁLVULA V-30 ABERTA
V-31_FE / V-31_AB	VÁLVULA V-31 FECHADA / VÁLVULA V-31 ABERTA
V-32_FE / V-32_AB	VÁLVULA V-32 FECHADA / VÁLVULA V-32 ABERTA
V-33_FE / V-33_AB	VÁLVULA V-33 FECHADA / VÁLVULA V-33 ABERTA
V-34_FE / V-34_AB	VÁLVULA V-34 FECHADA / VÁLVULA V-34 ABERTA
V-35_FE / V-35_AB	VÁLVULA V-35 FECHADA / VÁLVULA V-35 ABERTA
V-36_FE / V-36_AB	VÁLVULA V-36 FECHADA / VÁLVULA V-36 ABERTA
V-37_FE / V-37_AB	VÁLVULA V-37 FECHADA / VÁLVULA V-37 ABERTA
V-38_FE / V-38_AB	VÁLVULA V-38 FECHADA / VÁLVULA V-38 ABERTA
V-39_FE / V-39_AB	VÁLVULA V-39 FECHADA / VÁLVULA V-39 ABERTA
V-40_FE / V-40_AB	VÁLVULA V-40 FECHADA / VÁLVULA V-40 ABERTA
V-41_FE / V-41_AB	VÁLVULA V-41 FECHADA / VÁLVULA V-41 ABERTA
TQ-28/29/30/31_SEL	TANQUE 28/29/30/31 SELECIONADO
F001/F002_SEL	FILTRO F001/F002 SELECIONADO
P1-2	PASSO 1 CONCLUÍDO
P2-3 / P2.3.1	PASSO 2 CONCLUÍDO
P3-4	PASSO 3 CONCLUÍDO
P4-5	PASSO 4 CONCLUÍDO
P_AUX_1,2,3,4,5,6,7,8 e 9	POSIÇÕES AUXILIARES
PARAR DESCARGA	SOLICITAÇÃO PARA PARAR DESCARGA

Figura 5.9 - Rede de Petri da sequência 1 - Início/Repouso

Figura 5.10 - Rede de Petri da sequência 2 - Limpeza da Tubulação

Figura 5.11 - Rede de Petri da sequência 3 - Descarga dos Tanques

Figura 5.12 - Rede de Petri da sequência 4 - Início da Parada

Figura 5.13 - Rede de Petri da sequência 5 - Limpeza para Parada

5.2.1.4 ETAPA 4: ANÁLISE, VALIDAÇÃO E VERIFICAÇÃO DO SISTEMA POR MEIO DAS RDPs

Esta etapa consiste em validar a Rede de Petri do sistema modelado com base nas propriedades das Redes de Petri, conforme apresentado na tabela 2-1.

De acordo com o sistema a ser automatizado, julgou-se necessário a verificação das seguintes propriedades:

- **Impasse (*deadlock*):** Não pode haver nenhum *deadlock* na rede, pois significará que o sistema travará e conseqüentemente as atividades seguintes não serão efetuadas;
- **Limitação e segurança:** A rede é segura ou 1-limitada, isto é, cada posição da rede não terá mais que uma marca, assegurando que não ocorrerão habilitações ou inabilitações indesejadas;
- **Vivacidade:** Para assegurar que todas as transições desejadas serão habilitadas;
- **Alcançabilidade:** Verificar se todos os estados (posições) desejadas serão executadas.

Cada seqüência do sistema a ser automatizado não continua sem que o passo anterior seja concluído. A análise da Rede de Petri será feita para cada uma dessas seqüências.

A tabela 5-3 mostra as 16 marcações iniciais possíveis conforme o tanque, filtro ou bomba selecionada. Essas marcações são importantes, pois se deve fazer uma análise para cada uma delas, conforme o passo que se está analisando.

Tabela 5-3 - Marcações iniciais possíveis da RdP das figuras 5.9 a 5.13

MARCAÇÃO INICIAL	CONFIRMA DESCARGA	TQ-28_SEL	TQ-29_SEL	TQ-30_SEL	TQ-31_SEL	F001_SEL	F002_SEL	P001_SEL	P006_SEL
M1	1	1				1		1	
M2	1	1				1			1
M3	1	1					1	1	
M4	1	1					1		1
M5	1		1			1		1	
M6	1		1			1			1
M7	1		1				1	1	
M8	1		1				1		1
M9	1			1		1			
M10	1			1		1			1
M11	1			1			1	1	
M12	1			1			1		1
M13	1				1	1			
M14	1				1	1			1
M15	1				1		1	1	
M16	1				1		1		1

Para analisar a Rede de Petri que modela o sistema de automação em questão, pode-se usar qualquer das técnicas de análise apresentada na seção 2.7 ou uma combinação das técnicas de acordo com a exigência do sistema de automação. Para fins de demonstração serão usadas as técnicas matemáticas de análise.

a) Análise do Passo 1: Início/Repouso

A figura 5.14 mostra a árvore de alcançabilidade da seqüência 1 – Início/Parada, dado o estado inicial M0.

Figura 5.14 - Árvore de Alcançabilidade da seqüência 1 - Início/Parada

Tabela 5-4 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.14

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21
M0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
M1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0
M2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
M3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Com base na árvore de alcançabilidade é possível afirmar que a RdP tem as seguintes características:

- **Impasse (*deadlock*):** Não há impasse, pois todas as transições desejadas são habilitadas;
- **Segura:** A RdP é segura, pois na visualização de todas as coordenadas da árvore de alcançabilidade o maior índice é igual a 1;
- **Limitação:** A rede é 1-limitada, pois é segura;
- **Vivacidade:** A RdP é viva, pois todas as transições desejadas são habilitadas ao menos uma vez;
- **Alcançabilidade:** Todas as marcações são alcançadas.

b) **Análise da seqüência 2: Limpeza da Tubulação**

Para análise desta seqüência, deve-se levar em consideração 4 possíveis marcações que tenham todos os quatro tanques selecionados, portanto, serão analisadas as marcações M1, M5, M9 e M13.

As figuras 5.15, 5.16, 5.17 e 5.18 mostram a árvore de alcançabilidade da seqüência 2 – Limpeza da Tubulação, considerando-se a marcação inicial M1, M5, M9 e M13, isto é, tanque 28 selecionado, tanque 29 selecionado, tanque 30 selecionado e tanque 31 selecionado, respectivamente, conforme tabela 5-3.

Figura 5.15 - Árvore de Alcançabilidade da seqüência 2 –Limpeza da Tubulação com marcação

Tabela 5-5 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.15

[illegible]

Figura 5.16 - Árvore de Alcançabilidade da seqüência 2 –Limpeza da Tubulação com marcação

Tabela 5-6 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.16

[illegible]

Figura 5.18 - Árvore de Alcançabilidade da sequência 2 –Limpeza da Tubulação com marcação M13

Tabela 5-8 - Marcações Alcançáveis da Árvore de Alcançabilidade da Figura 5.18

	P1	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30	P31	P32	P33	P34	P35	P36	P37	P38
M13	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
M13.1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
M13.2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
M13.3	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0
M13.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
M13.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Analisando-se as árvores de alcançabilidade da sequência 2 com todas as opções possíveis, isto é, para as marcações iniciais M1, M5, M9 e M13, tem-se que:

- **Impasse (*deadlock*):** Não há impasse, pois todas as transições são habilitadas, conforme a marcação inicial;
- **Segura:** A RdP é segura, pois na visualização de todas as coordenadas da árvore de alcançabilidade o maior índice é igual a 1;
- **Limitação:** A rede é 1-limitada, pois é segura;

- **Vivacidade:** A RdP é viva, pois todas as transições desejadas são habilitadas ao menos uma vez, conforme a marcação inicial;
- **Alcançabilidade:** Todas as marcações desejadas são alcançadas, conforme a marcação inicial.

c) **Análise da seqüência 3: Descarga dos Tanques**

Para análise desta seqüência, deve-se levar em consideração 16 possíveis marcações que tenham todos os quatro tanques selecionados, as duas bombas e os dois filtros, portanto, deveriam ser analisadas as marcações M1 a M16. Porém, para efeito de demonstração serão analisadas somente as marcações M1 a M4, considerando-se a não solicitação de parada e descargas dos tanques.

As figuras 5.19 a 5.22 apresentam as árvores de alcançabilidade da seqüência 3 – Descarga dos tanques para as marcações iniciais M1, M2, M3 e M4, respectivamente.

As tabelas 5-9 a 5-12 ilustram as marcações alcançáveis originadas das árvores de alcançabilidade das figuras 5.19 a 5.22, respectivamente.

Figura 5.19 - Árvore de Alcançabilidade da sequência 3 - Descarga dos Tanques - Marcação M1

Figura 5.20 - Árvore de Alcançabilidade da sequência 3 – Descarta dos Tanques – Marcação M2

Figura 5.21 - Árvore de Alcançabilidade da sequência 3– Descarga dos Tanques – Marcação M3

Figura 5.22 - Árvore de Alcançabilidade da sequência 3– Descarga dos Tanques – Marcação M4

Analisando-se as árvores de alcançabilidade da sequência 3 para marcações iniciais M1, M2, M3 e M4, tem-se que:

- **Impasse (*deadlock*):** Não há impasse, pois todas as transições desejadas são habilitadas, conforme a marcação inicial;
- **Segura:** A RdP é segura, pois na visualização de todas as coordenadas da árvore de alcançabilidade o maior índice é igual a 1;
- **Limitação:** A rede é 1-limitada, pois é segura;
- **Vivacidade:** A RdP é viva, pois todas as transição são habilitadas ao menos uma vez, conforme a marcação inicial;
- **Alcançabilidade:** Todas as marcações desejadas são alcançadas, conforme a marcação inicial.

5.2.1.4.1 CONSIDERAÇÕES SOBRE A ANÁLISE DO SISTEMA MODELADO PELA RDP.

A análise analítica da Rede de Petri em questão provou que o sistema pode operar com segurança.

Entretanto, foi necessário investir bastante esforço para analisá-la. Atualmente existem diversos softwares livres que permitem que as análises das Redes de Petri sejam efetuadas de forma muito mais rápida e eficiente.

Neste trabalho de pesquisa o software PnetLab foi empregado para também efetuar a análise da Rede de Petri.

Os resultados obtidos nos ensaios de simulação foram os mesmos obtidos analiticamente.

5.2.1.5 ETAPA 5: CONVERSÃO DO MODELO DO SISTEMA EM REDES DE PETRI PARA LINGUAGEM LADDER

Nesta etapa o objetivo é converter a Rede de Petri que modela o sistema em linguagem de programação Ladder para implementação em hardware.

As tabelas 5-13 e 5-14 apresentam a lista de entrada (I) e saída (O) do Controlador Lógico Programável. A figura 5.23 ilustra o diagrama Ladder obtido por meio da tabela 4-2.

No diagrama Ladder da figura 5.23 existem duas instruções de salto: LBL(label) e JMP(salto) . A instrução JMP efetua um salto até a instrução LBL quando a condições da linha da instrução JMP é verdadeira. A instrução "Label" retorna a varredura do programa a partir deste Label. A instrução LBL é sempre verdadeira.

Tabela 5-13 - Lista de Entradas (I) do CLP

ENDEREÇO FÍSICO	NOME	TIPO DE DADO	DESCRIÇÃO
10.0	CONFIRMAR_DESCARGA	BINÁRIO	ENTRADA DIGITAL 0 - SLOT 0
10.1	ABORTAR_DESCARGA	BINÁRIO	ENTRADA DIGITAL 1 - SLOT 0
10.2	PARAR_DESCARGA	BINÁRIO	ENTRADA DIGITAL 2 - SLOT 0
10.3	TQ-28 SELECIONADO	BINÁRIO	ENTRADA DIGITAL 3 - SLOT 0
10.4	TQ-29 SELECIONADO	BINÁRIO	ENTRADA DIGITAL 4 - SLOT 0
10.5	TQ-30 SELECIONADO	BINÁRIO	ENTRADA DIGITAL 5 - SLOT 0
10.6	TQ-31 SELECIONADO	BINÁRIO	ENTRADA DIGITAL 6 - SLOT 0
10.7	FILTRO F001_SELECIONADO	BINÁRIO	ENTRADA DIGITAL 7 - SLOT 0
11.0	FILTRO F002_SELECIONADO	BINÁRIO	ENTRADA DIGITAL 0 - SLOT 1
11.1	BOMBA P001_SELECIONADO	BINÁRIO	ENTRADA DIGITAL 1 - SLOT 1
11.2	BOBAM P002_SELECIONADO	BINÁRIO	ENTRADA DIGITAL 2 - SLOT 1
11.3		BINÁRIO	ENTRADA DIGITAL 3 - SLOT 1
11.4		BINÁRIO	ENTRADA DIGITAL 4 - SLOT 1
11.5		BINÁRIO	ENTRADA DIGITAL 5 - SLOT 1
11.6		BINÁRIO	ENTRADA DIGITAL 6 - SLOT 1
11.7		BINÁRIO	ENTRADA DIGITAL 7 - SLOT 1

Tabela 5-14 - Lista de Saídas (O) do CLP

ENDEREÇO FÍSICO	NOME	TIPO DE DADO	DESCRIÇÃO
O0.0	BOMBA P001	BINÁRIO	SAÍDA DIGITAL 0 - SLOT 0
O0.1	BOMBA P002	BINÁRIO	SAÍDA DIGITAL 1 - SLOT 0
O0.2	VÁLVULA V28	BINÁRIO	SAÍDA DIGITAL 2 - SLOT 0
O0.3	VÁLVULA V29	BINÁRIO	SAÍDA DIGITAL 3 - SLOT 0
O0.4	VÁLVULA V30	BINÁRIO	SAÍDA DIGITAL 4 - SLOT 0
O0.5	VÁLVULA V31	BINÁRIO	SAÍDA DIGITAL 5 - SLOT 0
O0.6	VÁLVULA V32	BINÁRIO	SAÍDA DIGITAL 6 - SLOT 0
O0.7	VÁLVULA V33	BINÁRIO	SAÍDA DIGITAL 7 - SLOT 0
O1.0	VÁLVULA V34	BINÁRIO	SAÍDA DIGITAL 0 - SLOT 1
O1.1	VÁLVULA V35	BINÁRIO	SAÍDA DIGITAL 1 - SLOT 1
O1.2	VÁLVULA V36	BINÁRIO	SAÍDA DIGITAL 2 - SLOT 1
O1.3	VÁLVULA V37	BINÁRIO	SAÍDA DIGITAL 3 - SLOT 1
O1.4	VÁLVULA V38	BINÁRIO	SAÍDA DIGITAL 4 - SLOT 1
O1.5	VÁLVULA V39	BINÁRIO	SAÍDA DIGITAL 5 - SLOT 1
O1.6	VÁLVULA V40	BINÁRIO	SAÍDA DIGITAL 6 - SLOT 1
O1.7	VÁLVULA V41	BINÁRIO	SAÍDA DIGITAL 7 - SLOT 1

Figura 5.23 - Diagrama Ladder do Sistema modelado em Redes de Petri

Figura 5.24 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação)

Figura 5.25 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação)

Figura 5.26 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação)

Figura 5.27 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação)

Figura 5.28 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação)

Figura 5.29 - Diagrama Ladder do Sistema modelado em Redes de Petri (continuação)

Figura 5.30 - Diagrama Ladder do Sistema modelado em Redes de Petri

5.3 COMPARAÇÃO DO PROCESSO DE PROGRAMAÇÃO TRADICIONAL DE AUTOMAÇÃO DO SISTEMA ESTUDADO COM O PROCESSO DE PROGRAMAÇÃO DO SISTEMA COM A MARPASI.

Os principais quesitos e resultados que podem ser levados em consideração na análise comparativa entre a programação da automação do sistema de tanques de recebimento de sebo com a programação do mesmo sistema empregando a MARPASI são:

- a) Tamanho do programa Ladder final;
- b) Organização e clareza do programa Ladder final;
- c) Facilidade para alteração e manutenção;
- d) Tempo e esforço para elaboração, testes e validação.
- e) Verificação da assertividade do programa Ladder.

a) Tamanho do Programa

Segundo [02] Para determinar genericamente o tamanho do programa em Ladder deve-se observar principalmente duas variáveis:

- N_o = número de operações no programa. O número de operações de um programa Ladder envolve as diversas correlações lógicas entre as instruções e também o número de resultados das mesmas. Neste trabalho de pesquisa serão considerada como sendo o número de instruções de saída.
- N_s = número de variáveis de estados, qualquer variável (bit interno, externo, contator, temporizador, etc).

A tabela 5-15 apresenta os dados relativos ao tamanho do programa obtidos por meio da MARPASI com os dados do tamanho do programa do sistema implementado atualmente por meio do emprego das técnicas tradicionais da engenharia de automação.

Tabela 5-15 - Comparação do Tamanho do Programa Ladder sem e com a MARPASI

	TAMANHO DO PROGRAMA		
	N_o	N_s	<i>Total</i>
LADDER COM MARPASI	63	54	117
LADDER SEM MARPASI	89	70	159

Observa-se que o Ladder obtido com a MARPASI é menor tanto no número de linhas como no número de variáveis de estados.

É importante ressaltar que o tamanho do diagrama Ladder obtido sem a MARPASI depende muito da experiência do engenheiro de programação e desenvolvimento do sistema de automação.

b) Organização e clareza do Sistema de Automação

A programação elaborada por meio da MARPASI disponibiliza uma documentação que facilita a organização e dá clareza ao sistema de automação, permitindo um maior entendimento do sistema de automação por outras pessoas, além do próprio engenheiro de automação.

Este índice de qualidade de software nem sempre é encontrado em programas Ladder uma vez que alguns programadores de CLPs não empregam a cultura de documentação de software . A MARPASI acaba de certa forma impondo essa cultura.

c) Facilidade para alteração e manutenção

O programa Ladder gerado por meio da aplicação da MARPASI também apresenta uma maior facilidade de alteração, uma vez que as modificações devem ser feitas nas Redes de Petri que apresentam uma maior facilidade de modelagem do sistema. Nas Redes de Petri a identificação do que é necessário para que o sistema vá para determinada condição é verificada pela simples análise dos seus pré-sets de posição, enquanto que em um programa Ladder pode ser necessária a verificação de várias linhas do programa.

d) Tempo e esforço para elaboração, testes e validação

Com relação ao tempo e esforço mensuráveis para o desenvolvimento do projeto de automação, os atuais resultados obtidos neste trabalho de pesquisa induzem que a aplicação da MARPASI pode gerar uma significativa economia, isto porque, mesmo que aparentemente, a MARPASI introduz mais etapas em um processo de automação, a mesma permite chegar a um resultado final confiável, de fácil entendimento e conseqüentemente com menos investimentos de tempo e esforço.

e) Verificação da assertividade do programa Ladder.

Por meio das ferramentas de simulação ou mesmo por meio da aplicação das propriedades das redes de petri, pode ser verificar a assertividade do programa Ladder.

6 CONCLUSÕES

Este trabalho de pesquisa teve como objetivo apresentar o desenvolvimento, testes e validação da MARPASI.

O capítulo 1 apresenta uma introdução geral com os objetivos e as motivações que ensejaram no desenvolvimento deste trabalho de pesquisa. O capítulo 2 introduz uma breve teoria geral das Redes de Petri, suas propriedades e as técnicas de análise de sistemas modeláveis pela mesma. O capítulo 3 apresenta uma introdução sobre Controladores Lógicos Programáveis e sua linguagem de programação Ladder. O capítulo 4 apresenta o desenvolvimento da MARPASI com todas as suas etapas. O capítulo 5 demonstra a viabilidade de uso da MARPASI por meio de estudo de caso.

Os testes de validação da MARPASI permitem concluir que a mesma pode ser empregada como ferramenta de auxílio da engenharia de automação para otimizar o processo de automação desde a pré-venda até a manutenção do projeto.

A MAPARSI pode contribuir para a diminuição do tempo de programação dos algoritmos de controle elaborados por meio da metodologia tradicional, visto que, a mesma permite a elaboração de programas com menos erros ou equívocos. Isto se deve ao fato de que na metodologia tradicional de programação as etapas de definição, análise, especificação e validação dos requisitos são feitas de maneira simplificada e geralmente não são utilizados métodos e ferramentas de engenharia eficazes.

A MARPASI contribui também para facilitar a detecção de erros no algoritmo de controle do sistema de automação, pois, além da utilização da engenharia de requisitos que define o sistema de automação de forma estruturada e documentada o sistema deve ser modelado e analisado por meio das Redes de Petri

que serve como mais um refinamento do sistema, fazendo com que o engenheiro de automação tenha mais informações quanto ao desempenho operacional do sistema modelado.

A MARPASI pode contribuir também para tornar as modificações do algoritmo de controle menos trabalhosas, visto que, na etapa 2 da mesma (análise, especificação e validação) é gerada toda documentação do sistema. Nesta documentação são relatados todos problemas e soluções acordadas entre os usuários e a engenharia de automação, facilitando desta forma a manutenção e eventuais modificações no algoritmo de controle. A documentação de software possibilita também o conhecimento das principais propriedades e limitações do sistema de automação e também o arquivamento eficiente da cultura do projeto.

Outra contribuição importante da MARPASI é o fato da mesma permitir que o algoritmo de controle seja simulado, analisado e validado por meio das propriedades descritas na teoria das Redes de Petri e também por software dedicados a simulação da mesma.

Com isto, pode-se concluir que as principais contribuições da MARPASI: diminuir o tempo total de programação, facilitar detecção de erros, tornar as modificações menos trabalhosas e possibilitar a simulação, análise e validação do sistema de automação que utilizam CLPs.

Assim sendo a aplicação da MARPASI permite uma diminuição do custo de programação e de manutenção do sistema de automação

6.1 SUGESTÕES DE COMPLEMENTAÇÃO A ESTE TRABALHO

Pode-se sugerir os seguintes trabalhos futuros como complementação a este:

- Validar a metodologia MARPASI em outros sistemas de automação a fim de confirmar a eficácia da MARPASI.
- Incrementar a metodologia para modelagem de sistemas híbridos, isto é, sistemas de variáveis contínuas (SVC) e sistemas a eventos discretos (SED), pois, na prática, estão ficando cada vez mais freqüentes os sistemas de controle que tratam em conjunto estas duas classes de controle (SVC e SED).
- Implementar a metodologia de forma a permitir o reuso de uma Rede de Petri cujas propriedades são conhecidas, pois uma mesma Rede de Petri pode ser o modelo de diversos sistemas.
- Pesquisar o desenvolvimento da metodologia com outros tipos de Redes de Petri (Rede de Petri colorida, Rede de Petri orientada a objetos, entre outras) que estão sendo desenvolvidas com o intuito de aumentar sua aplicabilidade a sistemas variados e aumentar o poder das mesmas.
- Desenvolver uma metodologia para mensurar de forma objetiva os ganhos de produtividade em projetos de automação quando é empregada a MARPASI.
- Permitir o desenvolvimento de um conjunto modular de bibliotecas de sistemas de automação em Ladder para otimizar os processos de automação.

Assim sendo, acredita-se que este trabalho de pesquisa tenha atingido os seus objetivos iniciais, inclusive contribuir para o contínuo desenvolvimento da ciência e da tecnologia da automação de processos.

7 REFERÊNCIAS BIBLIOGRÁFICAS

- [01] LATHA, K.; UMAMAHESWARI, B. **Supervisory control of an automated system with Ladder logic programming and analysis using Petri Net.** Conference on Systems, Man and Cybernetics. IEEE. v. 3, Page(s):5 pp, 2002
- [02] LUCAS, M. R.; TILBURY, D. M. **Quantitative and qualitative comparisons of CLP programs for a small testbed with a focus on human issues.** Proceedings of the American Control Conference, Anchorage, AK, 08/05/2002.
- [03] VENKATESH, K.; ZHOU, M; CAUDILL, R. J. **Comparing Ladder logic diagrams and Petri Net for sequence controller design through a discrete manufacturing system.** IEEE Transactions on Industrial Electronics, v. 41, n. 6, 1994.
- [04] MORAES, C.C.; CASTRUCCI, P.L. **Engenharia de Automação Industrial.** LTC – Livros Técnicos e Científicos Editora, 2001.
- [05] FONSECA, M. **Motivação para sistemas abertos.** Atan Sistemas.
- [06] HUUCK, R. **Software Verification for Programmable Logic Controllers.** 2003. Dissertation. der Technischen Fakultät der Christian-Albrechts-Universität zu Kiel .17/04/2003.
- [07] PALOMINO, R.C. **Uma Abordagem para a Modelagem, Análise e Controle de Sistemas de Produção Utilizando Redes de Petri.** Dissertação(Mestrado) - Universidade Federal de Santa Catarina,1995.
- [08] JENSEN, Kurt. **Coloured Petri Nets and the invariant-Method.** Theoretical Computer Science, vol. 14, p. 317-336, 1981.
- [09] MURATA, Tadao. **Petri Nets: Properties, Analysis and Applications.** Proceedings of the IEEE, vol. 77, n. 4, p. 541-580,1989.
- [10] PETERSON, J. L. **Petri Net Theory and the modelling of system.** Prentice-Hall Editions,1981.
- [11] JENSEN, Kurt. **Coloured Petri Net.** Lecture notes in Computer Science, n. 254, p. 248-299,1986.
- [12] DAVID, R.; ALLA, Hassane. **Petri Nets for Modelling of Dynamic System- A Survey.** Automática, vol. 30, n. 2, p. 175-202,1994.

- [13] SILVEIRA, P. R., SANTOS, W. E. **Automação e Controle Discreto**. Editora Érica. 1º edição. São Paulo, 1999.
- [14] NATALE, F. **Automação Industrial**. Ed. Érica. 1ª Edição. São Paulo, 2000.
- [15] OLIVEIRA, J. C. P. **Controlador Lógico Programável**. Editora Makron Books. São Paulo, 1993.
- [16] BIGNELL, J. W.; DONOVAN, R. L. **Eletrônica Digital**, Editora Makron Books, vol. 1 e 2, São Paulo, 1995.
- [17] JAFARI, M.; BOUCHER, T. O. **A rule-based system for generating a Ladder logic control program from a high-level systems model**. Journal of Intelligent Manufacturing, vol. 5, nº. 2, p.103-120, 1994.
- [18] VENKATESH, K.; ZHOU, M.C. **Modeling, Simulation, and control of Flexible manufacturing Systems – A Petri Net Approach**. World Scientific Publishers, Singapore, 1998.
- [19] PENG, S. S.; ZHOU, M. C. **Ladder Diagram and Petri Net Based Discrete Event Control Design Methods**, IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, 1-9, 2004.
- [20] LEE, J. S.; HSU, P. L. **A New Approach to Evaluate Ladder Logic Diagrams and Petri Nets via the IF-THEN Transformation**”. Conference on Systems, Man and Cybernetics. IEEE, p. 2711-2716, 2001.
- [21] PENG, S. S.; ZHOU, M. C. **Conversion between Ladder Diagrams and Petri Net in Discrete Event Control Design – A Survey**. In: IEEE Conference on Systems, Man and Cybernetics. v.4, p. 2682– 2687, 7-10 Oct. 2001
- [22] JIMÉNES, I.; LOPÉS, E.; RAMÍRES, A. **Synthesis of Ladder Diagrams from Petri Nets Controller Models**. Proceedings of the 2001 IEEE International Symposium on Intelligent Control, p.225-230, México City, México, 2001.
- [23] JONES, A. H.; UZAM, M.; AJLOUNI, N. **Design of discrete event control systems for programmable logic controllers using T-timed Petri nets**. Proc. 1996 IEEE Int. Symp. Computer-Aided Control System Design, Dearborn, MI, p. 212-217, 1996.

- [24] BOEHM, B. W. **Guidelines for Verifying and Validation software requirements and design Specification**. P.A Samet (editor). Proc Euro IFIP 79. North-Holland Publishing Company. Amsterdam, New York, p. 711-719. Sept, 1979.
- [25] FREY, G; LITZ, L. **Formal Methods in CLP Programming**. Proc. IEEE Conf. on Systems Man and Cybernetics, SMC 2000, Nashville(TN), p. 2431-2436, oct 2000.
- [26] PELED, D. A. **Software Realibity Methods**. Springer, Berlim, New York, 2001.
- [27] FREY, G; LITZ, L; KLEIN, STÉPHANE. **A Petri Net based Approach to the Development of correct Logic Controllers**. Institute of Automatic Control University of Kaiserslautern, Germany.
- [28] FREY, G; LITZ, L. **Correctness Analysis of Petri Net Based Logic Controllers**. Proc. American Control Conference, ACC 2000, Chicago(IL), p. 3165-3166, June 2000.
- [29] DAVID, R.; ALLA H. **Petri Nets and Grafcet - Tools for Modelling Discrete Event Systems**. Prentice Hall. New York, London, 1992.
- [30] DESROCHERS, A.A; AL-JAAR, R. Y. **Application of Petri Nets in Manufacturing Systems: Modelling, Control and Performance Analysis**. IEEE Press, Piscataway, USA, 1994.
- [31] KOTONYA, G.; SOMMERVILLE, I. **Requirements engineering: process and techniques**. Chichester: John Wiley & Sons, 2940, 1998.
- [32] MARTINS, L. E. G. **Uma metodologia de elicitação de requisitos de software baseada na teoria da atividade**. Tese (Doutorado). Universidade de Campinas(Unicamp).SP, 2001.
- [33] JUNIOR, L. M.; PEREIRA, S. L. **Controladores Lógicos Programáveis – Laboratório de Automação**. Escola Politécnica da Universidade de São Paulo. SP, 2006.
- [34] VENKATESH, K.; ZHOU, M.; CAUDILL, R.J. **Discrete Event Control Design for Manufacturing Systems Via Ladder Logic Diagrams and Petri Nets: A Comparative Study**. New Jersey Institute of Technology. USA, 1995.