

ESTRATÉGIA DE MODELAGEM DA
TAREFA DE PROGRAMAÇÃO
REATIVA DA PRODUÇÃO

UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**ESTRATÉGIA DE MODELAGEM DA TAREFA DE PROGRAMAÇÃO
REATIVA DA PRODUÇÃO**

Flavio Aldrovandi Montoro

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do Título de Mestre em Ciência da Computação.

Orientador: **PROF. DR. ORIDES MORANDIN JUNIOR**

SÃO CARLOS
2009

**Ficha catalográfica elaborada pelo DePT da
Biblioteca Comunitária da UFSCar**

M798em

Montoro, Flavio Aldrovandi.

Estratégia de modelagem da tarefa de programação reativa da produção / Flavio Aldrovandi Montoro. -- São Carlos : UFSCar, 2011.

156 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2009.

1. Inteligência artificial. 2. Engenharia do conhecimento. 3. Programação reativa da produção. 4. Conhecimento tácito. 5. Apoio à decisão. 6. Representação do conhecimento. I. Título.

CDD: 006.3 (20^a)


Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

**“Estratégia de Modelagem da Tarefa de
Programação Reativa de Produção”**

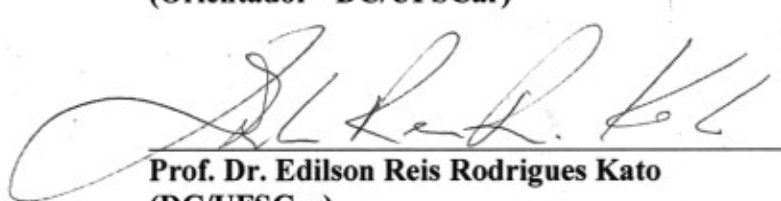
FLAVIO ALDROVANDI MONTORO

Dissertação de Mestrado apresentada ao
Programa de Pós-Graduação em Ciência da
Computação da Universidade Federal de São
Carlos, como parte dos requisitos para a
obtenção do título de Mestre em Ciência da
Computação

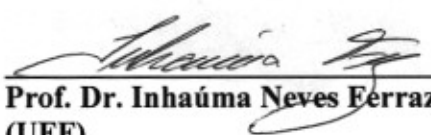
Membros da Banca:



Prof. Dr. Orides Morandin Júnior
(Orientador - DC/UFSCar)



Prof. Dr. Edilson Reis Rodrigues Kato
(DC/UFSCar)



Prof. Dr. Inhaúma Neves Ferraz
(UFF)

São Carlos
Agosto/2009

Dedico este trabalho aos meus pais, Edson e Mirnei; a minha namorada, Debora, por todo amor e carinho; a minha irmã, Liege; e a minha avó, Eunice.

Agradecimento

Agradeço aos meus pais, Edson e Mirnei, por todo suporte que me deram em todos esses anos acompanhados de muito amor, carinho e dedicação. Por terem acreditado e investido na minha formação pessoal e profissional.

A minha irmã, Liege, por todo amor, carinho e paciência que ela sempre teve comigo.

A minha vizinha querida, Eunice Caranguejo, por toda a ternura e afeto.

Ao meu avô, Antônio, por toda a ajuda.

A minha namorada, Debora, por todo amor, ternura. Pelo companheirismo e suporte, principalmente nos momentos mais difíceis.

Ao meu orientador, Orides, por sempre auxiliar, orientar e ter dado todo o suporte necessário para a realização desse trabalho e para meu crescimento pessoal e profissional.

A minha prima, Elis, pelo companheirismo nessa nossa longa jornada, por estar sempre disposta a me ajudar, tornando esses anos mais alegres.

Ao meu amigo Rafael pela amizade e por ter me apoiado o tempo em que ficou por aqui.

Aos meus amigos Danilo, Élen e Mayra pelas tardes agradáveis no café.

Ao meu amigo Ageu, pela ajuda durante o mestrado.

Ao meu amigo Marcos, pela grande ajuda na implementação.

Aos amigos que fiz no mestrado, por todo companheirismo, apoio, noites em claro estudando juntos, por ter tornado essa passagem por São Carlos mais agradável.

Resumo

Buscando uma vantagem competitiva, as empresas têm dado maior atenção à engenharia do conhecimento, visando o uso de técnicas para extrair e representar os conhecimentos já existentes, porém algumas vezes inexplorados ou explorados de maneira inadequada. A proposta deste trabalho é apresentar uma estratégia de modelagem do conhecimento no âmbito da programação reativa da produção. Essa estratégia de modelagem visa realizar a aquisição tanto do conhecimento tácito quanto do explícito, para representá-los de uma maneira que obtenha resultados mais próximos da realidade, visando também um tipo de representação que seja de fácil compreensão para a validação dos especialistas do domínio. Com o objetivo de validar o modelo proposto, foi desenvolvido um sistema computacional, para avaliar o comportamento do sistema perante dados do domínio e verificar se os resultados obtidos por ele estão compatíveis com o desejado. Esse modelo visa auxiliar a tomada de decisão dos especialistas perante eventos não programados que possam ocorrer durante a execução da programação, possibilitando a tomada de decisão do especialista, de uma maneira mais abrangente, pelo fato de considerar o conhecimento de vários especialistas e também possibilitando a tomada de decisão pelo operador na falta do especialista da área. A validação desse trabalho se deu em dois momentos, o primeiro sendo a validação do sistema computacional que foi realizado módulo por módulo, e depois a integração deles. O segundo foi a validação do modelo proposto em dois aspectos, o primeiro validado por meio dos casos de uso levantados tendo como base a fábrica presente no laboratório TEAR, e o outro aspecto foi a facilidade do entendimento dos modelos pelos especialistas, seguindo os mesmos métodos de representação utilizados em um projeto existente entre o laboratório TEAR e uma indústria.

Palavras-Chave: Engenharia do Conhecimento, Programação Reativa da Produção, Conhecimento Tácito, Tomada de Decisão, Representação do Conhecimento.

Abstract

Aiming at improving competitive advantage, organizations in general have been giving more attention to knowledge engineering, as well as its techniques and methods to acquire and represent knowledge already in place. This knowledge is sometimes not fully explored, or not explored properly. This approach is to develop a knowledge modeling strategy for reactive production scheduling, which focuses on tacit and explicit knowledge acquisition and representation. The knowledge representation aims at getting results closer to the organization reality and should also be comprehensible and easy to be validated by experts. To validate the proposed model, a computational system was developed to evaluate the model behavior under a specific domain, as well as to verify if the results are satisfactory. The model supports decision-making when unplanned events occur during the production process, enabling the possibility of evaluating greater knowledge to make the decision and also allowing the operator to make the decision without any expert support. The validation was performed in two steps; the first one, the computational system validation, was carried out by unit tests on every module, and after that, an integration test was performed. The second one, the proposed model validation, was verified in two ways. The first one was the validation using the industry use cases based on industry platform existing in the TEAR laboratory; the acceptance and understanding of the models by experts were verified using the same models already validated in a project between an industry and the TEAR laboratory.

Keywords: Knowledge Engineering, Reactive Production Scheduling, Tacit Knowledge, Decision-making, Knowledge Representation.

Lista de Figuras

Figura 2.1 –Valor agregado pela ótica da engenharia do conhecimento: Dados, informação e conhecimento.....	23
Figura 2.2 Processo de entrevista (adaptado de Kendal e Creen, 2007).	25
Figura 2.3 Representação de um conhecimento por redes semânticas.....	27
Figura 2.4 Exemplo de Frames.....	28
Figura 2.5 Exemplo de uma árvore de decisão binária.	29
Figura 2.6 Exemplo de lógica proposicional.	30
Figura 2.7 Exemplo da codificação de um cromossomo para a representação do conhecimento. (adaptado de Deriz, 2007).	32
Figura 2.8 Exemplo de conjuntos <i>Fuzzy</i>	33
Figura 2.9. Exemplo de condições para a ativação de uma transição.	36
Figura 2.10 Exemplos de elementos do fluxograma com seus significados	38
Figura 2.11. Exemplo de uma atividade representada por um fluxograma.	38
Figura 2.12 Mapa do conhecimento representando a estrutura de apresentação dos trabalhos de pesquisa.....	40
Figura 3.1 Exemplo de modelagem do conhecimento sobre o roteiro de fabricação do produto P1 em redes de Petri.	53
Figura 3.2 Fluxograma da Ocorrência quebra de máquina.	55
Figura 3.3 Funcionalidade macro do uso do sistema.....	57
Figura 3.4 Fluxograma do tratamento da ocorrência de inclusão/exclusão de produtos.....	58
Figura 3.5 Fluxograma do modulo tratamento da ocorrência de falta de local para estocagem.	59
Figura 3.6 Fluxograma do tratamento da ocorrência de falta de matéria prima.....	60
Figura 3.7 Fluxograma do tratamento da ocorrência de esvaziar <i>buffer</i> de matéria prima.	62
Figura 3.8 Fluxograma do tratamento da ocorrência de falta de operador.....	63
Figura 3.9 Sistema do módulo <i>fuzzy</i> – (adaptado de REZENDE, 2003).....	64
Figura 3.10 Conjuntos da variável de entrada "nível de estoque".....	64
Figura 3.11 Conjuntos da variável de entrada "data devida".	65
Figura 3.12 Conjuntos da variável de entrada "Prioridade Inicial".	65
Figura 3.13 Conjuntos da variável de entrada "margem de contribuição".	65

Figura 3.14 Conjuntos da variável de saída "margem de contribuição".	66
Figura 3.15 Arquitetura do Sistema.	68
Figura 3.16 Fluxograma da manutenção dos dados.	69
Figura 4.1 Arranjo físico da fábrica.	71
Figura 4.2 Representação do roteiro de fabricação do produto P1.	74
Figura 4.3 Arquitetura do Sistema Computacional.	75
Figura 4.4 Arquitetura do módulo de programação reativa.	76
Figura 4.5 Comunicação entre as ferramentas de desenvolvimento.	77
Figura 4.6 Modelo entidade relacionamento contemplando a base de dados desenvolvida.	77
Figura 4.7 Interface de configuração do programador.	78
Figura 4.8 Interface do Operador e Programador com o sistema.	79
Figura 4.9 Saída apresentada pelo sistema.	79
Figura 5.1 Mapa do conhecimento representando as funcionalidades e os módulos que as compõem.	82
Figura 5.2 Relatório sobre o teste do módulo AvaliarRoteiros.	84
Figura 5.3 Diagrama de seqüência entre os módulos pertencentes à ocorrência quebra de máquina.	85
Figura 5.4 Relatório sobre o teste de integração dos módulos pertencentes à ocorrência quebra de máquina.	86
Figura 5.5 Fluxograma do tratamento da ocorrência: Quebra de máquina.	88
Figura 5.6 Programação inserida no sistema computacional.	90
Figura 5.7 Escolha da opção "Quebra de Máquina".	90
Figura 5.8 Escolha para bloquear os recursos M13 e M22.	91
Figura 5.9 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.	91
Figura 5.10 Fluxograma do tratamento da ocorrência: Falta de matéria prima.	93
Figura 5.11 Programação inicial inserida no sistema computacional.	94
Figura 5.12 Escolha da opção "Falta de Matéria Prima".	95
Figura 5.13 Escolha para indicar a falta da matéria prima MPA.	95
Figura 5.14 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.	96
Figura 5.15 Fluxograma do tratamento da ocorrência: Falta de estocagem.	97
Figura 5.16 Programação inicial inserida no sistema computacional.	98
Figura 5.17 Escolha da opção "Falta de Local para Estocagem".	99

Figura 5.18 Escolha do produto que está sem local para estocagem.....	99
Figura 5.19 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.	100
Figura 5.20 Fluxograma do tratamento da ocorrência: Falta de operador.....	101
Figura 5.21 Programação inicial inserida no sistema computacional.	102
Figura 5.22 Escolha da opção "Falta de Operador".....	103
Figura 5.23 Escolhas de quantos operadores estão disponíveis.	103
Figura 5.24 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.	104
Figura 5.25 Fluxograma do tratamento da ocorrência "Esvaziar <i>Buffer</i> de entrada".	107
Figura 5.26 Programação inicial inserida no sistema computacional.	112
Figura 5.27 Definição da lista de produtos preferenciais.	112
Figura 5.28 Definição do “nível de estoque” e “margem de contribuição” de cada produto.	113
Figura 5.29 Definição da quantidade de matéria prima nos buffers de entrada.	113
Figura 5.30 Definição da quantidade de operadores disponíveis.	114
Figura 5.31 Escolha do tratamento da ocorrência: Esvaziar <i>buffer</i> de matéria prima.....	114
Figura 5.32 Programação reativa da produção, após tratamento da ocorrência: “Esvaziar <i>buffer</i> de entrada”.	115
Figura 5.33 Fluxograma da funcionalidade: "Exclusão Dura".	116
Figura 5.34 Programação inicial inserida no sistema computacional.	117
Figura 5.35 Escolha da ocorrência: "Exclusão Dura".	117
Figura 5.36 Seleção dos produtos a serem removidos da programação.....	118
Figura 5.37 Apresentação da programação reativa da produção.....	118
Figura 5.38 Fluxograma da funcionalidade: "Inclusão Dura".	119
Figura 5.39 Programação inicial inserida no sistema computacional.	120
Figura 5.40 Escolha da ocorrência: "Inclusão Dura".	121
Figura 5.41 Inserção do produto P6.	121
Figura 5.42 Apresentação da programação reativa da produção.....	122
Figura 8.1 Modelo do roteiro de fabricação do produto P1.....	139
Figura 8.2 Modelo do roteiro de fabricação do produto P2.....	140
Figura 8.3 Modelo do roteiro de fabricação do produto P3.....	141
Figura 8.4 Modelo do roteiro de fabricação do produto P4.....	142
Figura 8.5 Modelo do roteiro de fabricação do produto P5.....	143
Figura 8.6 Modelo do roteiro de fabricação do produto P6.....	144

Figura 8.7 Modelo do roteiro de fabricação do produto P7.....	145
Figura 8.8 Modelo do roteiro de fabricação do produto P8.....	146
Figura 8.9 Modelo do roteiro de fabricação do produto P9.....	147
Figura 8.10 Modelo do roteiro de fabricação do produto P10.	148
Figura 8.11 Modelo do roteiro de fabricação do produto P11.	149
Figura 8.12 Modelo do roteiro de fabricação do produto P12.	150
Figura 8.13 Modelo do roteiro de fabricação do produto P13.	151
Figura 8.14 Modelo do roteiro de fabricação do produto P14.	152
Figura 8.15 Modelo do roteiro de fabricação do produto P15.	153
Figura 8.16 Modelo do roteiro de fabricação do produto P16.	154
Figura 8.17 Modelo do roteiro de fabricação do produto P17.	155
Figura 8.18 Modelo do roteiro de fabricação do produto P18.	156

Lista de Tabelas

Tabela 2.1 Descrição formal da redes de Petri (adaptado de Murata 1989).....	36
Tabela 2.2 Algumas interpretações típicas de lugares e transições (adaptado de Murata, 1989)	37
Tabela 3.1 Exemplo de uma tabela de aquisição do conhecimento sobre o roteiro de fabricação do produto P1.....	52
Tabela 3.2 Tabela contendo o conhecimento sobre a ocorrência quebra de máquina.....	53
Tabela 3.3 Caso de Uso da Ocorrência Quebra de Máquina.....	54
Tabela 3.4. Base de regras.....	66
Tabela 4.1 Roteiros de Fabricação	72
Tabela 4.2 Tabela de aquisição do conhecimento sobre roteiros de fabricação.....	72
Tabela 4.3 Tabela de aquisição do roteiro de fabricação do produto P1.....	73
Tabela 5.1 Programação inicial para a validação do tratamento da ocorrência: Quebra de máquina.	87
Tabela 5.2 Tabela de roteiros de fabricação dos produtos testados na validação do tratamento da ocorrência: Quebra de máquina.	88
Tabela 5.3 Programação reativa da produção para a validação do tratamento da ocorrência: Quebra de máquina.....	89
Tabela 5.4 Programação inicial para a validação do tratamento da ocorrência: Falta de matéria prima.....	92
Tabela 5.5 Programação reativa da produção para a validação do tratamento da ocorrência: Falta de matéria prima.	94
Tabela 5.6 Programação inicial para a validação do tratamento da ocorrência: Falta de local para estocagem.	96
Tabela 5.7 Programação reativa da produção para a validação do tratamento da ocorrência: Falta de local para estoque.....	98
Tabela 5.8 Programação inicial para a validação do tratamento da ocorrência: Falta de operador.....	100
Tabela 5.9 Programação reativa da produção para a validação do tratamento da ocorrência: Falta de Operador.	102

Tabela 5.10 Programação inicial para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	104
Tabela 5.11 Quantidade e tipo de matéria prima presente no <i>buffer</i> de entrada	105
Tabela 5.12 Lista de produtos Preferenciais para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	105
Tabela 5.13 Tabela de Margem de contribuição dos produtos.....	106
Tabela 5.14 Lista de programação intermediária para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	109
Tabela 5.15 Lista de programação intermediária para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	110
Tabela 5.16 Lista de programação intermediária para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	110
Tabela 5.17 Lista de programação intermediária após cálculo <i>fuzzy</i> para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	110
Tabela 5.18 Lista de programação intermediária após alocação dos operadores para a validação do tratamento da ocorrência: Esvaziar <i>Buffer</i> de matéria prima.	111
Tabela 5.19 Programação inicial para a validação do tratamento da ocorrência: Exclusão dura.	115
Tabela 5.20 Programação reativa para a validação do tratamento da ocorrência: Exclusão dura.	116
Tabela 5.21 Programação inicial para a validação do tratamento da ocorrência: Inclusão dura.	119
Tabela 5.22 Programação reativa para a validação do tratamento da ocorrência: Inclusão dura.	120
Tabela 8.1 Tabela de aquisição do roteiro de fabricação do produto P1.....	131
Tabela 8.2 Tabela de aquisição do roteiro de fabricação do produto P2.....	131
Tabela 8.3 Tabela de aquisição do roteiro de fabricação do produto P3.....	132
Tabela 8.4 Tabela de aquisição do roteiro de fabricação do produto P4.....	132
Tabela 8.5 Tabela de aquisição do roteiro de fabricação do produto P5.....	132
Tabela 8.6 Tabela de aquisição do roteiro de fabricação do produto P6 roteiro 1.	133
Tabela 8.7 Tabela de aquisição do roteiro de fabricação do produto P6 roteiro 2.	133
Tabela 8.8 Tabela de aquisição do roteiro de fabricação do produto P7.....	133
Tabela 8.9 Tabela de aquisição do roteiro de fabricação do produto P8.....	134
Tabela 8.10 Tabela de aquisição do roteiro de fabricação do produto P9.....	134

Tabela 8.11 Tabela de aquisição do roteiro de fabricação do produto P10.....	134
Tabela 8.12 Tabela de aquisição do roteiro de fabricação do produto P11.....	135
Tabela 8.13 Tabela de aquisição do roteiro de fabricação do produto P12 roteiro 1.	135
Tabela 8.14 Tabela de aquisição do roteiro de fabricação do produto P12 roteiro 2.	135
Tabela 8.15 Tabela de aquisição do roteiro de fabricação do produto P13.....	136
Tabela 8.16 Tabela de aquisição do roteiro de fabricação do produto P14.....	136
Tabela 8.17 Tabela de aquisição do roteiro de fabricação do produto P15.....	136
Tabela 8.18 Tabela de aquisição do roteiro de fabricação do produto P16.....	137
Tabela 8.19 Tabela de aquisição do roteiro de fabricação do produto P17.....	137
Tabela 8.20 Tabela de aquisição do roteiro de fabricação do produto P17 roteiro 1.	137
Tabela 8.21 Tabela de aquisição do roteiro de fabricação do produto P17 roteiro 2.	138

Lista de Abreviaturas e Siglas

AG	Algoritmo genético
ASCII	Código Padrão Americano para o Intercâmbio de Informação (<i>American Standard Code for Information Interchange</i>)
AGV	Veículos auto guiados (<i>Automated Guided Vehicle</i>)
CSP	Problema de satisfação das restrições (<i>Constraints Satisfaction Problem</i>)
FMS	Sistema flexível de manufatura (<i>Flexible Manufacturing Systems</i>)
KBI	Aperfeiçoamento baseado em conhecimento (<i>Knowledge-Based Improvement</i>)
KCOM	Componente de conhecimento (<i>Knowledge COMponent</i>)
KDD	Descoberta de conhecimento em base de dados (<i>Knowledge Discovery Databases</i>)
MAP	Plataforma de análise para viabilizar a fabricação (<i>Manufacturability Analysis Platform</i>)
MFIKM	Modelo de conhecimento e informação de instalações de manufatura (<i>Manufacturing Facility Information and Knowledge Model</i>)
MTO	Produção sob encomenda (<i>Make To Order</i>)
OAR	Relação-objeto-atributo (<i>Object-Attribute-Relation</i>)
PDM	Gerenciadores de dados do produto (<i>Product Data Management</i>)
SCCP	Sistema de concorrência de processo cíclico (<i>System of Concurrent Cyclic Process</i>)
SSS	Sistema de suporte a programação (<i>Scheduling Support System</i>)
UML	Linguagem de modelagem unificada (<i>Unified Model Language</i>)
VIS	Simulação interativa visual (<i>Visual Interactive Simulation</i>)

Sumário

1	Introdução	16
2	Engenharia do conhecimento	22
2.1	Teoria Básica.....	22
2.1.1	Dados, Informação e Conhecimento.....	23
2.1.2	Aquisição do Conhecimento.....	24
2.1.3	Representação do Conhecimento	25
2.2	Trabalhos de Pesquisa	39
2.2.1	Pesquisa em Diferentes Domínios.....	40
2.2.2	Pesquisas em Domínios Correlatos	45
3	Proposta	51
3.1	O Modelo Proposto	51
3.2	Funcionalidades	55
3.3	Arquitetura do Sistema	68
3.4	Formas de validação.....	70
4	Aplicação do Modelo.....	71
4.1	Cenário de aplicação	71
4.2	Aquisição e Representação dos Roteiros de Fabricação	72
4.3	Sistema Computacional	74
5	Validação.....	81
5.1	Validação da Implementação e da Lógica Estrutural dos Módulos.....	83
5.2	Validação da lógica de tratamento das ocorrências	87
5.2.1	Teste de validação do tratamento da ocorrência: Quebra de Máquina	87
5.2.2	Teste de validação do tratamento da ocorrência: Falta de Matéria Prima	92
5.2.3	Teste de validação do tratamento da ocorrência: Falta de Local para Estocagem.....	96
5.2.4	Teste de validação do tratamento da ocorrência: Falta de Operador	100
5.2.5	Teste de validação do tratamento da ocorrência: Esvaziar Matéria Prima do Buffer ..	104
5.2.6	Teste de validação da funcionalidade: Exclusão Dura.....	115
5.2.7	Teste de validação da funcionalidade: Inclusão Dura	119
6	Conclusão	123
6.1	Análise dos Resultados	124
6.2	Trabalhos Futuros	124
7	Referências	126

8	<i>Apêndice A – Aquisição e Representação dos Roteiros de Fabricação</i>	131
----------	---	------------

1 Introdução

O cenário industrial que temos hoje é muito disputado pelo elevado número de empresas atuando no mesmo mercado, e pelo avanço das técnicas utilizadas por elas. Com a ajuda da tecnologia, tem-se um aumento na quantidade de dados coletados, e com esse avanço, uma melhor análise dos mesmos.

Essa melhor qualidade da informação sobre o processo de cada empresa gera a possibilidade de um melhor conhecimento sobre a organização (FUGATE; STANK; MENTZER, 2009). Porém, nem sempre é utilizado por inteiro pelo simples fato de que dependendo do tipo do conhecimento, fica difícil extrair e manipulá-lo corretamente.

O conhecimento explícito é bem utilizado por ser um conhecimento de fácil aquisição e representação, porém nem sempre esse conhecimento, por si só, consegue retratar o domínio de maneira correta, necessitando o apoio do conhecimento implícito para complementá-lo.

O conhecimento implícito muitas vezes é descartado por ser de difícil aquisição e representação. Muitos modelos levam em consideração apenas a parte explícita do conhecimento, tornando-se algumas vezes pouco representativos ou eficientes.

Agregando ao modelo o valor do conhecimento implícito, tem-se muito a ganhar. Um grande exemplo disso é o simples fato de permitir que o conhecimento pertença à organização e não dependa exclusivamente de uma pessoa. Isso porque quando essa pessoa sair da empresa levará consigo todo o conhecimento adquirido com a experiência durante anos de trabalho, e para a empresa esse conhecimento ficará perdido.

Outra vantagem é o fato do modelo se aproximar mais do real, ficando mais semelhante à maneira que é executado no dia-a-dia da indústria. Esses procedimentos podem até, algumas vezes, ser otimizados com a ajuda da análise desses modelos.

Com o avanço da tecnologia é possível criar sistemas para auxiliar esses processos, como por exemplo, o de manufatura de um produto, otimizando e algumas vezes até aumentando a gama de variáveis analisadas, tornando o processo mais assertivo.

Agregando a aquisição e a representação dos conhecimentos individual e organizacional com os sistemas computacionais, é possível criar modelos que analisem de maneira mais eficiente, em termos de qualidade e tempo, as variáveis que são analisadas pelos humanos. Também torna possível a representação do conhecimento usado, viabilizando um refinamento no próprio conhecimento pelo fato de não centralizá-lo em uma única pessoa.

Uma área que se pode aplicar essa agregação de técnicas é na programação reativa da produção que, segundo Li e Ierapetritou (2008), é a atividade responsável por monitorar a execução da programação previamente planejada, lidando com eventuais problemas em sua execução.

Esses problemas podem ser devidos a eventos não esperados, que se não tratados podem vir a causar atrasos na execução da programação planejada (JANAK *et al.*, 2006). Esses atrasos trazem como consequência o não cumprimento da entrega dos produtos nas datas devidas, atrapalhando assim o relacionamento com o cliente.

Essa programação reativa da produção, se bem executada, pode assegurar uma boa qualidade em um curto período de tempo (TANG; WANG, 2008), melhorando a competitividade dos produtos, caso seja identificada a programação reativa que gere melhor desempenho (SUN; XUE, 2001).

Nesse ambiente de produção, o grupo de pesquisa do laboratório TEAR, Laboratório de Pesquisa e Inovação em Tecnologia e Estratégias de Automação, há alguns anos vem investigando várias formas de se obter uma melhor produção, visando à melhora dos processos por meio de técnicas de inteligência artificial ligadas ao problema inerente do contexto de produção.

Em Politano *et al.* (2000) apresentaram um sistema para solucionar em tempo real o problema da programação da produção em um ambiente flexível de manufatura. Essa programação engloba tanto as operações nos produtos quanto as operações de transporte, sendo estas consideradas simultaneamente por um sistema baseado em lógica *fuzzy*.

Morandín *et al.* (2000) adotaram a estratégia de modelagem de um sistema automatizado de manufatura por meio da técnica de Redes de Petri. Para superar os problemas de complexidade, inerentes a essa técnica, quando usada para modelar sistemas de grande porte, foi adotada uma estratégia modular considerando os recursos compartilhados e o planejamento de processos alternativos.

Em Carvalho *et al.* (2002) alegou que o uso da técnica de simulação pode ser usada para encontrar um cenário bom para o sistema de produção na quando ocorre um problema inesperado. Porém a simulação se torna inviável quando é necessário obter essa resposta em pouco tempo, pois existe inúmeras possibilidades por ser um problema de natureza combinatória. Para solucionar esse problema, foi apresentada uma técnica de redução de cenários baseada em lógica *fuzzy*, diminuindo assim a quantidade de cenários a serem simulados.

Maia *et al.* (2002) apresentou uma extensão do PPSS proposto por Kato (2000). Essa extensão consiste em analisar automaticamente o estado atual da fábrica e quando necessário executar a simulação para determinar a melhor reprogramação, reportando as alterações ao tomador de decisão. Com esse auxílio é possível obter uma reprogramação mais efetiva, levando em consideração os objetivos almejados.

Benicasa, Morandin e Kato (2003) propuseram um sistema *fuzzy* para realizar o despacho de veículos usando uma regra multicritérios que considera a distância entre o veículo auto guiado (AGV) e a estação de trabalho, o número de nós no percurso e o espaço restante no *buffer* de saída da estação de trabalho. A escolha por essas variáveis se deu após um estudo da literatura e de ter verificado que a maior parte dos trabalhos levantados usa a distância entre o AGV e a estação de trabalho e o número de espaços do *buffer* de saída. Já o número de nós foi considerado, pois a modelagem foi dividida em zonas nas quais apenas um AGV pode entrar de cada vez para evitar conflitos. O trabalho foi validado comparando-se o seu desempenho com o da regra FIFO, mostrando-se superior para o caso aplicado.

Castro *et al.* (2004) apresentaram um sistema *fuzzy* para realizar o sequenciamento da produção, sendo que para a criação da base de regras fuzzy, foi usado um algoritmo genético. O objetivo desse sistema *fuzzy* é definir o melhor sequenciamento visando diminuir o *lead time* e aumentar a produtividade. Para a validação foi proposto um sistema flexível de manufatura (FMS), no qual são produzidos 1200 peças, sendo estas divididas em 5 famílias distintas. Para esse cenário foi testado o sistema *fuzzy* com a geração de regras por meio do AG e outra criada pelo especialista. O resultado do sistema foi similar, porém a base de regra gerada pelo AG apresentou menos regras, se mostrando uma base mais enxuta.

Morandin e Kato (2005) propuseram uma abordagem usando uma variação de redes de petri, chamada de Redes de Petri Virtual (VPN), para superar o problema de modelagem de sistemas automatizados de manufatura. Esse problema se dá devido a complexidade e ao grande número de elementos, piorando ainda mais quando se avalia os recursos compartilhados. Com a VPN, é possível começar a modelagem por partes que contenham elementos necessários e com a estratégia modular, é possível criar ligações entre esses modelos realizando a representação de todo o sistema.

Para a validação desta abordagem, foi proposto um teste em um arranjo físico proposto por Morandin *et al.* (2000), composto de 6 máquinas CNC produzindo diferentes tipos de produtos, seguindo diferentes roteiros de produção. Também foi considerado o sistema de transporte composto por veículos auto guiados (AGVs). A proposta se mostrou

adequada para a representação modular de um sistema de manufatura automatizada, simplificando a análise por meio de modelos mais simples.

Morandin *et al.* (2006) apresentaram um sistema *fuzzy* para realizar o despacho de veículos em um sistema de manufatura. As regras *fuzzy*, que são responsáveis por determinar qual veículo irá atender a tarefa, foram construídas automaticamente por meio de um algoritmo genético, o qual considera um conjunto de treinamento previamente definido para criar uma base de regras *fuzzy* otimizada. A partir dessa otimização da base de regras, o sistema se torna mais rápido e assertivo, tendo em vista a necessidade dessa rápida resposta por se tratar de sistemas reativos de produção.

Foram realizadas simulações para validar o sistema proposto. Para tal experimento, foi usado um cenário proposto por Morandin *et al.* (2000). Os testes realizados provaram a maior velocidade na resposta obtida pelo sistema assim como a facilidade do entendimento das regras por meio dos especialistas.

Morandin *et al.* (2007) apresentaram uma estratégia de modelagem para o controle e o intertravamento de um sistema automático de manufatura (AMS – *automated manufacturing system*), usando redes de Petri virtuais (VPN – *virtual petri nets*) para realizar a conexão do nível estratégico com o nível de controle. Foi proposto a modelagem do sistema por meio de 8 estágios, iniciando com a modelagem do arranjo físico, passando pela definição dos intertravamentos, veículos de transporte e produtos, modelagem das máquinas e seus respectivos *buffers*, modelagem do fluxo de produção, ligação entre os módulos existentes, até chegar na etapa de simulação.

O modelo foi validado em um cenário proposto pelo autor e simulado com o auxílio da ferramenta CPN Tools. O resultado obtido demonstrou a possibilidade de ligar os modelos por meio da VPN para modelar os MAS mais complexos.

Morandin *et al.* (2008) propuseram um sistema de apoio à decisão para auxiliar o problema de sequenciamento da produção, especialmente quando ocorrem eventos não esperados. Esse sistema é responsável por filtrar os cenários possíveis e com isso diminuir o tempo de simulação para obter qual combinação é melhor naquele dado momento. Foi realizada uma análise de desempenho do sistema comparando a saída do sistema de apoio à decisão com os resultados obtidos por meio de simulação.

Foram avaliados 4 critérios de desempenho, sendo eles: escolha certa, ordenação do sequenciamento, classificação das sequências e primeira sequência. O resultado se mostrou satisfatório e de grande valia ao programador devido a drástica diminuição dos cenários a serem avaliados.

Para criar um modelo do conhecimento que auxilie a programação reativa da produção, pensou-se em unir, em uma única modelagem, tanto o conhecimento explícito quanto o implícito, para com isso, mapear de uma maneira mais próxima do real, como a programação reativa da produção é realizada pelos especialistas.

Também é importante ressaltar que algumas variáveis não são analisadas devido ao grande número de variáveis existentes, por algumas vezes possuírem alta complexidade ou até mesmo por não serem conhecidas pelos especialistas.

Avaliando esse argumento, para aumentar a assertividade da programação reativa julga-se necessário incorporar ao modelo algumas variáveis que, a princípio, o especialista do domínio não pondere.

O objetivo geral desse trabalho é apresentar uma estratégia de modelagem do conhecimento no âmbito da programação reativa da produção. Definir padrões para realizar a aquisição do conhecimento e sua representação, a fim de criar um modelo que seja capaz de representar o domínio desejado, atendendo todos os requisitos levantados.

Após a criação desse modelo que representa o domínio sugerido, se torna necessário validá-lo de acordo com o cenário estabelecido. Para isso foi proposto como objetivo específico desse trabalho o desenvolvimento de um sistema computacional de acordo com o modelo criado, seguindo os casos de uso levantados em conjunto com os especialistas.

Para a validação do modelo, primeiramente é necessário validar o sistema computacional. Após a validação do sistema, todos os casos de usos levantados devem ser testados, avaliando o comportamento do modelo, e verificando se ele atende ou não o caso de uso em questão.

Esse trabalho está dividido da seguinte forma, no capítulo 2 é apresentada uma parte da teoria sobre a engenharia do conhecimento, sendo essa dividida em dois momentos, no primeiro é apresentada a teoria básica para o entendimento da engenharia do conhecimento. No segundo momento são apresentados alguns trabalhos relacionados ao tema, alguns em domínios diferentes da proposta, porém com pontos importantes a contribuir para esse trabalho, e outros em domínios correlatos ao da proposta.

No terceiro capítulo é descrita a proposta, apresentando a estratégia de modelagem utilizada para modelar a programação reativa da produção. Também aborda a construção do sistema computacional baseado no modelo proposto. No capítulo 4, é definido o cenário para a aplicação do modelo, apresentando características desse cenário para que se possa aplicá-lo.

No quinto capítulo é apresentado o método de validação do modelo e do sistema computacional, são definidas diretrizes para realizar a validação de maneira que consiga comprovar a aderência do modelo ao domínio.

No sexto e último capítulo é apresentada a conclusão, apresentando os resultados obtidos, assim como a análise dos resultados e os trabalhos futuros.

2 Engenharia do conhecimento

2.1 Teoria Básica

Giarratano e Riley (1998) afirmam que filosoficamente existem, a princípio, dois tipos especiais de conhecimento expressos por Aristóteles, Platão, Descartes, Hume, Kant, e outros: *a priori* - que do latim significa "aquilo que precede", que seria um conhecimento adquirido por meio do pensamento dedutivo; e *a posteriori* - "que vem depois", que seria o conhecimento obtido depois de experiências.

O conhecimento pode ser classificado em três tipos, segundo Giarratano e Riley (1998): conhecimento procedural, conhecimento declarativo e conhecimento tácito.

O conhecimento procedural está relacionado a como fazer alguma coisa, por exemplo, como amarrar o tênis. Já o conhecimento declarativo está relacionado à veracidade de algum fato, por exemplo, saber que é verdade que quando colocamos a mão no fogo, irá queimar. O conhecimento tácito, também chamado de conhecimento inconsciente, é aquele conhecimento que é difícil ser expresso por uma linguagem, por exemplo, conseguimos mover a mão, mas não conseguimos expressar como fazemos isso.

Já Kendal e Creen (2007), além de classificar em conhecimentos procedural e declarativo, acrescentam mais uma classificação: meta-conhecimento, que está relacionado ao conhecimento sobre o conhecimento. Por exemplo, você vê na previsão do tempo que a temperatura está menos cinco graus, então sabe que está frio lá fora, logo você deduz que precisa usar um agasalho quando for sair.

Milton (2007), já apresenta uma classificação diferente, dividida em quatro tipos de conhecimento, sendo eles: procedural, concordando com os autores citados anteriormente; conceitual, que é o conhecimento relacionado a algo que você sabe como acontece; explícito, está relacionado às tarefas básicas realizadas pelos especialistas; e tácito, que é relacionado às experiências passadas.

A engenharia do conhecimento é um ramo responsável pela aquisição e representação do conhecimento. Citando Chorafas (1990), a engenharia do conhecimento é o processo de criação de construtos de Inteligência Artificial (IA).

Ainda baseado em Chorafas (1990), o engenheiro do conhecimento tem que encontrar meios de capturar, analisar e transformar o conhecimento na forma de regras. Já Dokas e Panagiotakopoulos (2006) afirmam que o engenheiro de conhecimento é aquele que

atua no processo de aquisição e representação do conhecimento, interagindo com o especialista do domínio específico para extrair a informação necessária e representando-as da maneira mais adequada.

As aplicações das técnicas de engenharia do conhecimento são amplamente utilizadas em várias áreas, sempre adquirindo o conhecimento do domínio e procurando formas de melhor representá-lo. As fases de aquisição e representação do conhecimento são feitas em paralelo conforme a necessidade do engenheiro.

2.1.1 Dados, Informação e Conhecimento.

Existe uma grande diferença entre os conceitos de dados, informação e conhecimento, sendo essa distinção de fundamental importância para a engenharia de conhecimento (Figura 2.1).

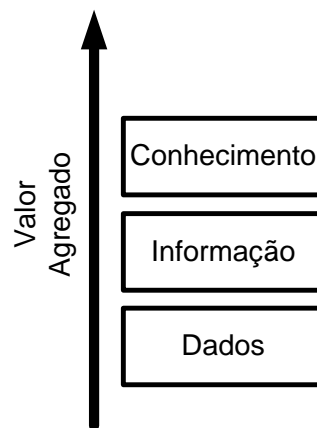


Figura 2.1 –Valor agregado pela ótica da engenharia do conhecimento: Dados, informação e conhecimento.

Kendal e Creen (2007) afirmam que não existe uma definição universal para dados, e fazem uma citação a Laudon e Laudon (1998) que explicita os dados como sendo um fluxo de fatos brutos representando eventos, sem ainda serem transformados de maneira que as pessoas possam usar e entender. Já Kasabov (1996), explica os dados como podendo ser símbolos sem um significado.

A informação, segundo Giarratano e Riley (1998), nada mais é do que os dados processados. Já Kasabov (1996), afirma que informação é qualquer dado estruturado que contenha um significado.

O conhecimento representa uma ou mais informações especializadas (GIARRATANO; RILEY, 1998). Segundo Kasabov (1996), o conhecimento nada mais é do que a informação condensada. Já Hayes (1992) *apud* Kendal e Creen (2007), afirma que o

conhecimento é o resultado do entendimento da informação. O conhecimento também leva em consideração as experiências anteriores.

Para facilitar a distinção e o entendimento desses três itens citados anteriormente, é dado o seguinte exemplo:

Temos esses números 677978726967737769788479. Essa seqüência de números são dados, pois os números não têm significado nenhum. Estruturando esses dados em grupos de dois números e considerando-os como decimais, teremos os dados transformados em informação (67 79 78 72 69 67 73 77 69 78 84 79). O conhecimento seria o resultado obtido unindo a informação citada anteriormente com o a informação de que esses números representam letras na tabela ASCII (*American Standard Code for Information Interchange*). A partir dessas informações, conseguimos traduzir esses números da seguinte forma:

67	79	78	72	69	67	73	77	69	78	84	79
C	O	N	H	E	C	I	M	E	N	T	O

Essa palavra obtida a partir das informações seria o conhecimento.

Dados, informação e conhecimento não são coisas estáticas, mas estágios no processo de uso e transformação de dados em conhecimento (KENDAL; CREEN, 2007). Esse conceito é muito utilizado para tomada de decisões, de maneira geral as pessoas estão sempre transformando dados em conhecimento para tomadas de decisão no dia-a-dia.

2.1.2 Aquisição do Conhecimento

Segundo Milton (2007) a aquisição do conhecimento está focada na criação de um depósito de conhecimento o qual poderá ser usado para dar suporte a um produto final que traz várias aplicações e benefícios.

A aquisição do conhecimento se dá pela colaboração entre o especialista da área e o engenheiro do conhecimento. É muito importante essa colaboração, pois é por meio dela que o engenheiro junto com o especialista, vão verbalizar o conhecimento da área para posteriormente representá-lo.

Existem várias maneiras de se obter esse conhecimento, e geralmente o engenheiro do conhecimento não se apóia em apenas uma delas. Um fator complicador nesse estágio é que não existe um método pré-definido para cada caso, é necessário que o engenheiro tenha a percepção para escolher os métodos mais adequados.

Nessa etapa, é necessário sempre procurar o maior número de fontes possíveis para que se tenha uma cobertura mais completa possível do conhecimento do domínio, pois

quanto mais completo for o conhecimento, melhor será o desempenho do sistema desenvolvido.

Dentre os métodos de aquisição existem os que são de maneira manual e os que são automáticos. Um método automático conhecido é o *Knowledge Discovery Database* (KDD), utilizado para capturar conhecimento de relações nas bases de dados.

Dos métodos manuais, o conhecimento pode ser extraído de várias fontes como, por exemplo: livros, estudos de casos, relatórios técnicos ou até mesmo de humanos. No caso de humanos, espera-se que eles sejam especialistas no assunto que se deseja obter o conhecimento.

Essa extração a partir dos especialistas pode ser por meio de entrevistas estruturadas, semi-estruturadas e não estruturadas, observação do dia-a-dia do especialista, conversas informais, questionários, entre outros (Figura 2.2). Existem vários métodos para conseguir extrair o conhecimento do especialista e cabe ao engenheiro encontrar a melhor técnica ou a combinação delas para executar a tarefa (KENDAL; CREEN, 2007; MILTON, 2007).

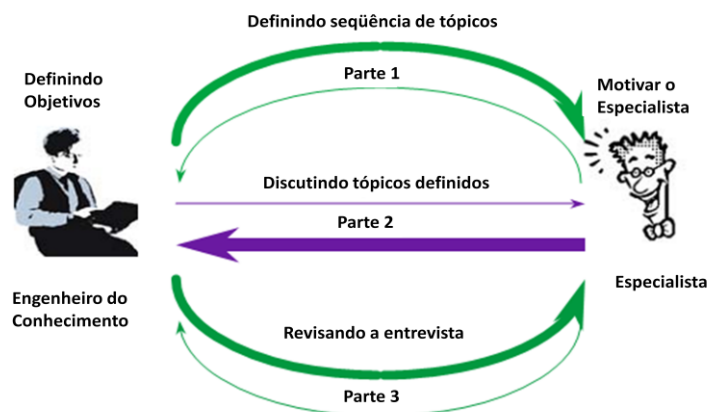


Figura 2.2 Processo de entrevista (adaptado de Kendal e Creen, 2007).

Segundo Milton (2007), para cada tipo de conhecimento existe uma técnica mais adequada para a aquisição, sendo a escolha da técnica relacionada ao tipo do conhecimento, se é tácito ou explícito; se é procedural ou conceitual.

2.1.3 Representação do Conhecimento

Brachman e Levesque (2004) afirmam que a representação do conhecimento é uma relação entre dois domínios, cujo propósito do primeiro é representar o segundo.

Já de acordo com Chorafas (1990), a representação do conhecimento é a tarefa de moldar em um computador o conhecimento coletado. Essa tarefa se torna complexa devido ao fato de não existir uma forma muito bem definida de se representar o conhecimento para

um tipo específico de problema. Cabe ao engenheiro detectar a melhor forma de representação, pois cada uma delas possui pontos positivos e negativos.

Kendal e Creen (2007) citam que existem dois tipos de conhecimentos a serem representados, o conhecimento profundo e o conhecimento superficial. Ambos diferenciam-se pelo modo de ser representado e as tarefas que vão executar.

O conhecimento superficial é mais dependente de uma tarefa, mais frágil por não ser possível extrapolá-lo para outros problemas, porém é bastante efetivo para um tipo específico de tarefa. Já o conhecimento profundo, por um lado é mais complicado de se representar, por outro, se torna menos dependente de uma tarefa específica, descreve melhor as relações do sistema e possui um nível abstrato mais bem definido.

Devido ao fato de não ser muito bem definido o modo de representar o conhecimento para um tipo específico, o engenheiro do conhecimento deve conhecer vários métodos para analisá-los de acordo com o domínio e determinar qual a representação que possui mais vantagens para resolver o problema.

A forma de modelagem do conhecimento é que vai determinar a eficiência e a eficácia do sistema para resolver problemas do mundo real (CHORAFAS, 1990).

2.1.3.1 Técnicas de Representação

Dentre as formas de representação podemos citar técnicas como, *frames*, redes semânticas, lógica proposicional, árvore de decisão, algoritmo genético, lógica *fuzzy*, dentre outras.

Cada uma dessas técnicas possui suas particularidades podendo apresentar um melhor ou pior desempenho para um determinado tipo de problema. A determinação de qual é a técnica mais apropriada depende do conhecimento do engenheiro e muitas vezes são determinadas por meio de testes empíricos.

2.1.3.1.1 Redes Semânticas

As redes semânticas de acordo com Kasabov (1996) utilizam grafos direcionados para representar a informação contextual. Já Chorafas (1990), afirma que as redes semânticas ilustram as relações por meio de diagramas constituídos de nós e arcos, sendo que os nós representam objetos, ações ou eventos e os arcos a relação entre os nós.

Kendal e Creen (2007) citam as redes semânticas como sendo gráficos poderosos e flexíveis para representar o conhecimento. Ainda afirmam que é usada

comumente como uma ferramenta de comunicação entre o engenheiro do conhecimento e o especialista da área no processo de aquisição do conhecimento.

A vantagem da rede semântica é a facilidade de representar um conhecimento hierárquico, sendo relativamente simples o entendimento dos gráficos. Outro ponto positivo nesse tipo de representação é o fato de poder representar diferentes objetos e as relações entre eles.

Por outro lado, fica difícil estabelecer todas as possíveis inferências em uma determinada rede, os diagramas podem se tornar complexos, a relação entre os nós pode se tornar um problema combinatório dependendo da quantidade de arcos entre os nós.

Para facilitar o entendimento de uma rede semântica, é dado um exemplo de uma aplicação (Figura 2.3). Pretende-se representar o conhecimento do mundo real, com as relações de: tem_dono, parte_de, instância_de, tem_cor, tem_km, estacionado, tem_idade, é_um.

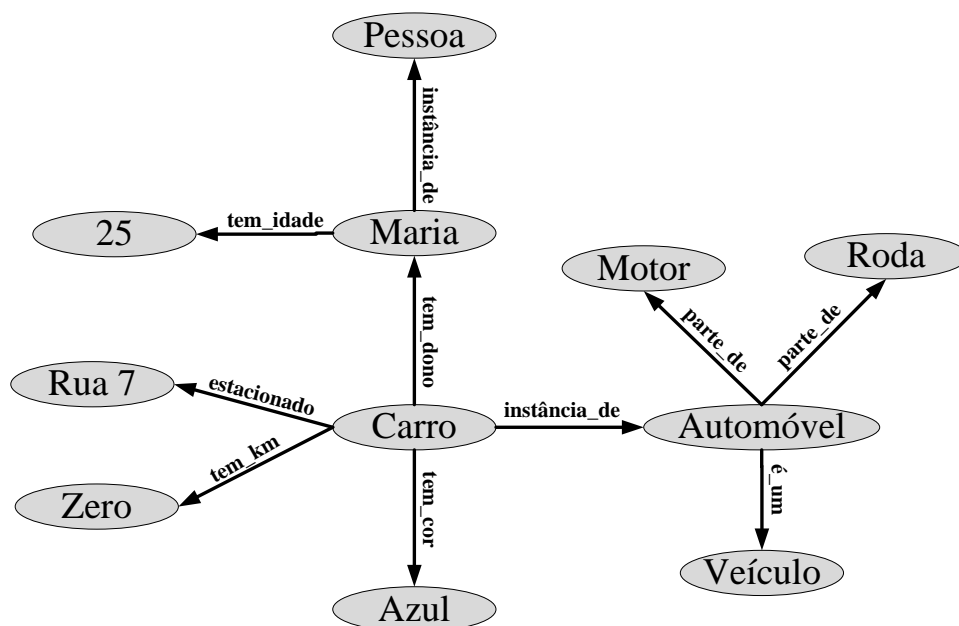


Figura 2.3 Representação de um conhecimento por redes semânticas.

O domínio representado é de uma pessoa com suas características e pertences. Nesse caso, por meio das relações, representou-se Maria como sendo uma pessoa de 25 anos de idade que possui um carro zero, azul estacionado na Rua 7.

O domínio do carro é representando em conjunto, demonstrando que esse carro é uma instância de um automóvel que possui rodas, motor e faz parte de uma categoria maior chamada veículo.

2.1.3.1.2 Frames

Frames, de acordo com Kasabov (1996), são estruturas nas quais se representam informações estruturadas para situações padrões. Já Kendal e Creen (2007), afirmam que os *frames* são uma versão simplificada das redes semânticas, por conter apenas uma relação do tipo "é um/uma".

Os *frames* provêm métodos de armazenamento do conhecimento por meio de informações específicas de um objeto, sendo elas dados ou rotinas.

Esses *frames* são representados por tabelas na qual a primeira linha contém o nome do objeto, também chamado de identificador. As seguintes linhas, também chamadas de *slots*, contêm as características desse objeto. Esses *slots* também podem possuir links para outros *frames*.

De acordo com Kendal e Creen (2007), podem existir três tipos de *slots*: um com nome que contém um dado primitivo relacionado a ele, outro que mostra uma relação "é um/uma", e o último contendo um código procedural.

Os *frames* possuem as vantagens de serem representados em tabelas e com isso facilitar a leitura e assimilação do conteúdo, permitem representar heranças diminuindo a complexidade da representação, combinam conhecimento procedural e declarativo em uma única representação.

Mas também possuem desvantagens como, por exemplo, não possuem informações semânticas e para alguns domínios podem se tornar um problema. (KENDAL; CREEN, 2007).

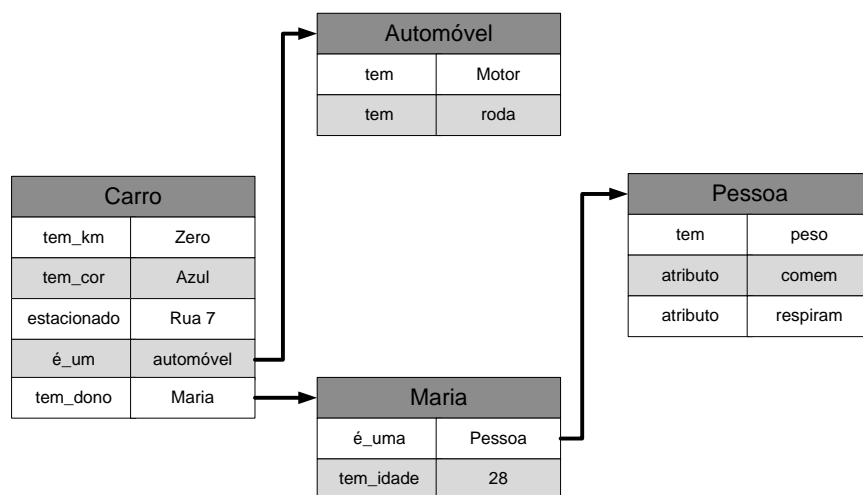


Figura 2.4 Exemplo de Frames.

Na Figura 2.4 é representado o mesmo domínio apresentado no exemplo das redes semânticas (Figura 2.3). É apresentada uma classe pessoa com um exemplo instanciado

Maria possuindo características próprias e herdadas da classe superior. Ainda apresenta o domínio do carro análogo ao domínio da pessoa.

2.1.3.1.3 Árvores de decisão

A árvore de decisão é uma técnica de representação de conhecimento. A facilidade para o entendimento, implementação e uso a torna uma técnica muito utilizada.

As árvores de decisão, segundo Kasabov (1996), são um caso especial de grafos orientados. Esses grafos possuem nós e arcos orientados, sendo que todo arco leva a um nó, e os nós que não possuem arcos são chamados de folhas. Já as árvores possuem um elemento a mais, chamado de raiz, na qual não chega nenhum arco.

A raiz é o nó inicial de uma árvore, toda a tomada de decisão é iniciada avaliando primeiramente a raiz, e termina quando se atinge uma folha, sendo essa a decisão tomada. A decisão tomada pode ser explicada por meio do caminho escolhido, esse caminho é a junção dos nós visitados desde a raiz, até a folha (GIARRATANO; RILEY, 1998).

Existem vários tipos de árvores, cada uma delas com suas vantagens e desvantagens. Dentre as mais comuns estão: Árvores binárias, ID3, Árvores B, Árvore AVL. Um ponto muito positivo nessas árvores é a facilidade para transformá-las em regras.

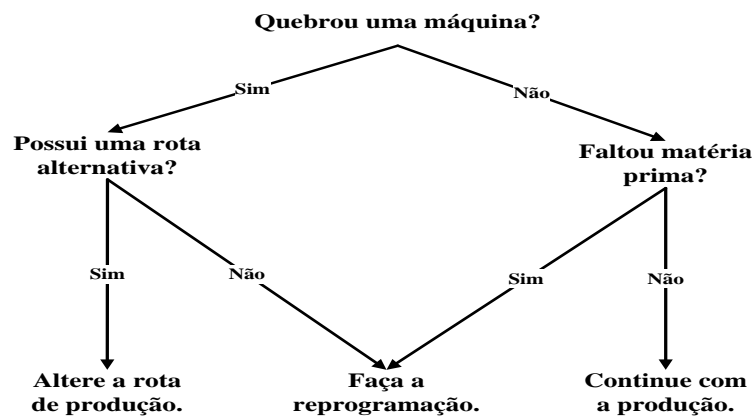


Figura 2.5 Exemplo de uma árvore de decisão binária.

Na Figura 2.5 é apresentado um exemplo de árvore binária representando possibilidades de escolha de uma atividade dentro de um domínio específico.

2.1.3.1.4 Lógica Proposicional

A lógica proposicional é formada de proposições, sendo que cada proposição pode ser verdadeira ou falsa (princípio do terceiro excluído). Existe também outra lei que garante que uma proposição assumirá apenas um valor, que é conhecida como a lei da não contradição.

Essas proposições podem ser classificadas em proposições simples (atômicas) e compostas. As simples são aquelas que só contêm uma única proposição, já as compostas podem conter duas ou mais, sendo ligadas por conectivos.

O conectivo é o elemento que une as proposições. Existem vários tipos de conectivos, por exemplo: não (\neg), e (\wedge), ou (\vee), se-então (\rightarrow), se e somente se (\leftrightarrow). Essa forma de representação não leva apenas a sintaxe em consideração, mas também a semântica da proposição, podendo atribuir os valores de: verdadeiro (v) ou falso (f) (BRACHMAN; LEVESQUE, 2004).

De acordo com Giarratano e Riley (1998), a lógica proposicional é simplesmente uma lógica formal simbólica para a manipulação de proposições.

Uma proposição pode ter todos os seus elementos com o valor verdade "v", caso isso ocorra, essa sentença passa a ser uma tautologia. Caso contrário, se todos os elementos tiverem o valor verdade "f", já passa a ser uma contradição. Caso ocorra da sentença não ser nem uma tautologia e nem uma contradição, ela é chamada de contingência.

Por meio das proposições compostas, ou sentenças, é possível realizar inferências usando regras explicitadas pela lógica proposicional.

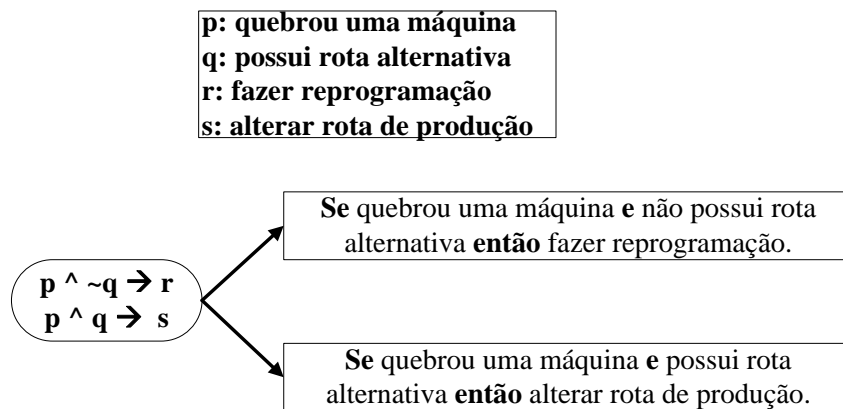


Figura 2.6 Exemplo de lógica proposicional.

Para melhor entendimento de como é uma representação do conhecimento utilizando a lógica proposicional, é dado um exemplo da representação de regras de uma indústria, possibilitando a inferência para a tomada de decisões (Figura 2.6).

2.1.3.1.5 Algoritmo Genético

Os algoritmos genéticos (AG's) são metas-heurísticas embasadas na teoria da evolução das espécies de Darwin. Esse algoritmo privilegia a população mais apta, fazendo com que ela se prolifere, diminuindo a quantidade de indivíduos menos aptos (REZENDE, 2003).

Os AG's operam por meio de evoluções nas quais atualiza a população. Essa população pode ser caracterizada como possíveis soluções para o problema (MITCHELL, 1997).

Para a escolha da nova população, o AG pode aplicar operações de mutação e cruzamento e após essas operações é realizado um cálculo de aptidão para determinar quais os indivíduos estão mais aptos a prosseguirem para a nova população.

Cada indivíduo é um cromossomo do sistema. Esse cromossomo deve ser modelado para o tipo de conhecimento que se deseja representar. Após essa modelagem, deve-se determinar como serão feitas as operações de mutação e cruzamento.

Essas operações são realizadas para garantir que novos indivíduos sejam criados, mas sem perder as características dos indivíduos anteriores.

A operação de cruzamento determina como será o novo indivíduo, também chamado de filho, a partir da combinação de dois indivíduos já existentes, chamados de pais. Para essa combinação geralmente se determina um ponto de corte aleatoriamente no cromossomo e troca-se uma das partes entre os cromossomos.

Na operação de mutação é determinada aleatoriamente uma posição e é trocado o valor nessa posição escolhendo outro aleatoriamente. Essa operação ocorre em apenas um cromossomo, diferentemente do cruzamento que é uma operação entre dois cromossomos.

Após as operações genéticas, é utilizada uma função para avaliar os indivíduos, chamada de função de aptidão ou *fitness*. Ela utiliza algum critério já previamente estabelecido para transformar em números essa aptidão de cada cromossomo.

Depois de ter o *fitness* de cada cromossomo, são selecionados alguns cromossomos para a próxima população. Essa seleção pode ser feita de várias maneiras, sendo uma delas o método da roleta.

O método da roleta estabelece uma fatia da roleta para cada cromossomo, que é proporcional a sua aptidão. Quanto melhor for avaliado o cromossomo, maior será a sua fatia e conseqüentemente maior a chance de ser escolhido para a próxima população.

É comum utilizar um método de parada para esse algoritmo, pois ele sempre busca a melhor solução por meio de "saltos" em regiões de solução sendo que nem sempre é viável computacionalmente essa busca.

Outro problema caso não seja usado um critério de parada, é que no decorrer das iterações a população pode se tornar muito homogênea, isso significa que o AG pode ter caído em um mínimo local e não necessariamente essa é uma resposta aceitável.

Alguns possíveis critérios de parada são: número de evoluções, variação da média dos melhores indivíduos das últimas gerações e proximidade do objetivo. Esse último só é válido para o caso que se conheça o objetivo.

A seguir é apresentado um exemplo da utilização do AG para a representação do conhecimento na programação da produção. Essa representação determina um cromossomo (Figura 2.7) como sendo a programação, e por meio das operações genéticas se obtém uma boa reprogramação caso ocorra um evento não programado, como por exemplo, uma quebra de máquina (DERIZ, 2007).

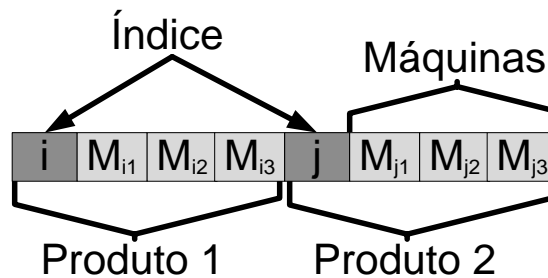


Figura 2.7 Exemplo da codificação de um cromossomo para a representação do conhecimento. (adaptado de Deriz, 2007).

A Figura 2.7 representa um cromossomo contendo informações da programação da produção. Nessa programação estão presentes dois produtos sendo especificado o produto e seu roteiro de fabricação.

2.1.3.1.6 Sistemas Fuzzy

Os sistemas *fuzzy* são sistemas computacionais que agregam a teoria de conjuntos *fuzzy*, lógica *fuzzy*, entre outras. (KLIR; YUAN, 1995).

Os conjuntos *fuzzy* se diferem dos conjuntos tradicionais apresentados pela teoria dos conjuntos pelo fato de que na teoria dos conjuntos, um elemento pertence ou não a um conjunto. Já nos conjuntos *fuzzy*, um elemento pertence a um conjunto com certa porcentagem, podendo pertencer a mais de um conjunto.

Essa porcentagem é conhecida como grau de pertinência, e seus valores são sempre no intervalo $[0,1]$ (KASABOV, 1996). Os conjuntos *fuzzy* são modelados por meio dos valores existentes da variável e os devidos graus de pertinência.

Existem vários tipos de conjuntos, sendo alguns deles, os triangulares e os trapezoidais. Não existe um estudo muito específico para determinar qual é a melhor maneira de representar o conjunto *fuzzy*, essa escolha pode ser realizada por meio do empirismo.

A lógica *fuzzy* trata informações incertas usualmente empregadas na comunicação humana. Por essa característica, a lógica *fuzzy* fica bem próxima da nossa

maneira de pensar. É possível representar os valores *fuzzy* por meio de variáveis lingüísticas. Essas variáveis que são palavras utilizadas no dia-a-dia.

Um exemplo para demonstrar esse tratamento de dados incertos por meio de variáveis lingüísticas é a altura. Dada uma pessoa com uma altura de um metro e setenta, queremos definir se ela é alta, baixa ou de estatura média.

Se essa pessoa estiver perto de outra de um metro e cinquenta pode ser considerada uma pessoa alta, mas se estiver perto de uma pessoa de um metro e noventa já pode ser considerada “baixa”. Para expressar a altura de um e setenta, podemos falar que é mais ou menos alta.

Essa característica, altura, pode ser expressa por um conjunto *fuzzy* (Figura 2.8) com as variáveis lingüísticas alta, média e baixa. E ainda é possível representar o raciocínio que é comumente utilizado: muito alta, mais ou menos alta.

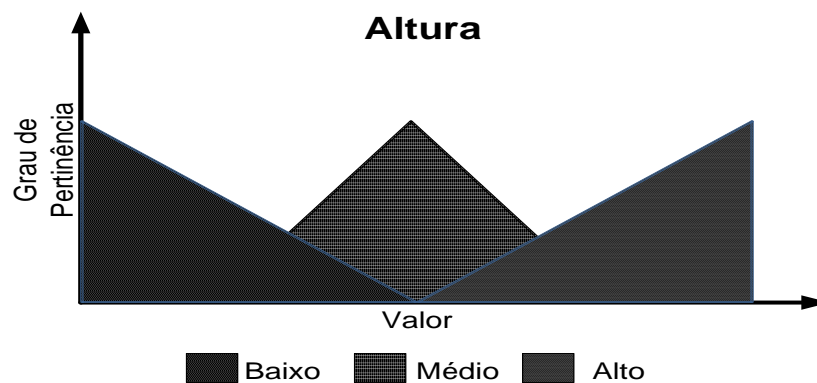


Figura 2.8 Exemplo de conjuntos *Fuzzy*.

Após a modelagem do conhecimento em conjuntos *fuzzy*, é possível fazer inferências por meio das operações *fuzzy*. Utilizam-se as entradas qualitativas como, por exemplo, “altura é média” e aplica-se um valor *fuzzy* para essa entrada. É por meio desse valor que as inferências são utilizadas (KENDAL; CREEN, 2007).

Para a inferência *fuzzy* é comum a utilização de regras *fuzzy*. Essas regras são da forma: se - então. A partir da união dessas regras é possível obter uma resposta, porém ainda do tipo *fuzzy*. Existem técnicas para transformar essa resposta em um valor escalar, sendo que as mais comuns são: centro de massas, média dos máximos, entre outras.

2.1.3.1.7 UML 2

A UML (*Unified Modeling Language* - Linguagem de Modelagem Unificada) é uma linguagem visual muito utilizada para modelar sistemas computacionais utilizando o

paradigma de orientação a objeto (GUEDES, 2005). Porém, pode-se utilizar a UML para modelar qualquer tipo de sistema, não apenas *softwares*.

De acordo com Booch *et al.* (2005), a UML é uma linguagem muito expressiva, direcionando todas as visões necessárias para a criação de um sistema.

Com a UML pode-se definir as características do sistema, seus requisitos, o comportamento, a estrutura lógica, a dinâmica de processos, e as necessidades físicas dos equipamentos presentes no sistema (GUEDES, 2005).

A UML é constituída de diagramas que fornecem diferentes visões do sistema, cada um deles com uma visão específica de uma parte ou de todo o sistema. A seguir são apresentados os diagramas e uma breve descrição de cada um deles.

Diagrama de Casos de Uso: é o diagrama mais geral da UML, é normalmente utilizado na fase de levantamento dos requisitos, porém é consultado a todo o momento do processo de modelagem.

Apresenta uma linguagem simples e é de fácil compreensão para que todos os envolvidos na criação, principalmente os especialistas do conhecimento, entendam o funcionamento do sistema.

Esse diagrama identifica os atores (usuários do sistema, mesmo que ele seja um *hardware*) e os serviços que o sistema irá prestar para os atores.

Diagrama de Classes: esse diagrama define a estrutura das classes do sistema. Neste diagrama é definido como será o relacionamento das classes assim como seus métodos e variáveis.

Diagrama de Objetos: esse diagrama é utilizado associado ao diagrama de classes complementando-o. São apresentados, em um determinado momento de execução, os valores armazenados pelos objetos do diagrama de classes.

Diagrama de Estrutura Composta: esse diagrama é usado para modelar as colaborações que são um conjunto de entidades que cooperam entre si para executar uma função específica. Esse diagrama foi incorporado a partir da UML 2.

Diagrama de Seqüência: esse diagrama apresenta a ordem temporal das trocas de mensagens entre os objetos envolvidos em um determinado processo. Geralmente é baseado em um diagrama de caso de uso.

Diagrama de Comunicação: conhecido até a UML 1.5 como diagrama de colaboração. Está diretamente ligado ao diagrama de seqüência, complementando-o.

Mostra as mesmas comunicações do diagrama de seqüência, porém com enfoque diferente, concentrando na ligação entre os objetos e as trocas de mensagens entre eles durante o processo.

Diagrama de Máquina de Estados: conhecido como gráfico de estados nas versões anteriores da UML, ele acompanha a mudança de estado de uma classe, de um caso de uso ou até mesmo de um subsistema ou um sistema completo.

Diagrama de Atividades: esse diagrama era um caso especial do antigo diagrama de estados, mas tornou-se independente na UML 2. Agora se baseia em redes de Petri e não mais em máquina de estados (GUEDES, 2005).

Esse diagrama se preocupa em descrever os passos para concluir uma atividade específica. Enfoca no fluxo de controle e no fluxo de objeto de uma atividade.

Diagrama de Interação Geral: esse diagrama também surgiu apenas na UML 2. Ele fornece uma visão geral dentro de um sistema, englobando diversos diagramas de interação para demonstrar o processo.

Diagrama de Componentes: esse diagrama está diretamente relacionado com a linguagem de programação escolhida para o desenvolvimento. Determina como serão estruturados os módulos e como eles irão interagir entre si. Esses módulos podem ser bibliotecas, formulários, arquivos, módulos de código fonte, módulos executáveis ou banco de dados.

Diagrama de Implantação: determina as topologias, protocolos de comunicação, necessidades de *hardware*, servidores de estação, ou seja, todos os requisitos físicos para que seja executado o sistema.

Diagrama de Pacotes: determina os submódulos do sistema e como eles estão relacionados.

Diagrama de Tempo: Esse diagrama descreve a mudança de estado de uma instância de classe durante o tempo. Geralmente usado para demonstrar a mudança no estado dos objetos em decorrência de eventos externos.

Esse diagrama foi incorporado a partir da UML 2.

2.1.3.1.8 *Redes de Petri*

Redes de Petri ordinária é uma técnica que se apóia em uma modelagem matemática e ao mesmo tempo apresenta uma representação gráfica.

Essa técnica é tida como promissora na representação de sistemas, tais como: concorrentes, assíncronos, distribuídos, paralelos, não-determinísticos ou estocásticos (MURATA, 1989).

Na representação matemática, é possível definir uma equação de estados, equações algébricas e outros modelos matemáticos que descrevem comportamentos de sistemas. A representação gráfica dá suporte similar aos fluxogramas, diagramas de blocos, podendo ser usada pela sua comunicação visual (MURATA, 1989).

As redes de Petri ordinária são grafos orientados compostos por quatro primitivas: transição, lugar, arco e marcação. Os lugares são representados por círculos, as transições por retângulos ou barras, o arco por uma flecha e a marca por um ponto (Figura 2.9).

Esses elementos da redes de Petri, pela visão matemática, formam uma tupla composta de quatro elementos apresentados na Tabela 2.1.

Tabela 2.1 Descrição formal da redes de Petri (adaptado de Murata 1989)

Tupla	$PN = (P, T, F, W, M_0)$.
P	Conjunto finito de lugares
T	Conjunto finito de transições
F	Conjunto de arcos
M_0	Marca Inicial

As transições e lugares são conectados por meio dos arcos, formando a rede. Uma transição somente tem ligação direta com um lugar e nunca com outra transição, o mesmo acontece com o lugar, só possui ligação com transições e nunca com lugares.

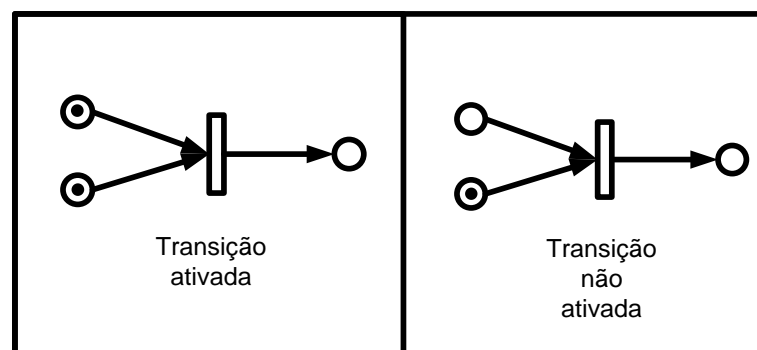


Figura 2.9 Exemplo de condições para a ativação de uma transição.

A presença da marca em um lugar indica que existe uma condição verdadeira associada a esse lugar. Se associado a esse lugar ativo estiver uma transição, ela será ativada se e somente se não existir nenhum outro lugar que não possua marca associada a ela (Figura 2.9).

Existem extensões das redes de Petri ordinária, cada uma delas possui suas vantagens e desvantagens. A escolha da extensão mais adequada pode ser realizada pela análise de alguns aspectos, como por exemplo: o domínio a ser representado, busca por menor complexidade, entre outros.

Dentre algumas das variantes existentes podemos citar: redes de Petri temporizada, trata a variante tempo; redes de Petri colorida, acrescenta diferentes tipos de dados a serem tratados; entre outras.

A modelagem gráfica de redes de Petri pode ser interpretada de várias formas, possibilitando representar diversas aplicações. Murata (1989) apresenta algumas interpretações típicas de transições e lugares (Tabela 2.2).

Tabela 2.2 Algumas interpretações típicas de lugares e transições (adaptado de Murata, 1989)

Lugares de entrada	Transições	Lugares de saída
Pré-condições	Eventos	Pós-condições
Dado de entrada	Passo computacional	Dado de saída
Sinal de entrada	Processamento de sinal	Sinal de saída
Recursos necessários	Tarefa ou Trabalho	Recurso liberado
Condições	Cláusulas na lógica	Conclusões
<i>Buffer</i>	Processamento	<i>Buffer</i>

2.1.3.1.9 Fluxogramas

O fluxograma é uma representação gráfica seqüencial de uma atividade, apresentada de forma detalhada ou resumida, no qual se descreve as operações envolvidas nessa atividade.

Algumas das vantagens da representação por fluxograma são: o fácil entendimento e leitura, a fácil identificação de pontos cruciais na atividade representada, a representação flexível, e o fato de permitir um maior grau de análise.

O fluxograma é composto de elementos gráficos, sendo que cada um contém um significado próprio. São apresentados na Figura 2.10 alguns exemplos desses elementos e seus respectivos significados.

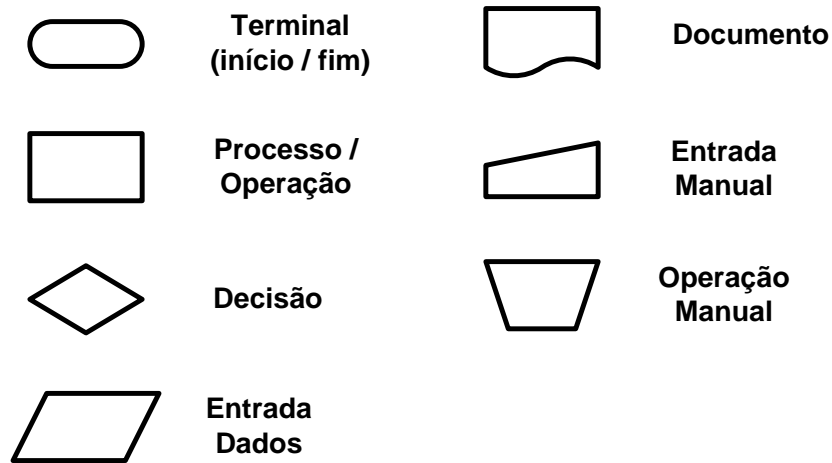


Figura 2.10 Exemplos de elementos do fluxograma com seus significados

Com o apoio desses elementos, conectados por arcos direcionados é possível representar diversas atividades, como por exemplo, a demonstrada na Figura 2.11.

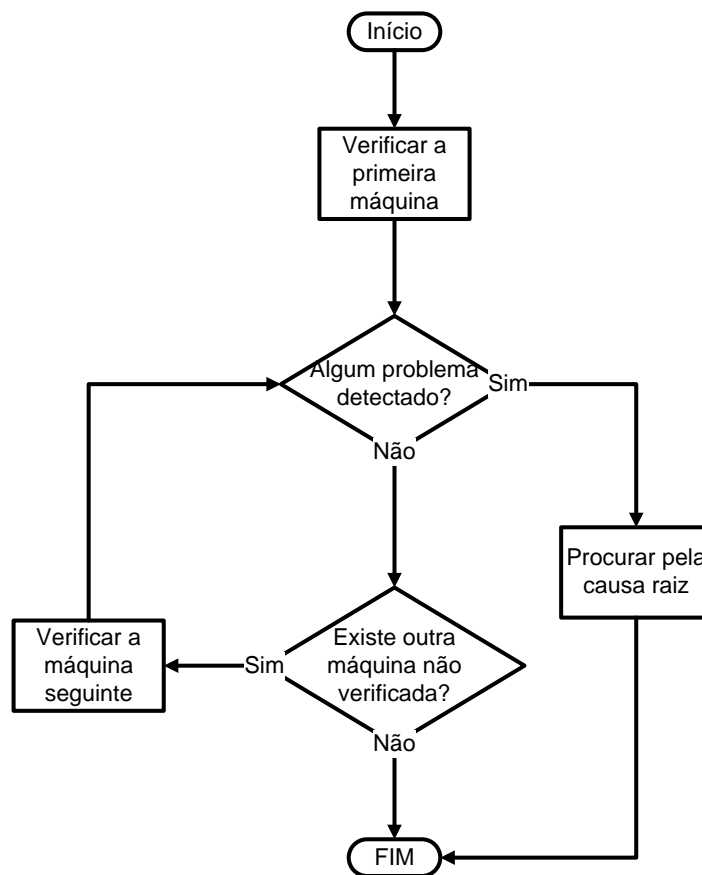


Figura 2.11. Exemplo de uma atividade representada por um fluxograma.

Essa atividade apresentada na Figura 2.11 por meio de um fluxograma representa uma rotina de atividade de verificação de problemas em máquinas. Caso seja detectado algum problema, é necessário tomar alguma medida, nesse caso procurar pela causa raiz.

2.2 Trabalhos de Pesquisa

Diversos trabalhos apontam o uso do conhecimento como uma vantagem competitiva para as organizações (ALAVI; LEIDNER, 2001; STEFANOVITZ; NAGANO, 2006; WOLFF; LEFEBVRE; RENAUD, 2006; GRUNDSPENKIS, 2007; MORENO *et al*, 2007). Uma dessas vantagens está relacionada ao apoio à decisão, aumentando assim a qualidade da produção e conseqüentemente dos produtos finais (O’KANE, 2000).

Serão apresentados nesta seção, alguns trabalhos de pesquisa relacionados à engenharia do conhecimento. Essa seção será dividida em duas subseções principais, sendo elas: pesquisas em diferentes domínios e pesquisas em domínios correlatos.

A subseção de pesquisa em diferentes domínios ainda é dividida em trabalhos que enfocam diferentes características do conhecimento como: reuso, conhecimento individual, conhecimento organizacional e representação do conhecimento.

Essas divisões são apresentadas na Figura 2.12, sendo representadas pela técnica de representação do conhecimento chamada de “mapas do conhecimento”. Segundo Yang (2007), o mapa de conhecimento é uma ótima técnica para apresentar de maneira gráfica o conhecimento explícito e tácito, possibilitando a descoberta de novos conhecimentos por meio da representação.

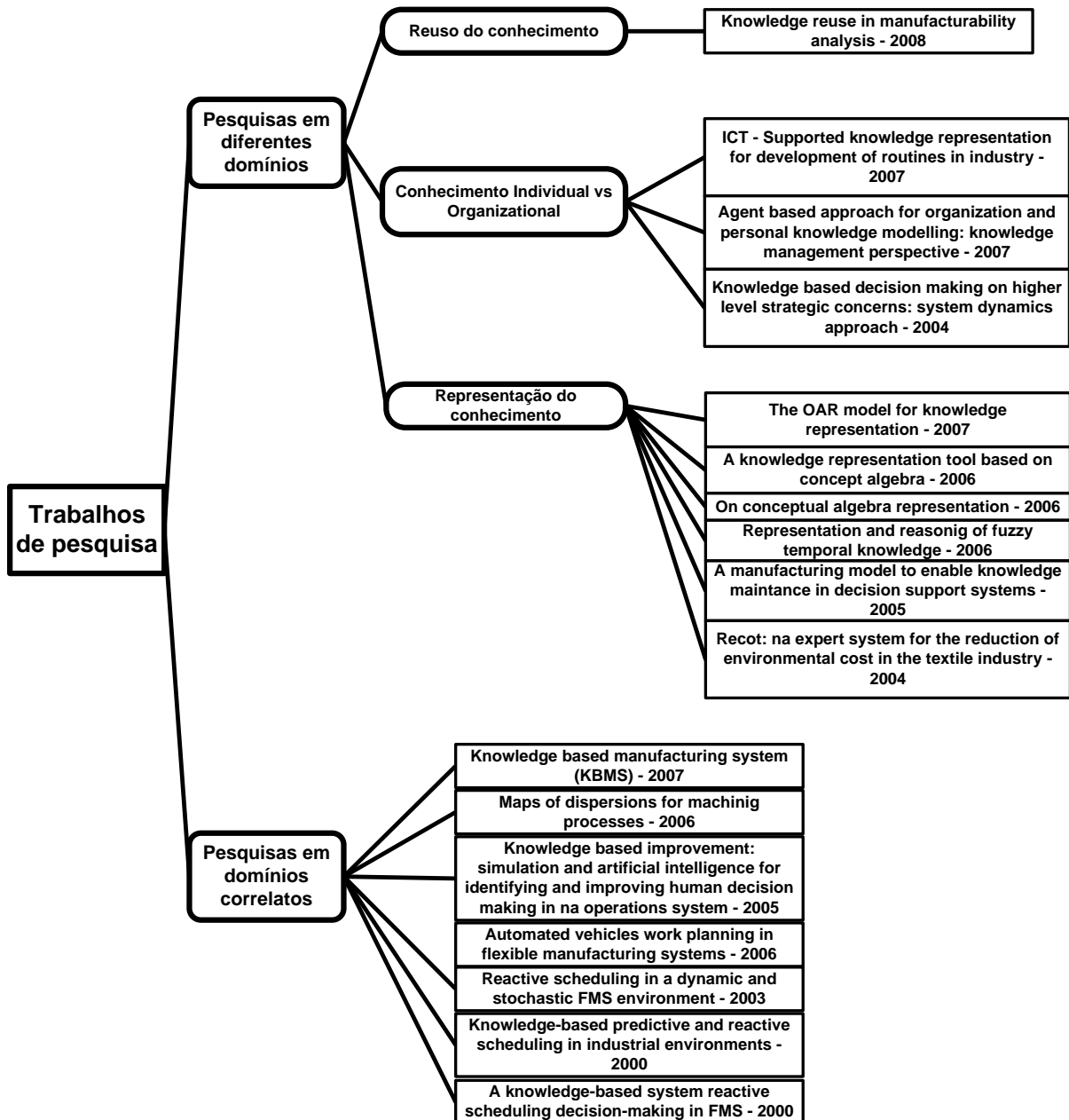


Figura 2.12 Mapa do conhecimento representando a estrutura de apresentação dos trabalhos de pesquisa.

2.2.1 Pesquisa em Diferentes Domínios

Na literatura são apresentadas algumas preocupações referentes ao conhecimento, focando algumas de suas características, classificações, representações, entre outras.

Um ponto que os autores Cochrane *et al* (2008) levam em consideração e afirmam ser de extrema importância é a preocupação com o reuso do conhecimento. Eles apresentam diretrizes de modelagem para melhorar o reuso do conhecimento no desenvolvimento de sistemas de apoio à decisão no âmbito da manufatura.

São estabelecidos três princípios a serem seguidos para garantir o reuso do conhecimento. O primeiro deles é separar conhecimento de informação, o segundo é classificar e separar o conhecimento em hierarquias distintas, uma sendo relacionada ao conhecimento sobre o produto e outra sobre o processo de manufatura, e o terceiro é criar e aplicar diferentes níveis de estratégias de manufatura.

É apresentada uma plataforma experimental chamada MAP (*Manufacturability Analysis Platform*) para testar e avaliar os princípios propostos. A plataforma apresenta uma arquitetura em camadas para separar informação de conhecimento, sendo a informação armazenada em bancos de dados e o conhecimento expresso pela modelagem orientada a objeto.

Foi utilizada na manufatura de câmara de combustão de um motor a jato. O modelo criado foi representado por meio de orientação a objeto seguindo alguns padrões da UML. O reuso foi aplicado na localização de regras e restrições entre as camadas da estratégia de manufatura e na representação do produto.

Alguns autores também vêm dando destaque ao conhecimento individual e ao organizacional, destacando a importância de cada um deles e da necessidade de obter os dois para realizar uma modelagem mais pertinente.

Alegando que a aprendizagem organizacional é importante para a produtividade da empresa, os autores Hjelmervik e Wang (2007) demonstram como um sistema de gestão do conhecimento bem desenvolvido e bem implantado pode apoiar a realização das rotinas de operação nas indústrias de manufatura, ao contrário da maioria das tecnologias de informação e comunicação, que não oferece suporte para o aprendizado organizacional.

Foram realizadas entrevistas de maneira exaustiva com os empregados relacionados à operação para estabelecer de que forma um sistema de gestão do conhecimento agregaria valor ao aprendizado organizacional. Foi dada mais liberdade às equipes de cada setor e implantado um sistema de melhores práticas, no qual ficam relatadas as práticas executadas que geraram bons resultados às equipes.

Foi desenvolvido um sistema de gestão do conhecimento para compartilhar entre as diferentes equipes de diferentes plantas essas melhores práticas, transformando o conhecimento individual em conhecimento organizacional e possibilitando o refinamento desse conhecimento.

O autor Grundspenkis (2007) afirma existir uma lacuna entre a engenharia de conhecimento e a inteligência artificial. Para diminuir essa lacuna, ele propõe uma solução

baseada em agentes inteligentes e que suportam o conhecimento organizacional e o individual.

São descritos dois modelos. O primeiro é a soma da modelagem do trabalho em questão, com a modelagem do conhecimento para o desenvolvimento de uma memória inteligente da empresa. O segundo é a descrição de comunidades de agentes e objetos, e conseqüentemente a descrição de um modelo conceitual do conhecimento individual baseado em agentes.

Definiu-se um processo de aquisição, formalização, representação e aplicação do conhecimento, tendo como modelo central do esquema o módulo de memória inteligente da empresa. Esse modelo está disposto em 7 camadas, sendo elas: base de conhecimento, aquisição do conhecimento, formalização do conhecimento, representação do conhecimento, processamento do conhecimento, aplicação do conhecimento e uso do conhecimento.

O modelo conceitual baseado em agentes foi descrito por três diferentes tipos de agentes, sendo um responsável pela comunicação com o motor de inferência, outro entre os agentes e um terceiro para buscar, filtrar e gerar o fluxo de trabalho.

Yim *et al* (2004), foca sua pesquisa no desenvolvimento de um método de tomada de decisão baseado em conhecimento no domínio de gestão de negócios. Os problemas presentes nesse domínio, segundo o autor, são caracterizados por serem complexos, dinâmicos, influenciados por fatores de conhecimento tácito, desestruturados e com repostas em longo prazo.

Esse método deve ser capaz de estruturar os conhecimentos individuais de forma que fiquem integrados, permitindo assim realizar o auxílio à tomada de decisão. A metodologia utilizada foi a dinâmica de sistemas, pois ela é capaz de representar esse cenário de maneira realística e refletir corretamente o domínio, além de representar tanto o conhecimento explícito quanto o implícito.

O foco não está na tomada de decisão em si, mas sim no conhecimento atrelado a ela. Para determinar esse conhecimento, o processo foi dividido em cinco fases: definição do problema, conceitualização do problema, formulação do modelo do conhecimento, teste e apoio à decisão, e aplicação.

A metodologia foi validada aplicando em uma empresa de telefonia para auxiliar à tomada de decisão e recuperar a fatia perdida do mercado. Para a aquisição do conhecimento foram realizadas entrevistas estruturadas com os gerentes e supervisores, para a conceitualização e formalização, foi usado diagrama de *loop* casual, apresentando os elementos, a interação e a influência entre eles.

A validação foi realizada parcialmente comparando-se os dados históricos de novos consumidores dos últimos 7 meses, e simulando-os por aproximadamente 4 anos. A outra parte da validação da metodologia foi realizada comparando-a com outras abordagens aplicadas em diversos ambientes de tomada de decisão.

Outro ponto importante na área de engenharia do conhecimento são as técnicas de representação do conhecimento. Na literatura são apresentadas várias técnicas para representar o conhecimento, tanto o implícito quanto o explícito.

Segundo Wang (2006), é possível representar e manipular o conhecimento por meio de uma teoria formal própria para manipulação de conceitos abstratos, conhecida por álgebra conceitual.

A álgebra conceitual se baseia na relação entre atributos e objetos, definindo teoremas e operações para sua manipulação. Por meio de objetos, atributos, e suas relações, é possível representar tanto o conhecimento tácito quanto o explícito e manipulá-los.

Wang (2007) formaliza o modelo OAR (*Object-Attribute-Relation*), para descrever os mecanismos da memória de longo prazo, que é a memória responsável por armazenar as experiências, habilidades, fatos e conhecimento pertencentes a um humano.

Tian e Wang (2007) também se baseiam na álgebra conceitual e no modelo OAR, para apresentar uma ferramenta de visualização para facilitar a representação do conhecimento.

Essa ferramenta de visualização se baseia na modelagem de objetos, cada um deles possui atributos que descrevem características do objeto. É possível estabelecer relações entre objetos, entre atributos, e entre objetos e atributos.

Após essa definição do modelo OAR, pela álgebra conceitual, é possível manipular os objetos por meio de 9 operações, sendo elas: herança, extensão, *tailoring*, substituição, composição, decomposição, generalização / agregação, especificação e instanciação.

Essa ferramenta de visualização, segundo os autores, se mostra uma boa ferramenta para a representação do conhecimento.

Manaf e Beikzadeh (2006) propõem um modelo para a representação e raciocínio do conhecimento temporal *fuzzy*. Esse modelo representa o conhecimento de maneira quantitativa e qualitativa, por meio de funções de pertinência *fuzzy* baseadas em intervalos.

Eles partiram do pressuposto de que o conhecimento, levando em conta a questão temporal, é cheio de incertezas e imprecisões. Para lidar com esse tipo de problema,

foi criado um modelo para processar o conhecimento temporal *fuzzy*, baseado na teoria dos conjuntos *fuzzy* e na álgebra de intervalos de Allen.

A álgebra de intervalos de Allen permite criar intervalos e estabelecer algumas relações entre eles, somando um total de 13 possíveis relações. Porém essa representação não trata a questão da incerteza, por isso os autores propuseram inserir a teoria da lógica *fuzzy*.

Com a inserção da lógica *fuzzy*, é possível atribuir variáveis linguísticas a esses intervalos e realizar inferências oferecidas pela lógica, agregando assim uma maior representatividade.

Para a aplicação desse modelo foi apresentado um algoritmo. Primeiramente é necessário transformar o cenário do problema em uma representação das relações temporais, seguindo a álgebra temporal de Allen. Cada nó representa o intervalo para uma ação existente e os arcos entre os nós representam a relação temporal entre eles.

Depois de definir as funções de pertinência *fuzzy* para toda incerteza e imprecisão existente no cenário, é necessário estabelecer a relação entre as ações e as funções de pertinência. Uma vez estabelecidas as relações, deve-se realizar inferências e checar consistências. Também é possível realizar a poda da rede para retirar as inconsistências.

Essa representação, segundo os autores, tem a vantagem de ser uma modelagem simples e de prática programação. Outro ponto vantajoso é que pode representar os conhecimentos tácitos e explícitos.

Guerra e Young (2005) definem um modelo de informação e conhecimento de instalações de manufatura. Esse modelo permite o armazenamento, acesso e gerenciamento tanto da informação quanto do conhecimento relativos ao domínio proposto.

O objetivo é dar subsídios às decisões de planejamento por meio da análise, tanto do conhecimento do processo quanto dos produtos. Também é almejada a captura de novos conhecimentos por meio do ciclo de vida de manutenção do conhecimento e do uso de diferentes tipos de representação.

Para a manutenção do conhecimento, são apresentadas três preocupações: a habilidade para lidar com vários tipos de conhecimentos, a descoberta de valiosos e novos conhecimentos, e a elaboração de um método que permita realizar a manutenção do conhecimento. Para a modelagem os autores utilizaram os métodos da orientação a objeto e desenvolveram um sistema de apoio à decisão baseado em conhecimento. Eles dividiram, a princípio, em duas modelagens macro: modelagem do produto e modelo de manufatura.

A modelagem de manufatura engloba os elementos de processo, recursos e estratégias de manufatura. A estrutura apresentada permite o armazenamento das informações

e conhecimentos sobre a usinagem dentro de um modelo de conhecimento e informação de instalações de manufatura (MFIKM - *Manufacturing Facility Information and Knowledge Model*).

O MFIKM foi modelado por meio de diagramas de classes, e nesse caso, a manutenção se dá pela alteração ou inclusão de algum atributo no diagrama. A partir desse diagrama foi construído um *software* para auxiliar essa manutenção e realizar a validação.

Metaxiotis (2004) apresenta um sistema especialista baseado em conhecimento para auxiliar as pequenas e médias indústrias têxteis da Europa visando um processo com menos poluição e conseqüentemente com menos gastos. O processo escolhido para atuar foi o de coloração do fio por entender que existem vários fatores que impactam em seu resultado final.

Foram propostas quatro etapas para a criação do sistema especialista, a primeira delas é a descrição formal dos fatores chaves que afetam o processo de coloração. A segunda foi descrever modelos para a representação da informação relevante. A terceira foi definir modelos para a representação do conhecimento. A quarta e última etapa foi realizar a integração dos modelos em um sistema de informação unificada que apóie as tomadas de decisões.

A aquisição do conhecimento e da informação foi realizada por meio de entrevistas estruturadas com os operários e engenheiros da fábrica. Foram identificadas e implantadas as melhores práticas visando uma produção limpa, ou seja, diminuindo a poluição e o impacto ambiental.

O conhecimento foi modelado por meio de regras “se-então”, caracterizando as regras de produção. O sistema especialista auxilia a aplicação das melhores práticas no processo. Para a validação, a proposta foi implantada em uma empresa grega do setor têxtil.

2.2.2 Pesquisas em Domínios Correlatos

Existem alguns trabalhos na literatura que dão o enfoque da engenharia do conhecimento próximo ao domínio da programação reativa da produção. Apresentam uma preocupação com a programação no controle da produção.

Halei e Wang (2007) apresentam um novo método no qual as tarefas de engenharia não são usadas para tomadas de decisões e sim para criar roteiros baseadas em conhecimento. Esses roteiros permitem uma maior flexibilidade e dinamismo ao processo de manufatura e simplifica o processo de tomada de decisão.

Os autores alertam para a importância de capturar todos os dados existentes na empresa, para não correr o risco de perder algum conhecimento.

Por meio dos roteiros é criada uma rede de árvores de produção de produtos para modelar o conhecimento sobre os produtos.

Foi desenvolvido um sistema de manufatura baseado em conhecimento com o intuito de aumentar a produtividade e diminuir custos com o processo. Esse sistema é capaz de auxiliar a tomada de decisão caso algum roteiro de produção esteja bloqueado, ele percorre os roteiros buscando alternativas que melhor se adequem àquele determinado momento e visando produzir no tempo devido.

É apresentado um modelo de programação dinâmica, que apresenta possíveis soluções no momento em que forem necessárias, tornando menor o tempo de produção.

Wolff, Lefebvre e Renaud (2006) sugerem um modelo de dispersões no processo de torneamento, não apenas levando em consideração as especificações geométricas da posição ou orientação, mas também a experiência dos atores no processo.

A representação escolhida para a modelagem do conhecimento foi a técnica de mapas de conhecimento, por incluir tanto o conhecimento explícito quanto o tácito, ambos muito importantes para se obter um modelo. Segundo os autores, esses dois tipos de conhecimento requerem métodos adicionais e efetivos, para isso apresentaram uma abordagem genérica para a capitalização do conhecimento.

Essa capitalização do conhecimento foi dividida em três fases: localizar e extrair o conhecimento, modelá-lo, e utilizá-lo. A primeira fase consiste em localizar e extrair o conhecimento tácito sobre o produto e sobre o processo de acordo com o ponto de vista dos tomadores de decisões.

A segunda fase consiste em modelar o conhecimento, sempre visando o reuso, e depois validá-lo com os especialistas. A terceira fase consiste em usar o conhecimento já modelado, sempre pensando em uma estrutura para reutilizá-lo.

Para a modelagem do conhecimento explícito e do tácito por meio dos mapas de conhecimento, os autores apresentam três etapas. A primeira delas foca em determinar a equação de regressão obtida por um exemplo de um projeto de experimentos. A segunda em pesquisar a aplicação nas áreas dos especialistas do conhecimento. E a terceira visa produzir regras de produção levando em consideração todas as áreas apresentadas na etapa anterior.

Aplicando essa abordagem genérica da capitalização do conhecimento, os autores obtiveram um resultado significativo na otimização dos processos.

Os autores Muszyński, Banaszak e Tomczuk-Piróg (2006), apresentam um problema relacionado à determinação de rotas dos veículos auto guiados (AGV – *Automated Guided Vehicle*), livre de eventuais problemas em um cenário com recursos compartilhados.

A abordagem proposta é baseada na representação do conhecimento do tipo de regras para as funções e especificações da estrutura do subsistema de transporte. Para essa representação os autores usaram redes de Petri e método da lógica algébrica.

Foi construído um modelo do problema de satisfação das restrições (CSP – *Constraints Satisfaction Problem*) para representar o sistema de concorrência de processo cíclico (SCCP - *System of Concurrent Cyclic Process*) baseado em conhecimento, pelo fato do CSP estar mais próximo do domínio do problema representado.

O objetivo dessa proposta é representar o sistema por meio do CSP de forma a evitar os conflitos.

Robinson *et al* (2005) apresenta uma metodologia baseada na simulação interativa visual (VIS – *Visual Interactive Simulation*) e na inteligência artificial (IA), que visa identificar e aperfeiçoar a tomada de decisão humana nos sistemas de operação.

A metodologia, conhecida por aperfeiçoamento baseado em conhecimento (KBI – *Knowledge-Based Improvement*), extrai o conhecimento do tomador de decisão por meio do VIS e aplica métodos da IA para representar as tomadas de decisões. Relacionando o VIS com a IA é possível prever o desempenho das operações sob a ótica de diferentes estratégias de decisões tomadas.

O processo da metodologia KBI consiste em cinco estágios: entendimento do processo de tomada de decisão; aquisição dos dados; definição das estratégias das tomadas de decisões pelos usuários; definição das conseqüências das estratégias escolhidas; busca por melhorias.

Para o primeiro estágio, é necessário identificar as variáveis e opções de decisões, os atributos das decisões e o nível desses atributos. As tomadas de decisões são representadas por dois vetores, sendo o primeiro correspondente à variável de decisão e o segundo aos atributos das decisões.

Para o segundo estágio, “aquisição dos dados”, foi usado o VIS para facilitar essa aquisição e armazenamento nos vetores criados anteriormente. No terceiro estágio, “definição das estratégias das decisões tomadas”, são determinadas para cada tomador de decisão, as estratégias escolhidas por eles. Para cada tomador de decisão foi criada uma árvore de decisão ID3 para representar suas estratégias tomadas.

No próximo estágio, “definição das conseqüências das estratégias escolhidas”, o KBI é utilizado para comparar o desempenho de cada tomador de decisão, assim a melhor estratégia pode ser identificada, lembrando que ela é a melhor dentre as que foram apresentadas.

No último passo, “busca por melhorias”, é usada a capacidade das árvores de decisões de representar todos os passos das escolhas (por meio do *backtracking*). As estratégias individuais são analisadas por todos os tomadores de decisões, buscando um aprimoramento das estratégias.

Essa metodologia foi testada e aplicada a uma linha de montagem da companhia Ford. Foram investigadas as decisões tomadas quando ocorriam eventos não planejados, nesse caso apenas quebra de máquinas.

Jinsong *et al* (2005) tratam a produção de produto orientada a configuração e a gestão do conhecimento para empresas manufatureiras que adotam a política de produção sob encomenda (MTO – *Make To Order*). É proposto um processo de modelagem genérica para a configuração do produto, representando as famílias de produtos com suas especificações e variações.

Os autores afirmam que os gerenciadores de dados do produto (PDM – *Product Data Management*) não dão suporte efetivo a configuração dos produtos por serem limitados em relação à representação do conhecimento. Diante disso eles focaram em uma modelagem de produto orientados à configuração e gestão do conhecimento.

Também apresentam uma organização e gestão do conhecimento do produto por meio de um componente de conhecimento (KCOM – *Knowledge COMponent*) no qual estão incluídas as regras de configurações e restrições.

Para a organização do conhecimento do produto, foram propostos três modelos: um modelo de montagem, que apresenta as relações entre as peças para produzir o produto e foi representado por meio de uma árvore de hierarquias; o outro é um modelo funcional do produto, que foca em uma abstração e categorização dos produtos em função de suas características; o terceiro é o modelo de configuração do produto representado por um metamodelo que apresenta a configuração entre as entidades e suas relações.

Sabuncuoglu e Kizilisik (2003) estudaram os problemas pertinentes a programação reativa em sistemas de manufatura estocásticos e dinâmicos. A partir desses estudos, foi desenvolvido um sistema de programação baseado em simulação para sistemas flexíveis de manufatura. Esse sistema leva em consideração várias políticas reativas de programação propostas pelos próprios autores.

O sistema de programação proposto é composto por três módulos principais: programador, modelo de simulação e controlador. O módulo do programador é responsável por realizar as decisões relativas à programação, levando em consideração o *status* atual do sistema. O módulo de simulação é responsável por simular as tomadas de decisão do módulo programador e invoca esse módulo cada vez que é necessária uma tomada de decisão. O terceiro módulo, controlador, é responsável por aplicar as regras de programação de acordo com as condições do ambiente ao longo do tempo.

Para representar o conhecimento sobre a programação reativa da produção, foi escolhida a técnica de árvores de decisões e aplicado a heurística busca em feixe (*Beam search*) para buscar pela melhor solução. As regras reativas de produção foram divididas em duas categorias: quando realizar a programação; e como realizar a programação. Cada uma dessas categorias contém várias regras de programação.

Foi realizado um teste por meio de simulação em uma fábrica contendo seis máquinas cada uma delas com *buffer* de entrada. Também foram considerados veículos autoguiados, estações de carga e descarga. O desempenho do algoritmo levou em consideração: capacidade de *buffer*, flexibilidade da seqüência, flexibilidade da rota, fator de atrasos, variantes do tempo de processo e nível de máquina quebrada.

Vários testes foram realizados, alterando as políticas reativas de programação com o intuito de avaliar a eficiência de cada uma delas. O algoritmo apresentou-se flexível suficiente para contornar os problemas ocorridos em tempo real.

Henning e Cerdá (2000) apresentam um *framework* baseado em conhecimento, utilizando tecnologia de orientação a objeto para construir sistemas de programação visando resolver problemas reais. São apresentados os aspectos relevantes da arquitetura do *framework* que pode ser usado tanto para programação reativa quanto a preditiva.

Os autores apresentam duas aplicações industriais que utilizam o *framework* proposto. As três aplicações utilizam um sistema de suporte à programação (SSS – *Scheduling Support System*) similar. A arquitetura é a mesma e a metodologia de solução de problema é bem semelhante.

São apresentados seis princípios que o SSS deve seguir: elevar a capacidade de solução de problemas sem substituir o especialista; capturar os modelos complexos dos produtos; representar claramente os modelos dos recursos, incluindo a estrutura complexa das redes de processamento e das políticas de operação associadas a elas; capturar o conhecimento da solução do problema, permitindo a flexibilidade do reuso e adaptação de

algoritmos de programação; combinar diferentes componentes baseados em conhecimento; e por fim, prever a função de programação como uma atividade de gestão do conhecimento.

A arquitetura em camadas proposta representa o domínio real por meio da modelagem orientada a objeto. Essa modelagem baseia-se em dois exemplos reais para o uso dos dados, utilizando a UML para modelá-los.

São estabelecidos modelos genéricos distintos para os produtos, recursos, programação e pedidos. Em outra camada é descrito um conceito de solução de problema, um para programação preditiva e outra para a reativa. Para a programação reativa são considerados eventos inesperados como quebra de máquina, falta de matéria prima, falta de operador e inconsistências de reprogramação.

Os resultados obtidos nos exemplos citados foram significativos em termos de redução de tempo para realizar a programação, identificação de gargalos na linha de produção, e descoberta de uso de políticas de produção erradas.

O'Kane (2000) propõe uma abordagem para a análise da programação reativa da produção em um sistema flexível de manufatura (FMS – *Flexible Manufacturing Systems*). O sistema desenvolvido de acordo com a abordagem proposta é baseado na tomada de decisão automatizada inteligente por meio da descoberta de conhecimento dos dados do status do FMS, visando à aprendizagem por experiências passadas.

O primeiro estágio da pesquisa consiste no desenvolvimento de um ambiente de tomada de decisão relacionando a simulação com a IA, no qual o elemento inteligente é um sistema especialista baseado em PROLOG capaz de analisar os dados das condições de um FMS, e capaz de oferecer automaticamente conselhos aos usuários.

Esse primeiro estágio foi desenvolvido com o objetivo de extrair dados de maneira automática de um ambiente FMS e realizar inferências para auxiliar a tomada de decisão.

A partir de várias simulações, foi criada uma base de conhecimento com as informações extraídas do primeiro estágio. De posse dessa base, o autor, por meio de regras no sistema especialista, possibilitou o aprendizado de novas regras de produção.

Com base nas regras já aprendidas e nas inferências realizadas, o sistema é capaz de realizar a programação reativa da produção e armazenar as novas tomadas de decisão em uma base para que se tenha o aprendizado.

3 Proposta

A proposta deste trabalho está focada na modelagem do conhecimento, contemplando a aquisição e a representação do conhecimento, explícito e implícito, da tarefa de programação reativa da produção. O intuito dessa modelagem é auxiliar a tomada de decisão do especialista, sendo ele o engenheiro ou o operador, permitindo certa autonomia para executar a programação reativa da produção.

3.1 O Modelo Proposto

Para a criação do modelo que auxilie a tomada de decisão, primeiramente foi necessário realizar a aquisição do conhecimento. Essa atividade foi realizada seguindo as diretrizes de um projeto existente entre o laboratório TEAR e uma indústria, sendo adaptadas ao cenário proposto nesse trabalho.

Primeiramente foram detectados alguns dados essenciais para o modelo, e para realizar a aquisição deles foram utilizadas tabelas, as quais foram preenchidas pelos especialistas do domínio. É importante ressaltar que durante a aquisição das informações, também foi realizada a validação das mesmas, passando pelo crivo dos especialistas após serem estruturadas.

À medida que a aquisição do conhecimento foi progredindo, realizaram-se também as atividades de representação. Essas atividades consistem em modelar os conhecimentos levantados na aquisição de uma maneira que seja fácil para o especialista do domínio entender, e conseqüentemente validar, e que ao mesmo tempo tenha aderência ao objetivo desejado.

Foram levantados os roteiros de fabricação de todos os produtos por meio de tabelas de aquisição e representados por redes de Petri, visando à facilidade que a representação gráfica traz na hora da validação com o especialista. Um exemplo dessa tabela de aquisição é apresentado na Tabela 3.1, e a representação correspondente dessa tabela em redes de Petri é ilustrada na Figura 3.1.

Tabela 3.1 Exemplo de uma tabela de aquisição do conhecimento sobre o roteiro de fabricação do produto P1.

Família	Produto	Roteiro
A	P1	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 11 / Operador
5	A5	Máquina 12 / Operador

A tabela de aquisição do conhecimento foi desenvolvida para realizar a aquisição do conhecimento de como é produzido um produto específico. Nessa tabela existem campos para a identificação do produto; identificação da família a qual esse produto pertence; identificação do roteiro de fabricação, pois existem produtos que possuem roteiros alternativos; os processos pelos quais esse produto passa; e os recursos necessários para produzir o produto, ligados a processos específicos.

A modelagem em redes de Petri, exemplo apresentado na Figura 3.1, é uma representação gráfica do conhecimento contido na tabela. Ele representa o fluxo de produção do produto, mostrando desde a entrada da matéria prima até o produto finalizado, indicando o momento em que os recursos são utilizados.

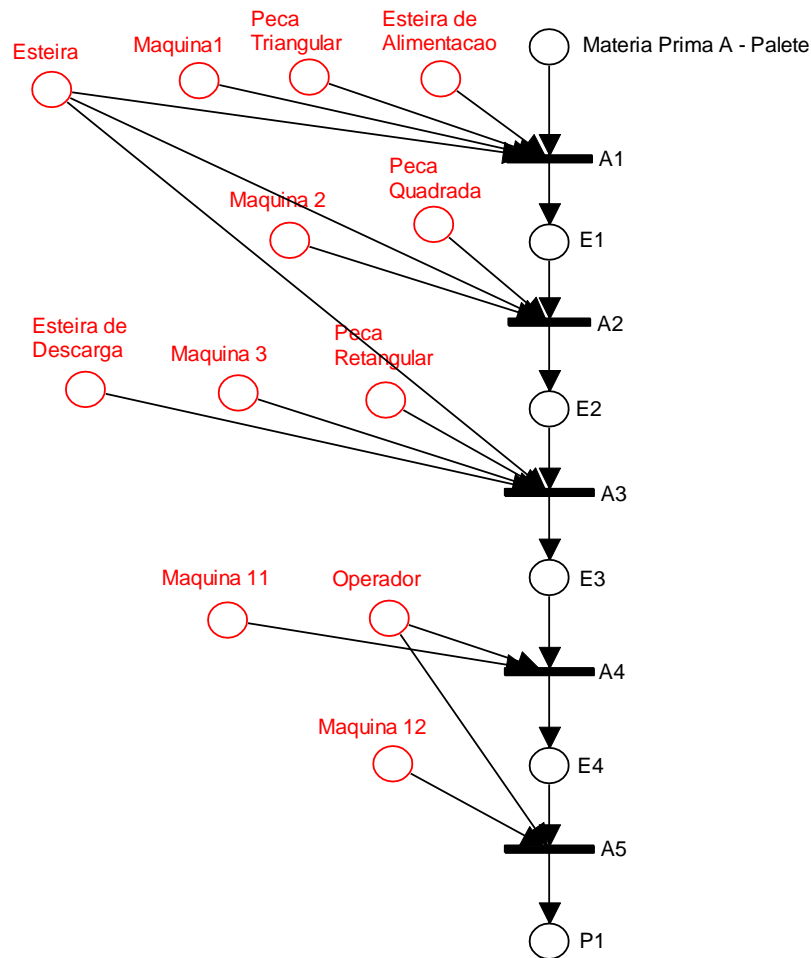


Figura 3.1 Exemplo de modelagem do conhecimento sobre o roteiro de fabricação do produto P1 em redes de Petri.

As ocorrências existentes no domínio foram identificadas e levantadas por meio de tabelas para a aquisição do conhecimento, sendo apresentado um exemplo na Tabela 3.2. Essas tabelas foram traduzidas em casos de uso (Tabela 3.3) e para facilitar a validação da modelagem do conhecimento sobre a ocorrência com o especialista, foram criados fluxogramas (Figura 3.2) baseados nesses casos de uso.

Tabela 3.2 Tabela contendo o conhecimento sobre a ocorrência quebra de máquina.

Família	Ocorrência	Produto	Alternativas	Condicionantes
A	Quebra de Máquina	P1/P2/P3/P4/P5/P6	Verificar a existência de roteiros alternativos	Se houver roteiros alternativos disponíveis.
B		P7/P8/P9/P10/P11/P12	Excluir produto da programação	Se não houver outro roteiro disponível.
C		P13/P14/P15/P16/P17/P18		

Essa tabela da aquisição do conhecimento sobre as ocorrências é composta de cinco campos, sendo eles: Família e Produto, para identificar em quais famílias e em quais produtos há o relato dessas ocorrências; o campo Ocorrência especifica a ocorrência tratada

pela tabela; o campo Alternativas relata quais alternativas são tomadas em decorrência do problema; e o campo Condicionantes, que serve para informar quais são as condições necessárias para realizar a alternativa em questão.

Após a devida validação da tabela de ocorrências, foram criados casos de uso para formalizar toda a informação contida em cada tabela. Foi criado um caso de uso para cada ocorrência identificada, o exemplo do caso de uso da ocorrência quebra de máquina (Tabela 3.2) é apresentado na Tabela 3.3.

Tabela 3.3 Caso de Uso da Ocorrência Quebra de Máquina.

Especificação do Caso de Uso	Número:	1
Nome do Caso de Uso	Ocorrência quebra de máquina	
Descrição ou Resumo	Ocorre quando uma ou mais máquinas, utilizadas na produção de um produto que esteja presente na programação atual, se torna indisponível.	
Ator Participante	Engenheiro de Programação	
Ator Operador	Operador	
Pré-Condição	Indisponibilidade de uma ou mais máquinas utilizadas na programação atual	
Curso Normal	1 - Identificação da quebra da máquina. 2 – Indicar quais máquinas estão indisponíveis. 3 – Buscar roteiros de produção para identificação dos roteiros que usam essas máquinas. 4 – Avaliar quais produtos serão afetados pelos roteiros indisponíveis. 5 – Identificar quais produtos presentes na programação foram afetados. 6 – Para os produtos identificados no passo anterior, procurar por roteiros alternativos disponíveis, caso haja. 7 – Se houver roteiro alternativo disponível, alterar o roteiro. 8 – Se não houver roteiro alternativo disponível, retirá-lo da programação.	
Curso Alternativo	X	
Evento Disparador	Existência de máquina quebrada	
Requisitos Funcionais	Máquinas, Operador.	
Requisitos Não Funcionais	X	
Data	16/03/2009	
Versão	1.0.0	

Essa tabela de caso de uso é uma descrição formal de como é feito o tratamento da ocorrência de máquina quebrada. Ela é composta por doze campos todos eles para descrever cada caso de uso. O primeiro campo identifica qual o caso de uso é tratado na tabela, o segundo é uma breve descrição ou um resumo do tratamento da ocorrência.

Os campos três e quatro são responsáveis por identificar os atores presentes. O quinto campo, Pré-Condição, identifica as necessidades para ocorrer esse evento. O sexto, Curso Normal, descreve passo a passo as ações necessárias para executar o tratamento da ocorrência em questão. O sétimo campo, Curso Alternativo, descreve uma alternativa ao fluxo normal do tratamento da ocorrência.

O oitavo campo, Evento Disparador, é a identificação daquilo que dispara esse tipo de ocorrência. Os campos nove e dez relatam os requisitos funcionais e não funcionais

presentes nessa ocorrência. Por último, os campos onze e doze servem de controle para a modelagem do caso de uso.

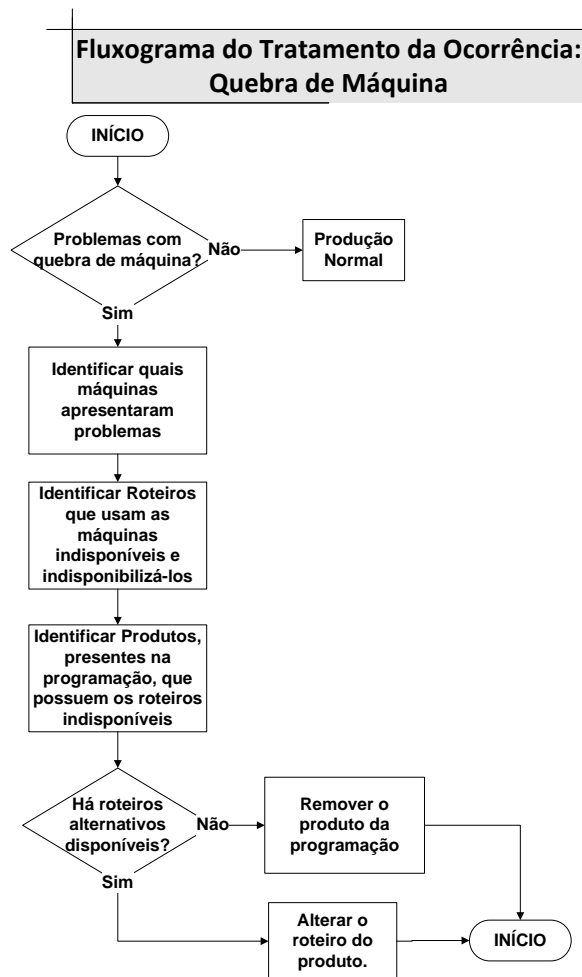


Figura 3.2 Fluxograma da Ocorrência quebra de máquina.

Após a formalização com o caso de uso, para a devida validação com os especialistas do domínio, foram criados fluxogramas descrevendo de maneira gráfica o fluxo normal do caso de uso. Foi escolhido o fluxograma por ser de fácil compreensão e já conhecido pelos especialistas da área.

Na Figura 3.2 é apresentado o fluxograma referente ao caso de uso da “ocorrência quebra de máquina” correspondente a Tabela 3.2. São descritos todos os passos que normalmente o engenheiro de produção executa quando está diante desse problema.

3.2 Funcionalidades

Após a etapa de modelagem do conhecimento levantado, é possível visar os aspectos funcionais do sistema. Podemos pensar na estrutura atendendo duas funcionalidades macro, sendo elas: configuração do sistema e uso do sistema.

A funcionalidade macro de configuração do sistema deve garantir que seja possível inserir todos os dados de entrada necessários para que o modelo funcione corretamente. Esses dados de configuração de entrada caracterizam o ambiente no qual o modelo irá realizar as inferências.

Podemos dividir essa funcionalidade macro em algumas tarefas específicas, sendo elas:

- Inclusão da programação planejada;
- Inclusão da lista de produtos preferenciais;
- Inclusão, exclusão e alteração de produtos.

A inclusão da programação planejada é uma tarefa que deve ser executada pelo engenheiro de programação, definindo-se a programação inicial que o modelo terá como base para a realização da programação reativa da produção.

A lista de produtos preferenciais são aqueles produtos que, no caso do impedimento da produção de algum produto programado *a priori*, são candidatos a serem produzidos pelo fato de que há um grande fluxo de pedidos do produto.

A inclusão desses produtos deve ser realizada pelo engenheiro de produção, pois pela experiência, ele é capaz de identificar os produtos que logo terão a necessidade de serem produzidos.

O sistema deverá permitir que o engenheiro adicione produtos que a princípio não estão relacionados na base de dados, garantindo a possibilidade de criação e produção de novos produtos. Também visando uma maior flexibilidade do sistema, será possível alterar características dos produtos já cadastrados, sendo essas características: roteiros de fabricação, margem de contribuição e nível de estoque.

A outra funcionalidade macro está relacionada ao uso do sistema que pode ser realizada tanto pelo engenheiro de produção quanto pelo operador. Essa funcionalidade macro é utilizada quando o usuário necessita de um apoio à decisão, para o caso da necessidade de interferir na programação planejada, adicionando ou removendo produtos.

Essa interferência se dá pelas ocorrências já mapeadas e modeladas na etapa de aquisição e representação do conhecimento, sendo cada uma delas tratada por uma função específica. Essa função avalia os dados de entrada necessários e como resultado desse tratamento, realiza a programação reativa da produção.

Sendo cinco tipos de ocorrências levantadas, foram criadas cinco funções: função de exclusão por roteiro, função exclusão/inclusão dura, função de exclusão direta, função de inclusão direta e função falta de operador.

Para detalhar o fluxo da funcionalidade macro do uso do sistema, está apresentado na Figura 3.3, o fluxograma da funcionalidade macro, destacando as cinco funções de tratamento das ocorrências.

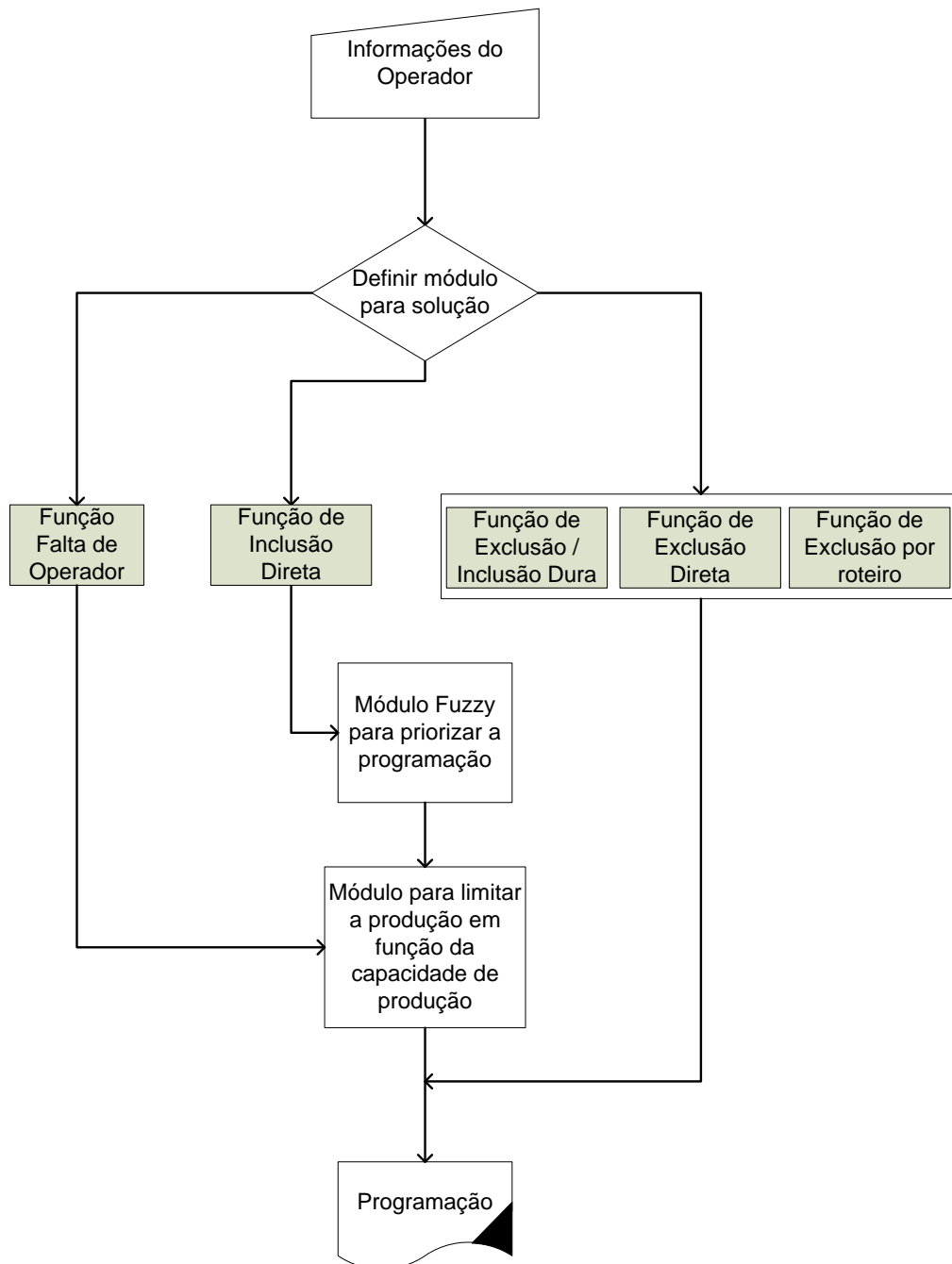


Figura 3.3 Funcionalidade macro do uso do sistema.

Inicialmente o operador ou o engenheiro entrará com os dados necessários para alimentar o sistema indicando qual foi o problema ocorrido.

Após essa informação, é verificada qual é a função responsável pelo tratamento daquele tipo de ocorrência. Depois de identificada a função responsável, as devidas operações são realizadas para determinar quais produtos saem e/ou quais entram na programação podendo passar ou não por um cálculo *fuzzy* e por um cálculo para limitar a produção em função da capacidade.

Detalhando mais cada uma das funções, temos a função de exclusão por roteiro que é responsável por tratar a ocorrência de quebra de máquina. Quando ocorre a indisponibilidade de alguma máquina, mapeiam-se os roteiros nos quais essa máquina está presente com o intuito de desconsiderá-los. São verificados todos os produtos presentes na programação atual, para identificar quais deles utilizam os roteiros indisponíveis.

À medida que vão sendo encontrados, é verificado se o produto possui algum roteiro alternativo disponível; em caso afirmativo, troca-se o roteiro pelo alternativo, caso contrário, esse produto é retirado da programação, conforme apresentado na Figura 3.2.

A função de inclusão/exclusão dura garante a robustez do modelo, permitindo que o usuário adicione ou retire produtos de maneira forçada. Essa função foi desenvolvida caso o usuário perceba algum evento não mapeado anteriormente, e possa, ainda assim, alterar a programação reativa apresentada pelo modelo, conforme apresentado na Figura 3.4.

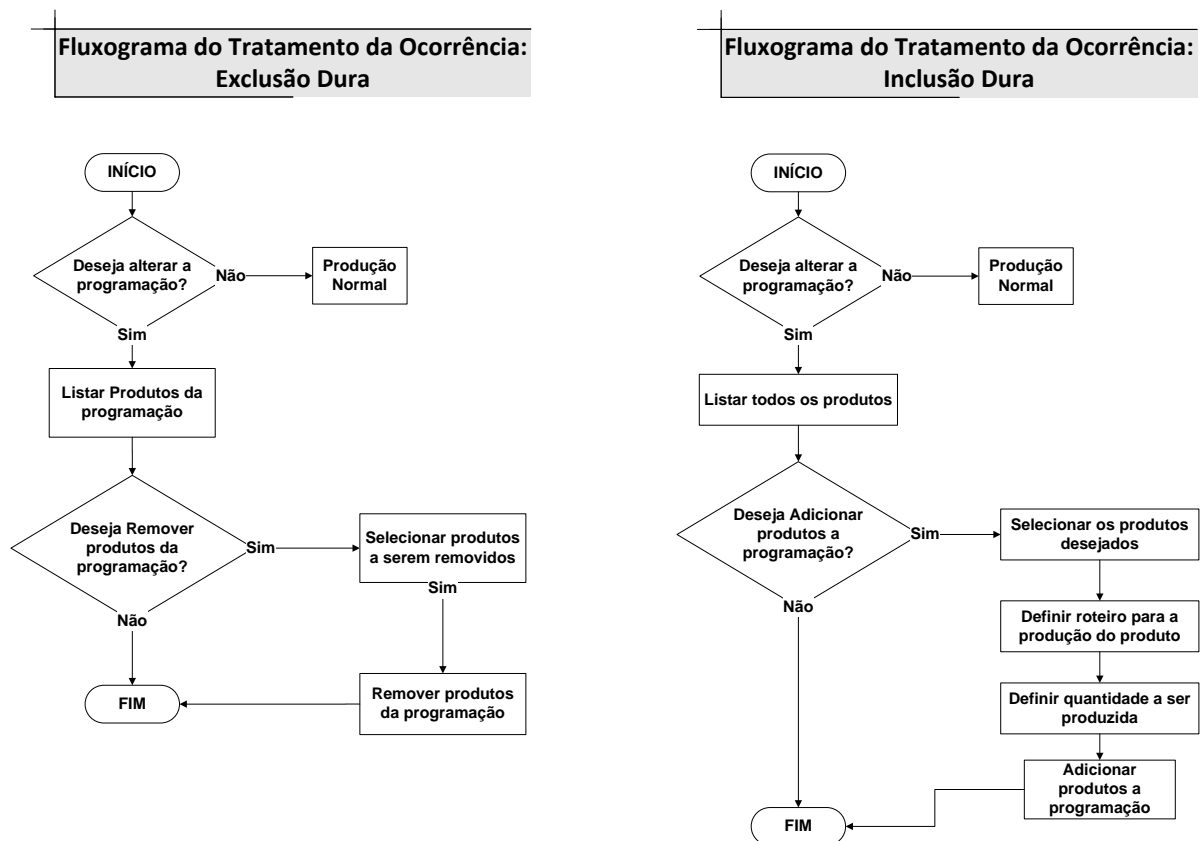


Figura 3.4 Fluxograma do tratamento da ocorrência de inclusão/exclusão de produtos.

A função de exclusão direta trata os problemas de falta de local para estocagem e de falta de matéria prima. Para a falta de local de estocagem, primeiramente é verificado produto a produto presente na programação e o seu respectivo local de estocagem, caso esteja cheio, esse produto é retirado da programação, conforme apresentado na Figura 3.5.

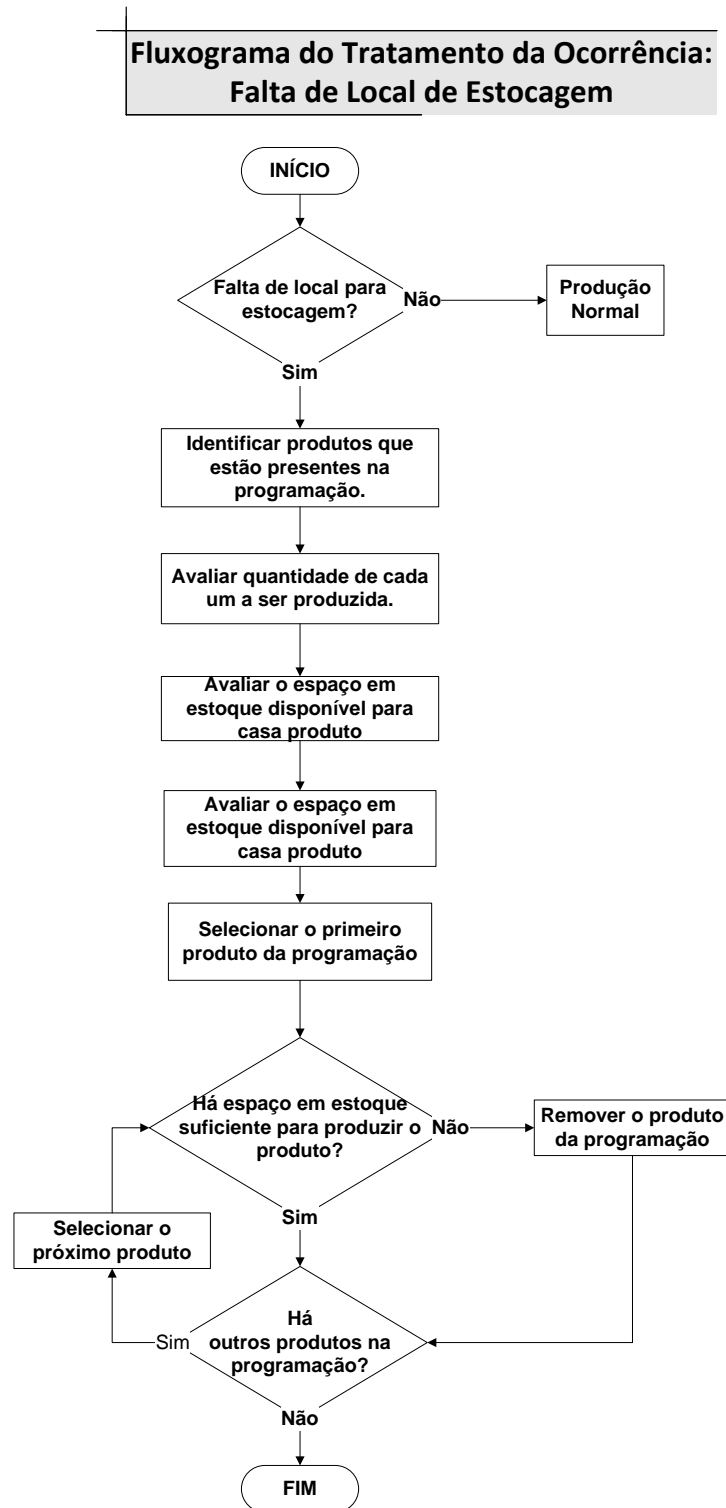


Figura 3.5 Fluxograma do modulo tratamento da ocorrência de falta de local para estocagem.

Já para a falta de matéria prima, verificam-se quais produtos dentro da programação utilizam a matéria prima que está em falta, em seguida esses produtos são retirados da programação, conforme ilustrado na Figura 3.6.

A função de inclusão direta é responsável por adicionar produtos à programação caso haja a necessidade de eliminar alguma matéria prima do *buffer*. Primeiramente verifica-se a lista de inclusão, já proposta pelo programador, e avalia se é possível produzir algum produto que esteja presente nessa lista com a matéria prima em questão.

Após a escolha dos produtos presentes na lista de inclusão, é necessário verificar a quantidade que se pode produzir, devido ao número limitado do estoque do produto. Caso não esgote a matéria prima, é necessário avaliar quais produtos não estão na lista de inclusão e que utilizam a matéria prima em questão.

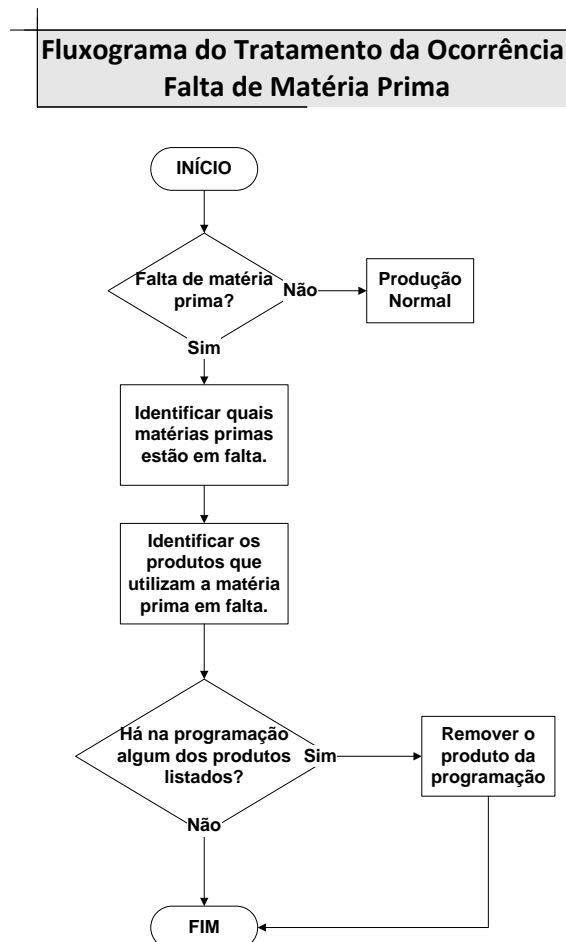


Figura 3.6 Fluxograma do tratamento da ocorrência de falta de matéria prima.

Após identificar esses produtos, é escolhido aquele que tem a maior margem de contribuição e que tenha espaço em estoque. Adotam-se essas avaliações até que se esgote a matéria prima ou até que não tenha mais produtos que atendam essas exigências.

Caso acabem as opções de produtos, segundo as condições citadas anteriormente, e ainda tenha matéria prima no *buffer* de entrada, simplesmente adota-se a estratégia de produzir o de maior margem de contribuição, mesmo que não tenha espaço em estoque, até que acabe com toda a matéria prima presente no *buffer*.

Todo esse procedimento é representado pelo fluxograma apresentado na Figura 3.7.

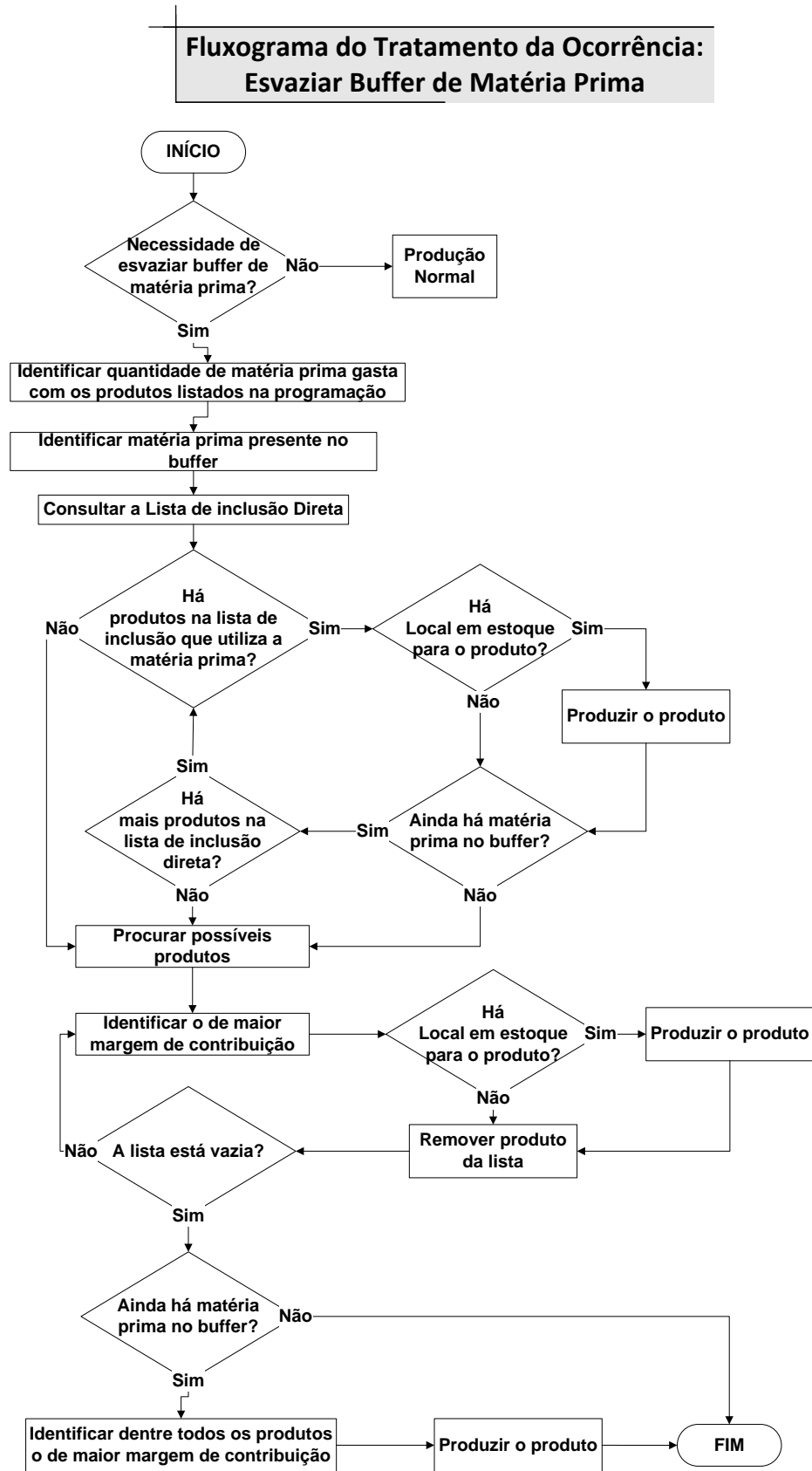


Figura 3.7 Fluxograma do tratamento da ocorrência de esvaziar *buffer* de matéria prima.

A função falta de operador é tratada de maneira diferente por não incluir nem excluir produtos à programação normal, apenas realocar a capacidade de produção em função dos operadores disponíveis, conforme apresentado na Figura 3.8.

Primeiramente, avaliam-se quantos operadores estão disponíveis. Aloca-se um operador para cada produto até que se esgotem os operadores. Essa alocação é realizada obedecendo à ordem da programação. Parte-se do pressuposto de que a ordem da programação já obedece à ordenação da prioridade.

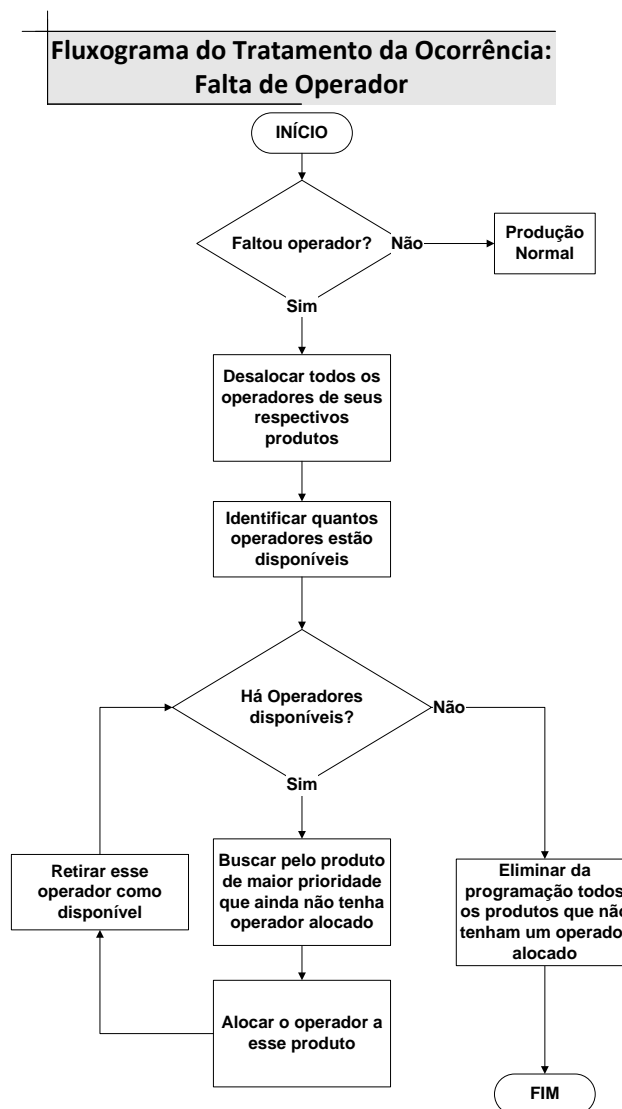


Figura 3.8 Fluxograma do tratamento da ocorrência de falta de operador.

No modelo proposto, criou-se um módulo *fuzzy* para realizar o cálculo da nova prioridade. Esse cálculo é necessário após a inclusão de novos produtos na programação, para definir a prioridade com que esses produtos deverão ser produzidos.

A nova prioridade é calculada ponderando as variáveis: “nível de estoque”, “data de entrega”, “margem de contribuição do produto” e “prioridade inicial”. Essas

variáveis foram levantadas em conjunto com o especialista considerando as variáveis de maior impacto na hora de definir a programação.

Após entrevistas não estruturadas, verificou-se que o especialista na hora da programação pondera o nível de estoque de cada produto e a carteira de pedidos, avaliando quais produtos têm que ser entregues primeiro. Durante a entrevista, identificou-se que a margem de contribuição ajudaria na programação caso também fosse levada em conta.

Para a modelagem *fuzzy*, utilizou-se o sistema *fuzzy* apresentado (Figura 3.9), sendo determinadas quatro variáveis de entrada, que são “nível de estoque”, “data devida”, “prioridade inicial” e “margem de contribuição do produto”; uma variável de saída: “nova prioridade”; uma base de regras; e para a máquina de inferência utilizou-se Mamdani.

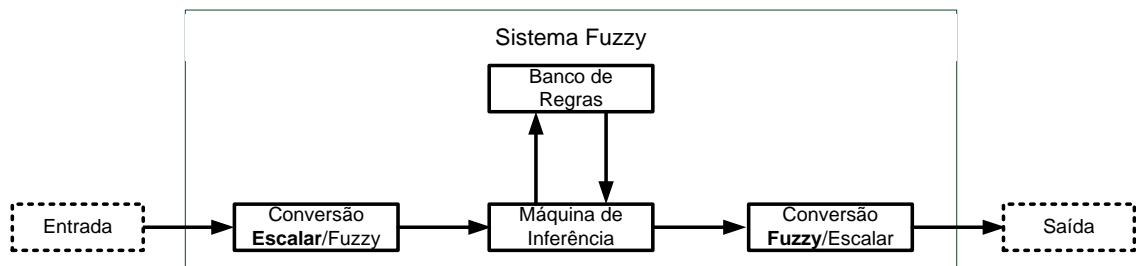


Figura 3.9 Sistema do módulo *fuzzy* – (adaptado de REZENDE, 2003).

As funções que representam os conjuntos *fuzzy* das variáveis, bem como seus domínios, foram determinadas de maneira empírica.

A variável “nível de estoque” é dividida em três conjuntos *fuzzy* triangulares, tendo as seguintes variáveis lingüísticas: “alto”, “médio”, “baixo”. O conjunto “baixo” pertence ao intervalo de 0 a 50 peças; o conjunto “médio” ao intervalo de 25 a 75 peças; e o conjunto “alto” ao intervalo de 50 a 100 peças (Figura 3.10).

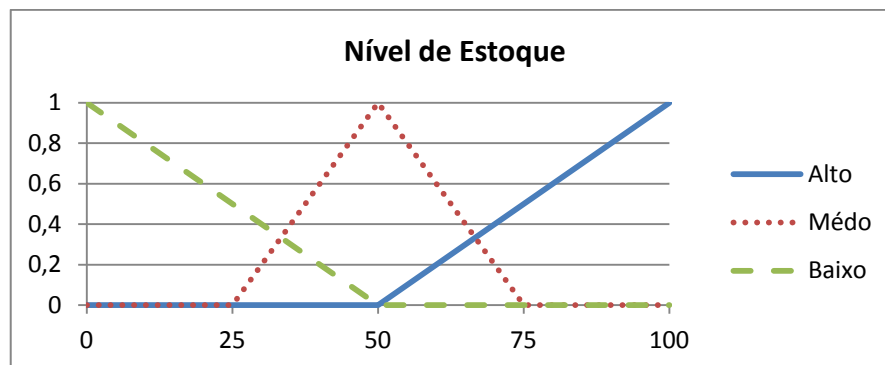


Figura 3.10 Conjuntos da variável de entrada “nível de estoque”.

A variável “data devida” é dividida em três conjuntos *fuzzy* triangulares, tendo como variáveis lingüísticas: “próxima”, “média” e “distante”. O conjunto “próxima” pertence

ao intervalo de 0 a 5 dias; o conjunto “média” ao intervalo de 2 a 7 dias; e o conjunto “distante” ao intervalo de 5 a 10 dias (Figura 3.11).

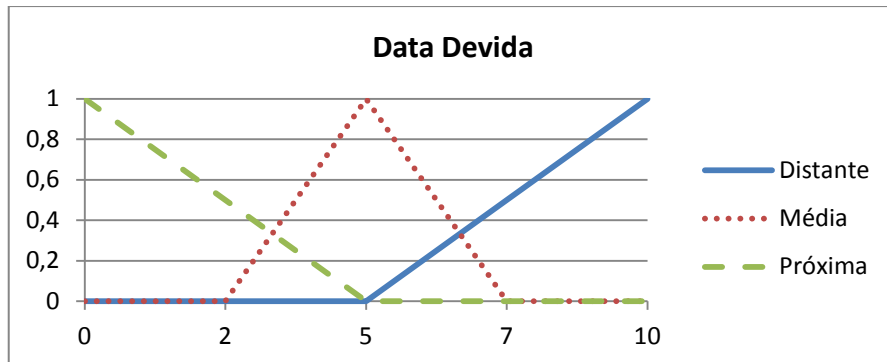


Figura 3.11 Conjuntos da variável de entrada "data devida".

A variável “prioridade inicial” também é dividida em três conjuntos *fuzzy* triangulares, tendo como variáveis lingüísticas: “alta”, “média” e “baixa”. O conjunto “baixa” pertence ao intervalo de 0 a 5; o conjunto “média” de 2 a 7; e o conjunto “alta” ao intervalo de 5 a 10 (Figura 3.12)

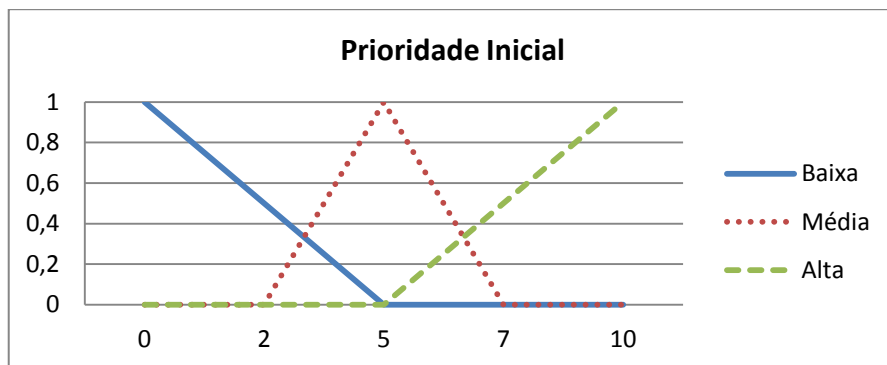


Figura 3.12 Conjuntos da variável de entrada "Prioridade Inicial".

A variável “margem de contribuição do produto” também é dividida em três conjuntos *fuzzy* triangulares, tendo como variáveis lingüísticas: “alta”, “média” e “baixa”. O conjunto “baixa” pertence ao intervalo de 0 a 5; o conjunto “média” ao intervalo de 2 a 7; e o conjunto “alta” ao intervalo de 5 a 10 (Figura 3.13).

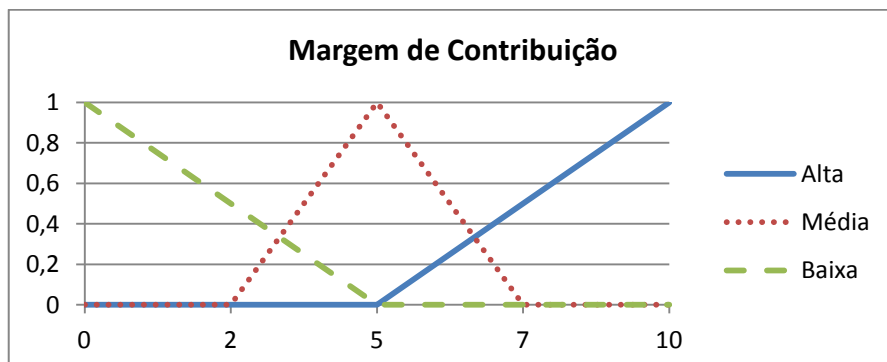


Figura 3.13 Conjuntos da variável de entrada "margem de contribuição".

A variável de saída, “nova prioridade” foi dividida em três conjuntos *fuzzy* triangulares, tendo como variáveis lingüísticas: “baixa”, “média” e “alta”. O conjunto “baixa” pertence ao intervalo de 0 a 5; o conjunto “média” ao intervalo de 2 a 7; e o conjunto “alta” ao intervalo de 5 a 10 (Figura 3.14).

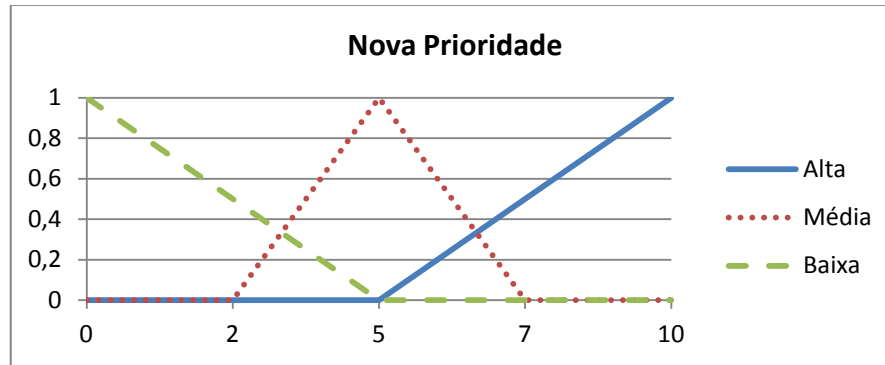


Figura 3.14 Conjuntos da variável de saída "margem de contribuição".

Para a base de regras, também determinada empiricamente, utilizaram-se todas as combinações possíveis entre as variáveis de entrada, pois não era o objetivo a otimização da base de regras. Essa base de regras é apresentada na Tabela 3.4.

Tabela 3.4. Base de regras.

Nº	Variáveis de entrada				Variáveis de saída
	“Prioridade Inicial”	“Nível de estoque”	“Data Devida”	“Margem de contribuição”	“Nova prioridade”
1º	Alta	Alto	Distante	Alta	Baixa
2º	Alta	Alto	Distante	Média	Baixa
3º	Alta	Alto	Distante	Baixa	Baixa
4º	Alta	Alto	Média	Alta	Baixa
5º	Alta	Alto	Média	Média	Baixa
6º	Alta	Alto	Média	Baixa	Baixa
7º	Alta	Alto	Próximo	Alta	Baixa
8º	Alta	Alto	Próximo	Média	Baixa
9º	Alta	Alto	Próximo	Baixa	Baixa
10º	Alta	Médio	Distante	Alta	Baixa
11º	Alta	Médio	Distante	Média	Baixa
12º	Alta	Médio	Distante	Baixa	Baixa
13º	Alta	Médio	Média	Alta	Média
14º	Alta	Médio	Média	Média	Média
15º	Alta	Médio	Média	Baixa	Média
16º	Alta	Médio	Próximo	Alta	Alta
17º	Alta	Médio	Próximo	Média	Média
18º	Alta	Médio	Próximo	Baixa	Baixa
19º	Alta	Baixo	Distante	Alta	Média
20º	Alta	Baixo	Distante	Média	Média
21º	Alta	Baixo	Distante	Baixa	Baixa
22º	Alta	Baixo	Média	Alta	Alta
23º	Alta	Baixo	Média	Média	Média
24º	Alta	Baixo	Média	Baixa	Baixa
25º	Alta	Baixo	Próximo	Alta	Alta
26º	Alta	Baixo	Próximo	Média	Alta

27°	Alta	Baixo	Próximo	Baixa	Média
28°	Média	Alto	Distante	Alta	Baixa
29°	Média	Alto	Distante	Média	Baixa
30°	Média	Alto	Distante	Baixa	Baixa
31°	Média	Alto	Média	Alta	Baixa
32°	Média	Alto	Média	Média	Baixa
33°	Média	Alto	Média	Baixa	Baixa
34°	Média	Alto	Próximo	Alta	Baixa
35°	Média	Alto	Próximo	Média	Baixa
36°	Média	Alto	Próximo	Baixa	Baixa
37°	Média	Médio	Distante	Alta	Baixa
38°	Média	Médio	Distante	Média	Baixa
39°	Média	Médio	Distante	Baixa	Baixa
40°	Média	Médio	Média	Alta	Média
41°	Média	Médio	Média	Média	Média
42°	Média	Médio	Média	Baixa	Média
43°	Média	Médio	Próximo	Alta	Alta
44°	Média	Médio	Próximo	Média	Média
45°	Média	Médio	Próximo	Baixa	Média
46°	Média	Baixo	Distante	Alta	Média
47°	Média	Baixo	Distante	Média	Média
48°	Média	Baixo	Distante	Baixa	Baixa
49°	Média	Baixo	Média	Alta	Alta
50°	Média	Baixo	Média	Média	Média
51°	Média	Baixo	Média	Baixa	Baixa
52°	Média	Baixo	Próximo	Alta	Alta
53°	Média	Baixo	Próximo	Média	Alta
54°	Média	Baixo	Próximo	Baixa	Média
55°	Baixa	Alto	Distante	Alta	Baixa
56°	Baixa	Alto	Distante	Média	Baixa
57°	Baixa	Alto	Distante	Baixa	Baixa
58°	Baixa	Alto	Média	Alta	Baixa
59°	Baixa	Alto	Média	Média	Baixa
60°	Baixa	Alto	Média	Baixa	Baixa
61°	Baixa	Alto	Próximo	Alta	Baixa
62°	Baixa	Alto	Próximo	Média	Baixa
63°	Baixa	Alto	Próximo	Baixa	Baixa
64°	Baixa	Médio	Distante	Alta	Baixa
65°	Baixa	Médio	Distante	Média	Baixa
66°	Baixa	Médio	Distante	Baixa	Baixa
67°	Baixa	Médio	Média	Alta	Média
68°	Baixa	Médio	Média	Média	Média
69°	Baixa	Médio	Média	Baixa	Média
70°	Baixa	Médio	Próximo	Alta	Alta
71°	Baixa	Médio	Próximo	Média	Média
72°	Baixa	Médio	Próximo	Baixa	Baixa
73°	Baixa	Baixo	Distante	Alta	Média
74°	Baixa	Baixo	Distante	Média	Média
75°	Baixa	Baixo	Distante	Baixa	Baixa
76°	Baixa	Baixo	Média	Alta	Alta
77°	Baixa	Baixo	Média	Média	Média
78°	Baixa	Baixo	Média	Baixa	Baixa
79°	Baixa	Baixo	Próximo	Alta	Alta
80°	Baixa	Baixo	Próximo	Média	Alta
81°	Baixa	Baixo	Próximo	Baixa	Média

A função para limitar a produção em função de sua capacidade, é responsável por determinar quantos produtos que estão na programação serão produzidos. Essa necessidade se dá devido ao fato de que durante a execução do modelo, é possível adicionar novos produtos à programação ou faltar operadores para a execução de todos.

Os produtos, após serem re-classificados pelo módulo *fuzzy*, são ordenados de acordo com sua nova prioridade. Após esse passo, determinam-se quantos produtos poderão ser produzidos de acordo com a capacidade de produção.

Também ocorre essa limitação da produção de acordo com sua capacidade no caso da falta de operador, porém essa não passa pelo módulo *fuzzy*.

O cálculo para a capacidade de produção leva em conta o número de operários disponíveis e a quantidade de matéria prima. Após determinar os produtos que poderão ser produzidos, é apresentada ao usuário a nova programação determinada pelo modelo.

3.3 Arquitetura do Sistema

Após a identificação das funcionalidades, pensou-se na estrutura para o sistema que irá apoiar o especialista na hora da tomada de decisão. Para a representação da estrutura, optou-se por utilizar diagrama de blocos (Figura 3.15) em conjunto com fluxogramas.

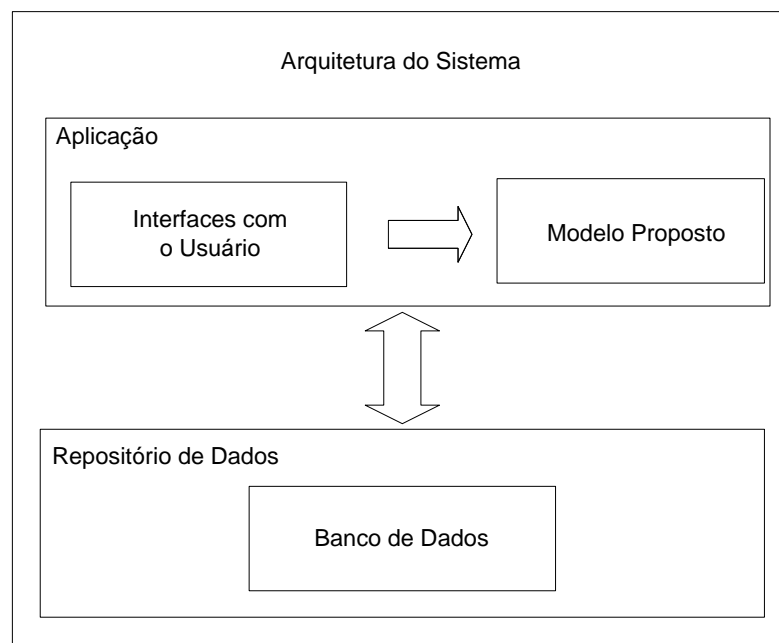


Figura 3.15 Arquitetura do Sistema.

Toda a manipulação do sistema de apoio à decisão será realizada por intermédio das interfaces com o usuário, que enviarão e receberão os dados para o modelo

proposto e para o banco de dados no caso da manutenção dos dados (inserção, atualização e remoção).

O modelo proposto se comunica com a base de dados para a obtenção e alteração dos dados necessários para definir a nova programação, assim como se comunica com as interfaces do usuário para a obtenção de dados necessários para a execução do modelo.

O bloco da base de dados é onde estão armazenados todos os dados sobre o domínio, sendo alguns deles:

- Lista da programação da produção;
- Lista de inclusão dos possíveis produtos;
- Margem de contribuição dos produtos;
- Data de entrega de cada pedido;
- Carteira de pedidos;
- Nível de estoque de cada produto;
- Roteiros de produção.

Essa arquitetura (Figura 3.15) atende as duas funcionalidades macro, sendo a primeira a de configuração do sistema, e a segunda o fluxo de execução do modelo. Para detalhar essas duas regiões optou-se pelo fluxograma por ser uma técnica de representação que atende as necessidades, e por ser de fácil entendimento.

Para detalhar a primeira parte, foi desenvolvido um fluxograma que representa a interação do programador com o banco de dados e do fluxo de dados entre o sistema corporativo e a base de dados (Figura 3.16).

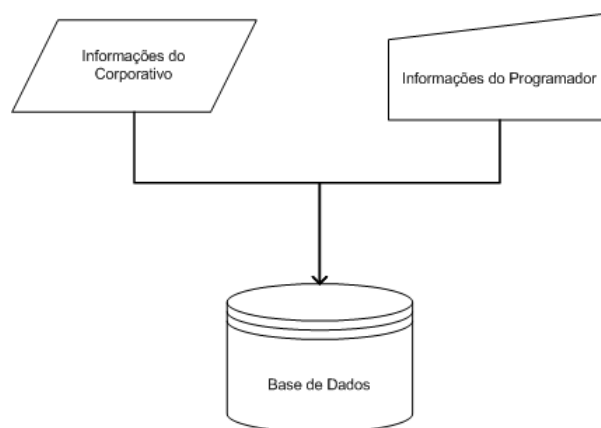


Figura 3.16 Fluxograma da manutenção dos dados.

Para detalhar o funcionamento da segunda região, foi construído um fluxograma descrevendo a execução do modelo e apresentando os módulos existentes para

tratar as ocorrências e as demais informações necessárias para a execução do modelo proposto (Figura 3.3).

3.4 Formas de validação

Cada uma dessas representações de conhecimento foi validada com os especialistas do domínio, para que a construção do modelo represente da maneira mais fiel possível o domínio representado, e que atenda aos objetivos desejados.

Para a validação do modelo proposto, foi criado um sistema computacional de acordo com o modelo proposto. Esse método de validação utilizará os casos de uso levantados anteriormente, se o sistema computacional criado atender de maneira correta todos os casos de uso, significa que ele está de acordo com o planejado.

Porém antes dessa validação do modelo proposto será necessário realizar outra validação, a do código. Para tal tarefa, serão realizadas duas validações: a primeira será o teste unitário de cada módulo, para verificar se ele executa a operação desejada de maneira correta; a segunda será o teste de integração, para verificar se os módulos estão interligados de maneira correta.

Após essas validações é possível analisar se o modelo proposto está cumprindo com seus objetivos de maneira correta.

4 Aplicação do Modelo

Após a especificação do modelo, foi proposto um cenário de aplicação para a criação de um sistema computacional que auxilie a tomada de decisão, com o objetivo principal de validar o modelo proposto.

4.1 Cenário de aplicação

O cenário proposto para a aplicação consiste em um modelo de fábrica que é composta por duas partes: uma parte real e a outra virtual, conforme apresentado na Figura 4.1.

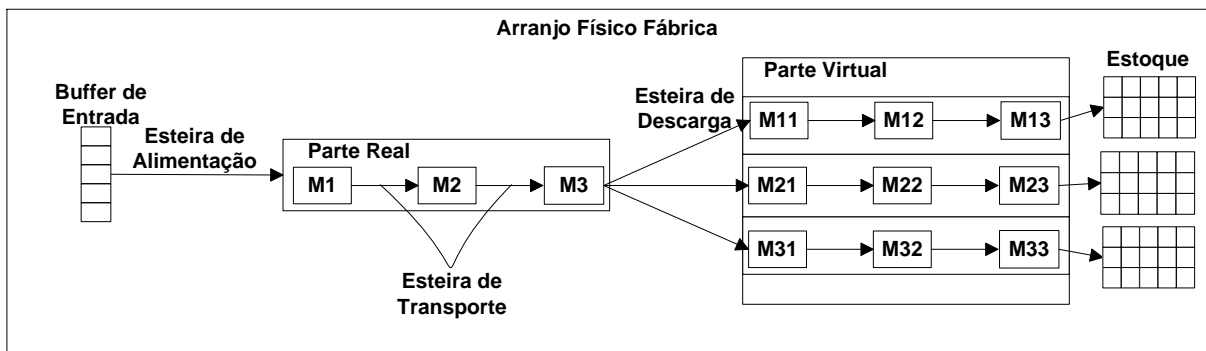


Figura 4.1 Arranjo físico da fábrica.

A parte real, presente no laboratório TEAR, é composta por uma linha contendo três máquinas, sendo elas: M1, M2, M3. Já a parte virtual é composta por nove máquinas divididas em três linhas de produção. A primeira linha contém as máquinas M11, M12, M13; a segunda linha contém as máquinas M21, M22, M23; e a terceira linha contém as máquinas M31, M32, M33.

Além das máquinas, foi proposto um *buffer* de entrada para armazenar a matéria prima e um local de estocagem para os produtos. Outros componentes já existentes na linha física do TEAR são: uma esteira de alimentação para a entrada da matéria prima no processo, uma esteira para transporte do produto pelas máquinas e uma esteira de descarga no final da parte física e início da parte simulada. Esse cenário está apresentado na Figura 4.1.

Essa fábrica é capaz de produzir um total de 18 produtos divididos em três famílias de produtos diferentes, sendo elas: A, B, C. Cada família possui seis produtos, sendo que cada um deles segue um roteiro de fabricação próprio, podendo ou não ter um roteiro alternativo (Tabela 4.1).

Tabela 4.1 Roteiros de Fabricação

FAMÍLIA	PRODUTO	ROTEIRO
A	P1	M1,M2,M3,M11,M12
	P2	M1,M2,M3,M12,M13
	P3	M1,M2,M3,M11,M13
	P4	M1,M2,M3,M11,M12,M13
	P5	M1,M2,M3,M11
	P6	M1,M2,M3,M12 M1,M2,M3,M13
B	P7	M1,M2,M3,M21,M22,M23
	P8	M1,M2,M3,M21,M22
	P9	M1,M2,M3,M21,M23
	P10	M1,M2,M3,M22,M23
	P11	M1,M2,M3,M21
	P12	M1,M2,M3,M22 M1,M2,M3,M23
C	P13	M1,M2,M3,M31,M32,M33
	P14	M1,M2,M3,M31,M32
	P15	M1,M2,M3,M31,M33
	P16	M1,M2,M3,M32,M33
	P17	M1,M2,M3,M31
	P18	M1,M2,M3,M32 M1,M2,M3,M33

Cada um desses produtos, descritos na Tabela 4.1, passa obrigatoriamente pelas três máquinas da parte real para depois passar pela parte virtual. Nas máquinas da parte virtual, os produtos não passam obrigatoriamente por todas elas e respeitam a definição de que todos os produtos pertencentes a uma mesma família passam especificamente por uma única linha de produção da parte virtual.

4.2 Aquisição e Representação dos Roteiros de Fabricação

Após essa caracterização do cenário, foi submetida aos especialistas uma tabela com o intuito de realizar a aquisição do conhecimento sobre os roteiros de fabricação. Essa tabela consiste em um campo para identificar o produto, outro para a identificação da família a qual o produto pertence, outro para a identificação do roteiro, outro para os processos pertencentes a esse produto e outro para os recursos pertencentes a cada processo (Tabela 4.2).

Tabela 4.2 Tabela de aquisição do conhecimento sobre roteiros de fabricação

Família	Produto	Roteiro
Identificação da família	Identificação do produto	Nº do roteiro

	Processo	Recurso
1	Nº do processo	Recurso 1 / Recurso 2 / ... / Recurso N
.	.	.
.	.	.
.	.	.
N	Nº do processo	Recurso 1 / Recurso 2 / ... / Recurso N

Foi criada uma tabela para cada roteiro de produção, sendo que um produto possui de um a dois roteiros. Essa tabela apresenta todos os passos de produção desde a entrada da matéria prima até o produto final, conforme apresentado como exemplo na Tabela 4.3.

Tabela 4.3 Tabela de aquisição do roteiro de fabricação do produto P1.

Família	Produto	Roteiro
A	P1	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 11 / Operador
5	A5	Máquina 12 / Operador

Após o preenchimento da tabela pelos especialistas, o conhecimento sobre o roteiro de fabricação de cada produto foi representado utilizando redes de Petri. Observando que, os produtos que possuem mais de um roteiro, foram representados com apenas um único modelo em redes de Petri. Um exemplo dessa modelagem do conhecimento é apresentado na Figura 4.2.

Na linha principal, iniciada com o círculo “Matéria Prima A – Palete” estão todos os processos representados pelos quais a matéria prima passará até que se tenha o produto final representado pelo lugar “P1”. Essas etapas no meio dos processos estão representadas pelos lugares nomeados de “E” seguidos por um número que representa a ordem com que o produto segue no processo.

Os recursos (lugares a esquerda da linha principal) estão ligados (arcos) a processos (transições) criando uma dependência. Nessa representação não está levando em consideração a ordem com que os recursos serão consumidos, apenas apontando em qual processo ele está ligado.

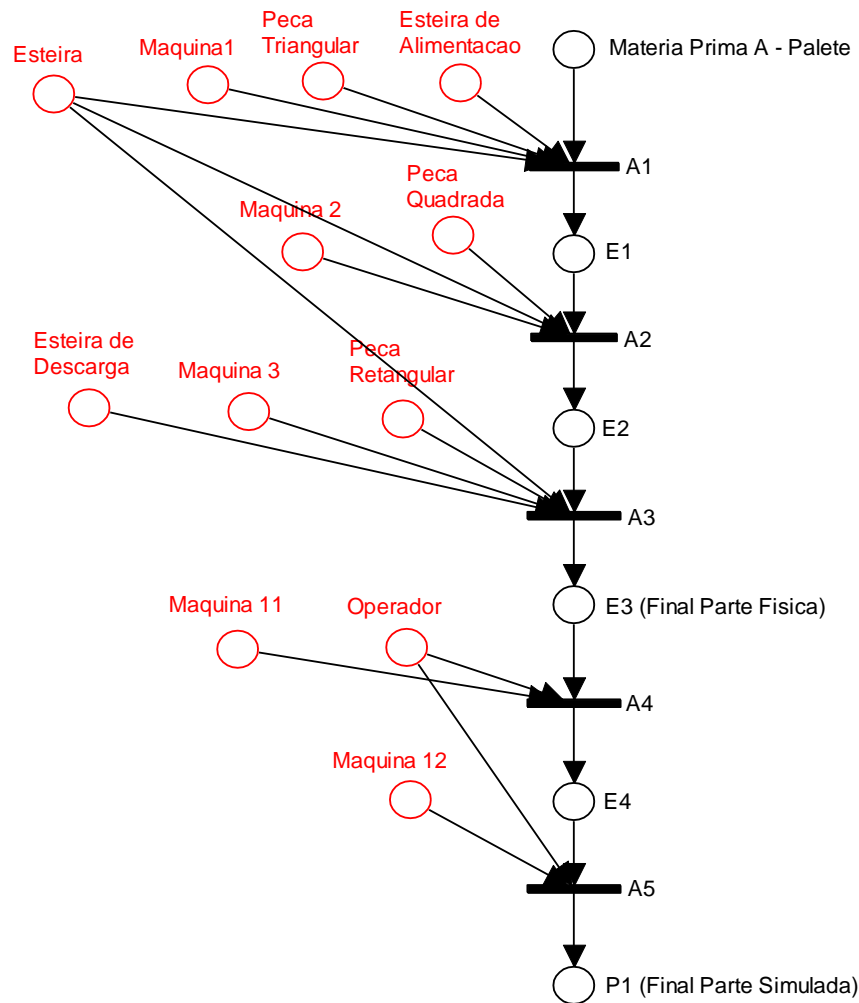


Figura 4.2 Representação do roteiro de fabricação do produto P1.

Assim como foi apresentada a aquisição e modelagem do roteiro de fabricação do produto P1, existem para todos os 18 produtos produzidos pela fábrica. Essas tabelas e modelagens estão apresentadas no apêndice A.

4.3 Sistema Computacional

Foi desenvolvido um sistema computacional com objetivo principal de servir como meio de validação do modelo proposto. Esse *software* consiste de uma interface de comunicação para o engenheiro e outra diferente para o operador, um módulo responsável pela programação reativa e uma base de dados, conforme ilustrado na Figura 4.3.

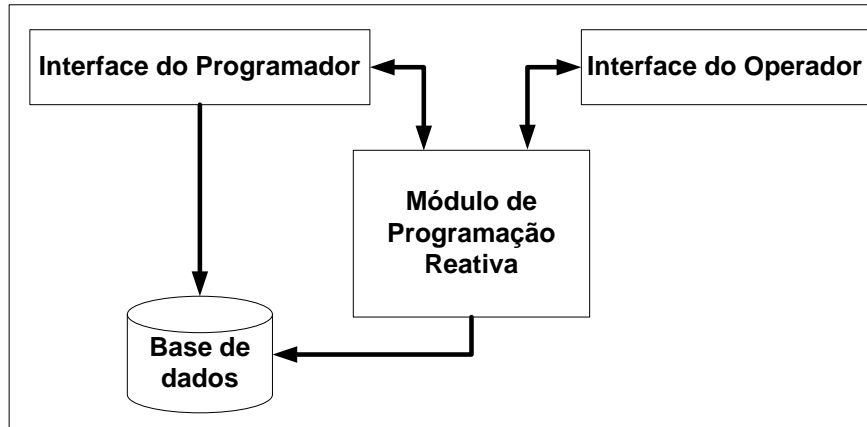


Figura 4.3 Arquitetura do Sistema Computacional.

A interface do operador é o meio de troca de informações do software com o usuário operador, na qual são informadas as ocorrências e alguns dados necessários para realizar a programação reativa, após esse processo é apresentada a nova programação para o operador.

A interface do programador, assim como a interface do operador, é um meio de comunicação com o *software*. Além disso, possui comunicação com a base de dados na qual é possível fazer inserção e manutenção dos dados existentes, como por exemplo, adicionar novos produtos, com seus respectivos roteiros de produção ou até mesmo incluir uma nova programação.

A base de dados é onde ficam armazenados todos os dados relativos às características da fábrica e dos produtos, como por exemplo, produtos existentes, recursos utilizados, lista de programação, entre outros.

O módulo de programação reativa é onde são realizados os cálculos para determinar a nova programação que será apresentada para o usuário. Esse módulo é dividido em módulos menores como apresentado na Figura 4.4.

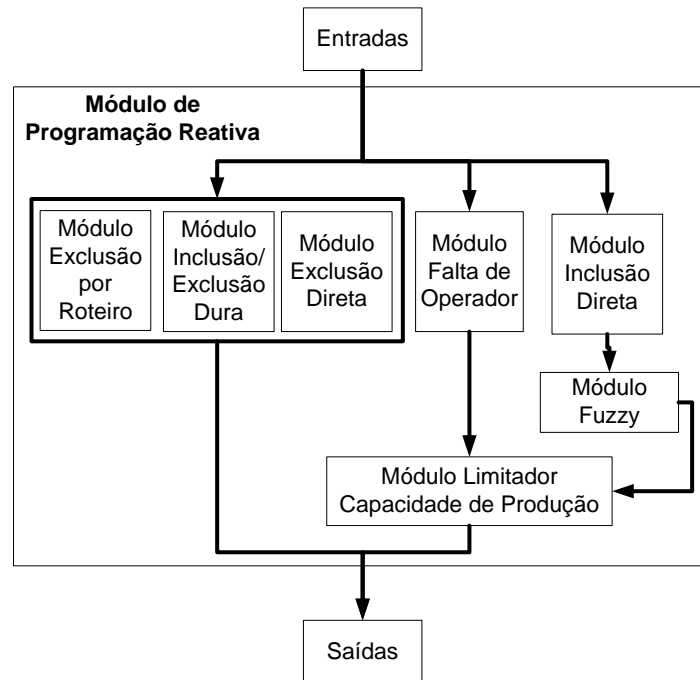


Figura 4.4 Arquitetura do módulo de programação reativa.

Os módulos de tratamento das ocorrências foram divididos em cinco submódulos: exclusão por roteiro, inclusão direta, exclusão direta, inclusão / exclusão dura e falta de operador. Possui também um submódulo *fuzzy* responsável por atribuir as novas prioridades e um limitador de capacidade de produção.

Esses módulos foram desenvolvidos utilizando a ferramenta MATLAB[®]. Essa ferramenta foi escolhida para apoiar a confiabilidade da modelagem *fuzzy* e ao mesmo tempo permitir uma fácil manipulação das matrizes de dados extraídas do banco de dados.

Para a modelagem do banco de dados utilizou-se a ferramenta ACCESS[®] pela fácil compreensão e por entender que ela atende as necessidades do sistema computacional. Já para a parte de interfaces, foi utilizada a ferramenta de desenvolvimento Visual Studio[®], pela facilidade da criação de interfaces bem como a fácil comunicação com o banco de dados e por se tratar de uma ferramenta bem completa que atenda a todas as necessidades levantadas.

A base de dados é o meio de comunicação entre a parte desenvolvida pelo Visual Studio[®] com a parte desenvolvida em MATLAB[®]. A única comunicação que há entre a parte do Visual Studio[®] com a parte do MATLAB[®] se faz no início do programa quando o usuário realizar a programação reativa da produção (Figura 4.5).

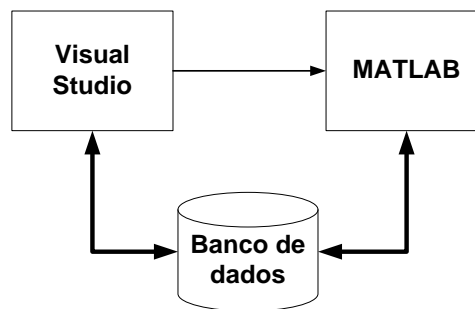


Figura 4.5 Comunicação entre as ferramentas de desenvolvimento.

A base de dados foi modelada contendo um total de nove tabelas relacionadas entre si para atender as necessidades funcionais do sistema conforme apresentada na Figura 4.6.

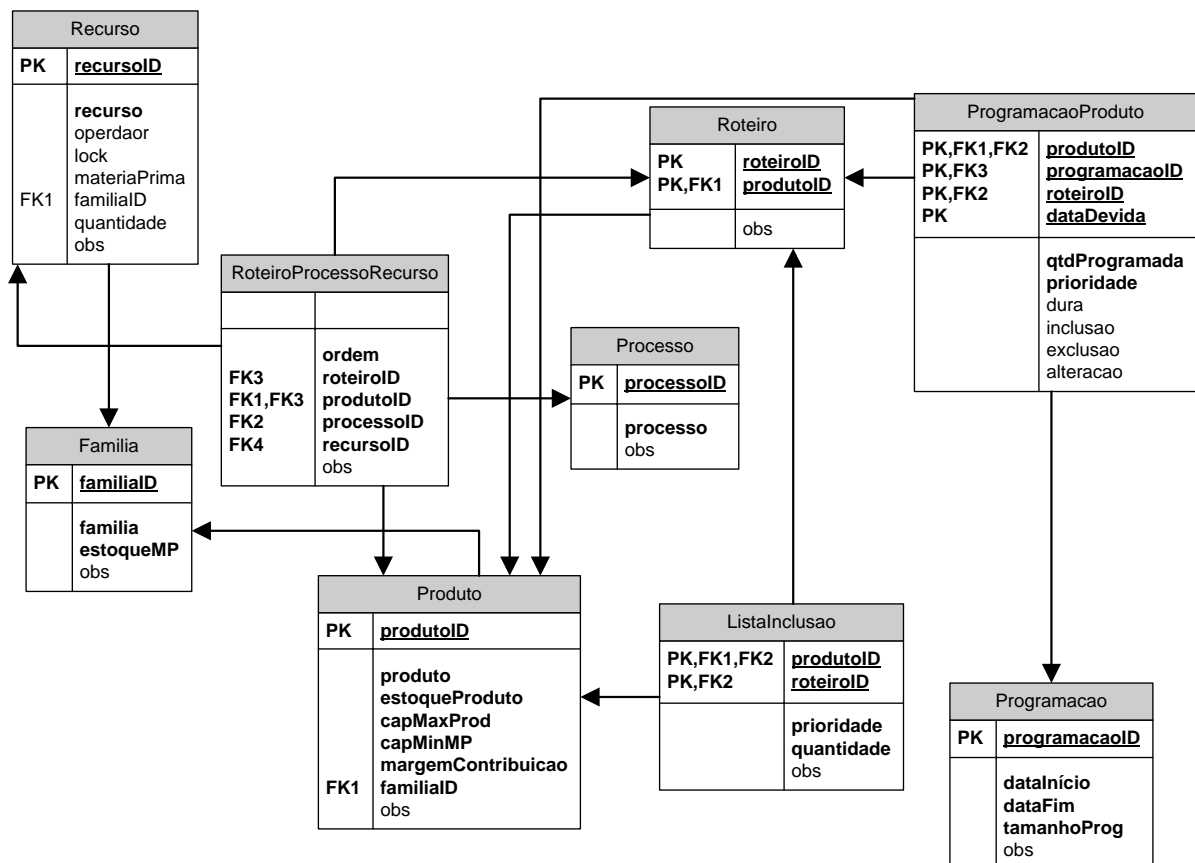


Figura 4.6 Modelo entidade relacionamento contemplando a base de dados desenvolvida.

Para armazenar informações referentes aos produtos levantados, foram criadas as tabelas “Produto”, “Processo”, “Recurso”, “Roteiro” e “Família”, e uma tabela, para relacionar essas informações, chamada de “RoteiroProcessoRecurso”.

Para o armazenamento da lista de produtos preferenciais pra a inserção foi criada a tabela “ListaInclusao” e para armazenar a programação, foram criadas duas tabelas,

uma contendo a identificação da programação, “Programacao”, e outra contendo os produtos contidos na programação, “ProgramacaoProduto”.

Para o uso do sistema computacional criado, se torna necessária a interação do usuário, que pode ser tanto o operador quanto o programador, sendo que cada um deles possuem uma interface de interação própria com o *software*.

Os papéis do programador além de ser um simples usuário do sistema, capaz de realizar a programação reativa da produção, também são responsáveis pela configuração do sistema. Essas configurações podem ser entre adicionar, remover e alterar: produtos, famílias, recursos, processos, roteiros de fabricação; incluir lista de produtos preferenciais; adicionar uma programação.

Essas informações são salvas direto na base de dados, sendo utilizadas para realizar a programação reativa (Figura 4.7).

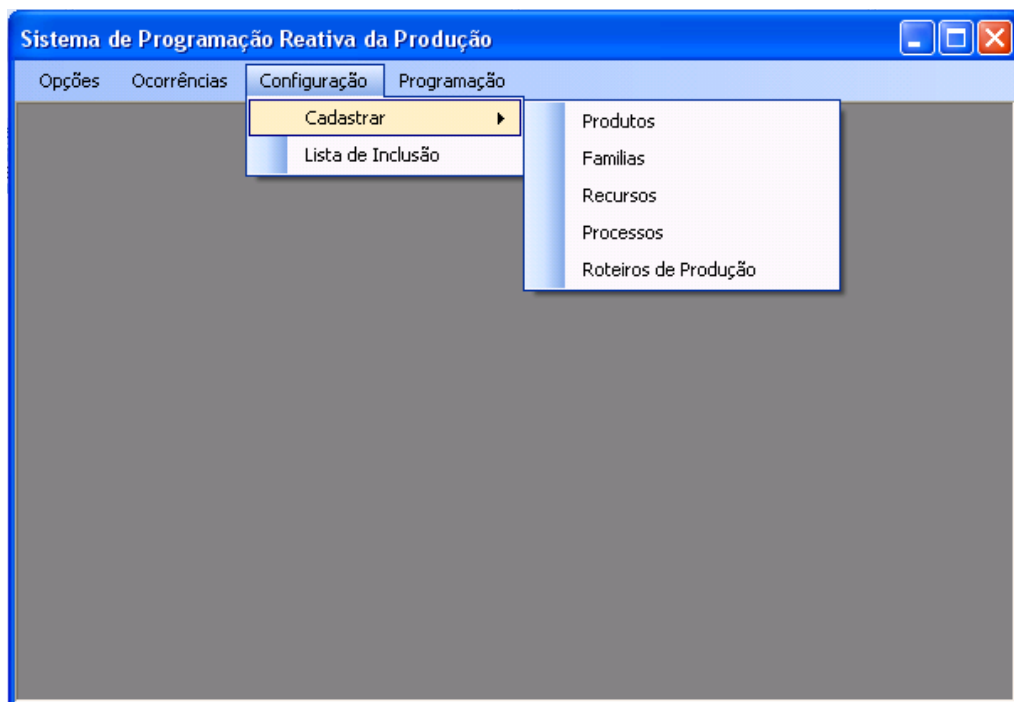


Figura 4.7 Interface de configuração do programador.

O outro papel do usuário programador é de realizar a programação reativa que assim como o usuário operador, existe uma interface específica para tal operação. Nessa interface os usuários, tanto o operador quanto o programador, indicam o evento ocorrido que causou a necessidade de realizar a programação reativa (Figura 4.8).

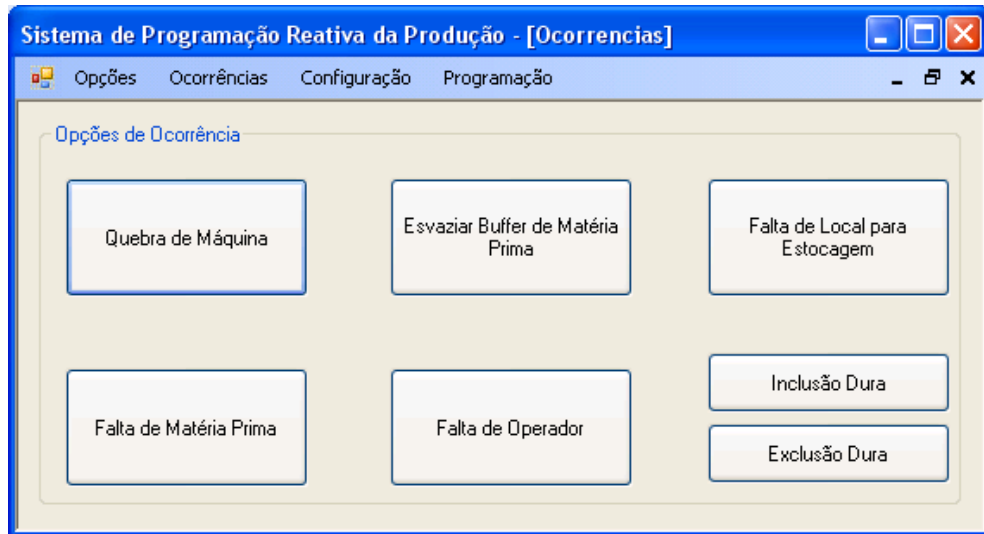


Figura 4.8 Interface do Operador e Programador com o sistema.

Há sete diferentes opções, sendo cada uma delas específicas para tratar cada ocorrência levantada na etapa de modelagem do conhecimento. Dentre as opções de ocorrência estão: “Quebra de máquina”, “Esvaziar *Buffer* de Matéria Prima”, “Falta de Local para Estocagem”, “Falta de Matéria Prima”, “Falta de Operador”, “Inclusão Dura” e “Exclusão Dura”.

Após a identificação da ocorrência, o sistema verifica o cenário atual por meio de perguntas pré-determinadas e analisando a base de dados. Com essas perguntas respondidas pelo próprio usuário, mais os dados presentes na base, o sistema computacional determina a lista da nova programação e a apresenta como saída (Figura 4.9).

Data de Início	Data de Fim	Tamanho	obs
25/3/2009 15:07	1/4/2009 15:07	10	*
*			

Produto	Roteiro	QTD	Prioridade	Obs
P4	4	10	8	*
P2	2	10	8	*
P12	12	10	5	*
P6	6	10	5	*
P1	1	10	5	*

Figura 4.9 Saída apresentada pelo sistema.

Essa saída é apenas uma sugestão para o usuário, sendo que ele ainda tem a opção de alterar essa sugestão realizando um novo cálculo por meio de alguma ocorrência ou

até mesmo acrescentar ou remover um produto da programação por meio da inclusão ou exclusão dura.

5 Validação

A validação desse trabalho foi dividida em duas etapas, a primeira trata a validação do sistema computacional em termos de implementação e da lógica estrutural dos módulos. A segunda parte é a validação do modelo como um todo, contemplando a lógica de tratamento das ocorrências levantadas por meio dos casos de uso.

Para a validação do sistema computacional, foi testado cada módulo de maneira individual para saber se cada um deles atinge o objetivo previsto. Após essa análise de maneira individual, foi testada a integração dos módulos até que o sistema computacional seja testado como um todo.

Essa integração dos módulos é determinada de acordo com a funcionalidade que essa integração trata. A Figura 5.1 representa por meio de mapas do conhecimento os módulos e suas estruturas para tratar as funcionalidades.

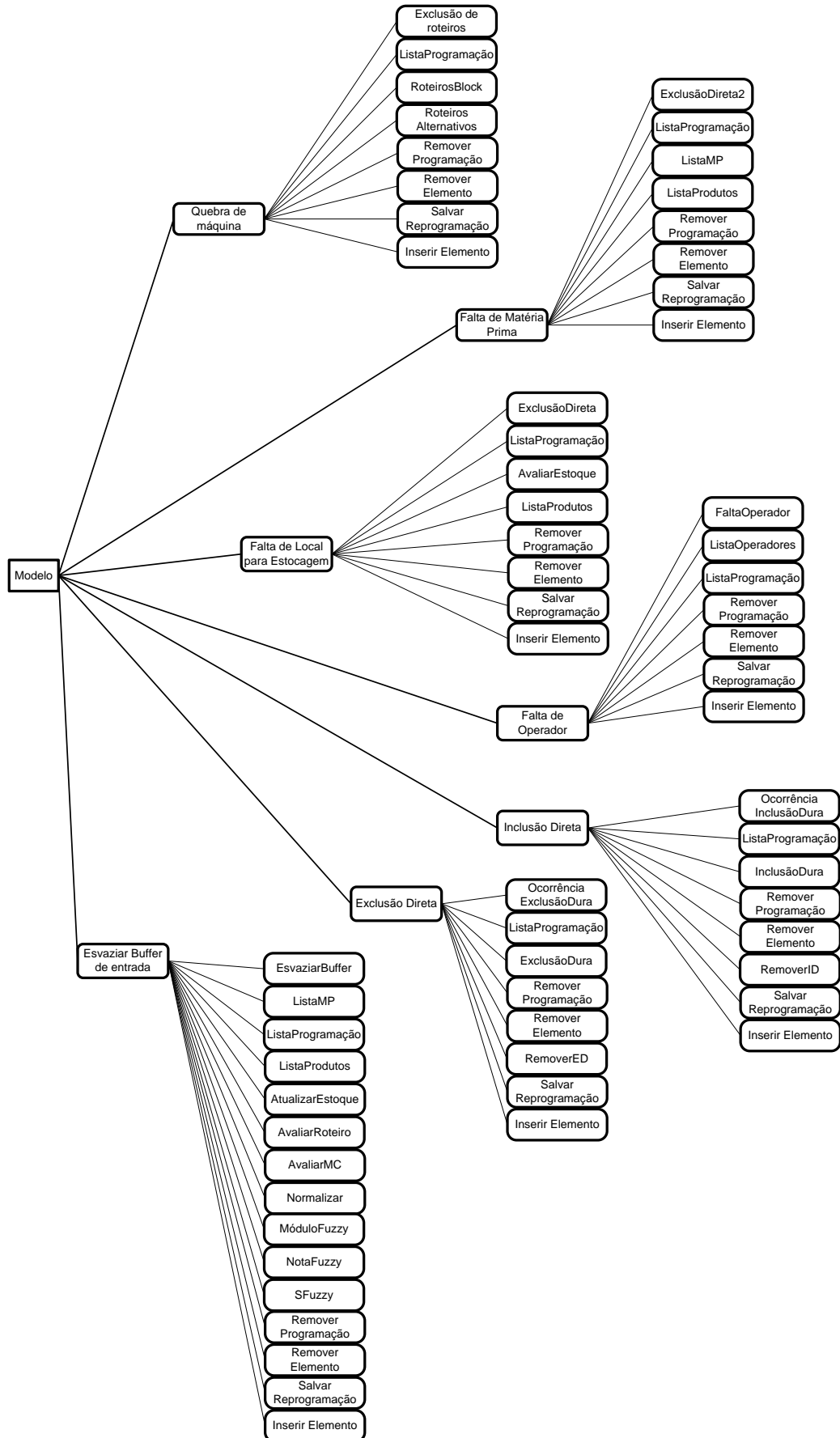


Figura 5.1 Mapa do conhecimento representando as funcionalidades e os módulos que as compõem.

5.1 Validação da Implementação e da Lógica Estrutural dos Módulos

Dentro da validação do sistema computacional, podemos dividir em duas partes, a primeira sendo a interface criada pela linguagem *Visual Basic .NET*; a segunda é a parte lógica do sistema que foi implementada utilizando a ferramenta MATLAB.

Para a validação dos módulos responsáveis pela parte lógica do sistema, foram utilizados testes unitários para verificar se cada módulo executa sua função de maneira correta. Para a realização desses testes, primeiramente foram determinados os dados de entrada e os dados de saída corretos para cada um dos módulos.

Esse mapeamento dos dados de entrada e de saída desejados possibilita a análise do funcionamento correto do módulo, avaliando se com os dados de entrada, os dados de saída obtidos são iguais aos desejado.

Para essa análise foi desenvolvido um módulo testador também utilizando a ferramenta MATLAB. Cada módulo criado possui um módulo testador correspondente e o mapeamento dos dados de entrada e dados desejados de saída.

Esse testador realiza uma chamada ao módulo a ser testado passando de maneira automática os dados de entrada já mapeados. Os dados de saída obtidos pelo módulo testado são comparados com a saída desejada gerando um relatório sobre a execução do teste.

Esse relatório apresenta as informações sobre os dados de entrada e de saída desejados já mapeados; os dados de saída obtidos; a quantidade de dados de entrada, de saída, e de saída desejados; a quantidade de acertos comparando os dados de saída obtidos com os desejados, conforme apresentado na Figura 5.2.

```

*****
* Testar Modulo: AvaliarRoteiros *
*****

*****
* Teste n° 1 *
*****

|Entrada| |saida| |saida desejada|
|-----| |-----| |-----|
|  1  | |  1  | |  1  |
|  2  | |  2  | |  2  |
|  3  | |  3  | |  3  |
|  4  | |  4  | |  4  |
|  5  | |  5  | |  5  |
|  6  | |  6  | |  6  |
|  7  | |  7  | |  7  |
|  8  | |  8  | |  8  |
|  9  | |  9  | |  9  |
| 10  | | 10  | | 10  |
| 11  | | 11  | | 11  |
| 12  | | 12  | | 12  |
| 13  | | 13  | | 13  |
| 14  | | 14  | | 14  |
| 15  | | 15  | | 15  |
| 16  | | 16  | | 16  |
| 17  | | 17  | | 17  |
| 18  | | 18  | | 18  |

N° de dados de entrada: 18
N° de dados de saida: 18
N° de dados de saida desejada: 18
Comparações certas: 18/18

```

Figura 5.2 Relatório sobre o teste do módulo AvaliarRoteiros.

Após testar todos os módulos de maneira individual, se torna necessário testar a integração entre eles. Essa integração dos módulos é realizada de acordo com as ocorrências levantadas anteriormente.

Cada ocorrência possui o seu conjunto de módulos para solucioná-las, o teste de integração é realizado para cada conjunto de módulos que representam uma ocorrência, totalizando seis grupos de módulos.

Cada grupo de módulos possui sua própria lógica e como consequência disso sua própria organização estrutural. Sendo sempre um módulo principal responsável por coordenar as chamadas aos outros módulos secundários, também há casos de que módulos secundários realizam chamadas a outros módulos, sendo estes chamados de módulos terciários. O módulo principal não faz nenhuma conexão com a base de dados, para isso utilizam-se os módulos secundários ou terciários.

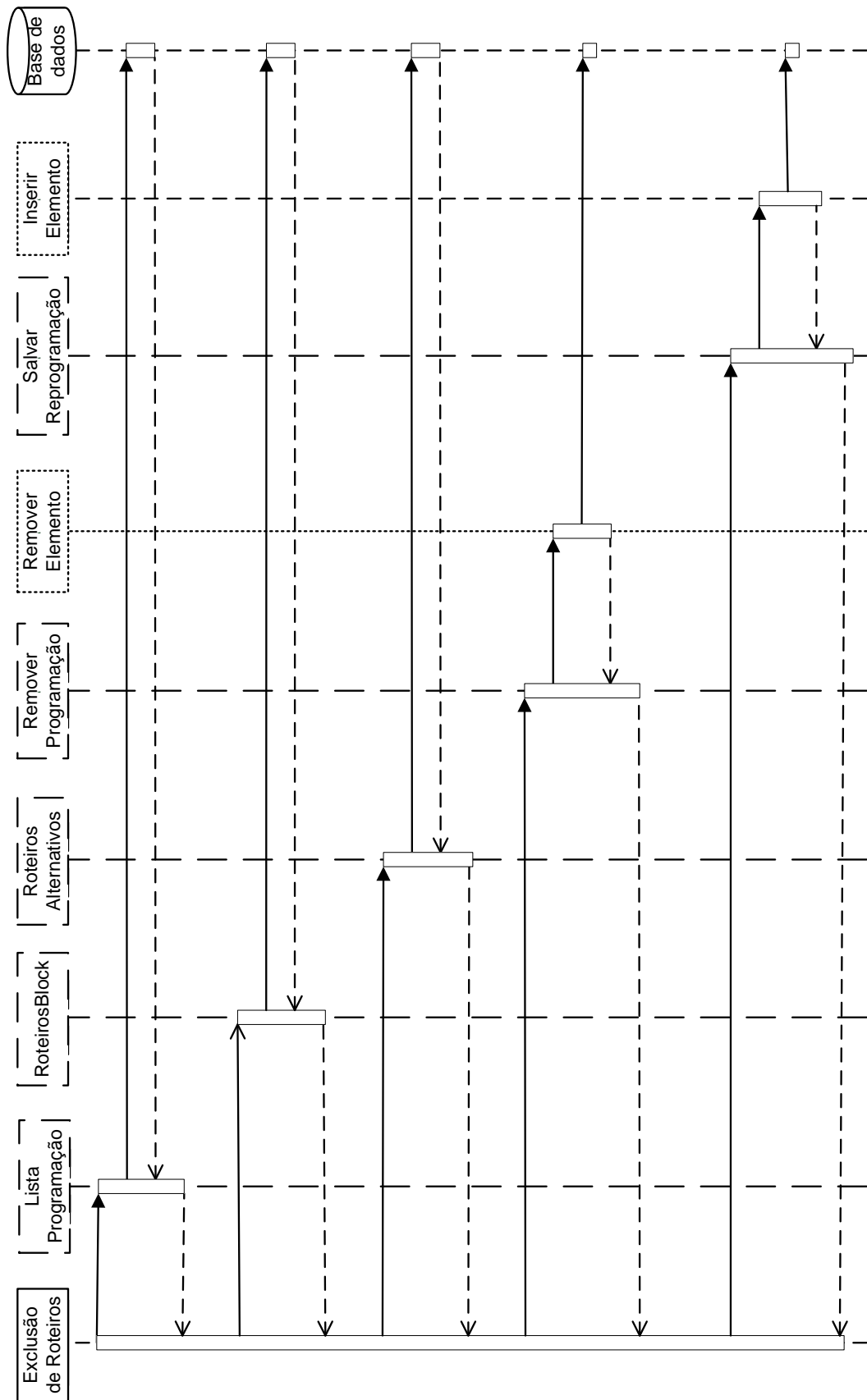


Figura 5.3 Diagrama de seqüência entre os módulos pertencentes à ocorrência quebra de máquina.

O diagrama de seqüência ilustrado na Figura 5.3 representa a integração dos módulos responsáveis por realizar a lógica de tratamento da ocorrência “quebra de máquina”. Sendo o módulo principal representado pelo bloco com linha normal, os com a linha tracejada os módulos secundários e os com a linha pontilhada os terciários.

Para ilustrar a integração entre os módulos de cada uma das ocorrências, foram gerados diagramas de seqüência para cada uma delas. As ocorrências podem eventualmente utilizar-se dos mesmos módulos, permitindo assim um reuso dos módulos para o tratamento de cada uma delas.

Para a validação da integração entre os módulos, foram mapeados os dados de entrada necessários e os dados desejados de saída para cada uma das ocorrências, possibilitando assim uma análise do funcionamento lógico da integração dos módulos.

Após esse mapeamento, foi desenvolvido um módulo testador para cada uma das ocorrências existentes, permitindo um teste automático da integração dos módulos e a criação de um relatório avaliando se as saídas obtidas estão de acordo ou não com as saídas desejadas já mapeadas (Figura 5.4).

```

*****
* Testar Funcionalidade: Falta de Materia Prima *
*****

*****
* Teste n° 1 *
*****

Dados de entrada:
*****

Esteque de entrada      Programação de entrada
-----
1  1  400                1  1  5  5
2  2  0                  2  2  5  8
3  3  0                  4  4  5  8
                              6  6  5  5
                              12 12 5  7

Dados de Saída:
*****

Saída Obtida           Saída Desejada
-----
1  1  5  5              1  1  5  5
2  2  5  8              2  2  5  8
4  4  5  8              4  4  5  8
6  6  5  5              6  6  5  5

=> Saída Desejada Alcançada

=====
Estatísticas
testes corretos: 1/1

```

Figura 5.4 Relatório sobre o teste de integração dos módulos pertencentes à ocorrência quebra de máquina.

5.2 Validação da lógica de tratamento das ocorrências

A partir de cinco ocorrências levantadas na parte de modelagem do conhecimento, foram gerados cinco casos de uso sendo um para cada ocorrência. Esses cinco casos de uso estão contemplando as ocorrências: “Quebra de máquina”, “Falta de matéria prima”, “Falta de local para estocagem”, “Esvaziar *buffer* de matéria prima” e “Falta de operador”.

Há também dois casos de uso levantados a partir de funcionalidades adicionadas ao modelo, sendo elas: “Inclusão dura” e “Exclusão dura”, garantindo assim robustez ao modelo, permitindo lidar com novos problemas que possam acontecer e não foram mapeados anteriormente.

Para cada caso de uso foi gerado um fluxograma para servir de guia nos testes do tratamento da ocorrência.

5.2.1 Teste de validação do tratamento da ocorrência: Quebra de Máquina

O primeiro teste de validação ao qual o modelo foi submetido utilizou o caso de uso referente ao tratamento da ocorrência “Quebra de máquina”. O teste foi realizado com a configuração apresentada na Tabela 5.1.

Tabela 5.1 Programação inicial para a validação do tratamento da ocorrência: Quebra de máquina.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P4	4	5	4
P6	116	5	3
P12	12	5	2

Para o teste foram escolhidos dois produtos que possuem roteiros alternativos, sendo eles: P6 que possui os roteiros 6 e 116; P12 que possui os roteiros 12 e 112. Cada um desses roteiros é distinto, sendo apresentados em detalhes os roteiros de todos os produtos (Tabela 5.2)

Tabela 5.2 Tabela de roteiros de fabricação dos produtos testados na validação do tratamento da ocorrência: Quebra de máquina.

Produto	Roteiro	Máquinas
P1	1	M1, M2, M3, M11, M12
P2	2	M1, M2, M3, M11, M13
P4	4	M1, M2, M3, M11, M12, M13
P6	6	M1, M2, M3, M12
	116	M1, M2, M3, M13
P12	12	M1, M2, M3, M22
	112	M1, M2, M3, M23

Após determinado o cenário a ser testado é apresentado na Figura 5.5 o fluxograma a ser seguido para a realização do teste.

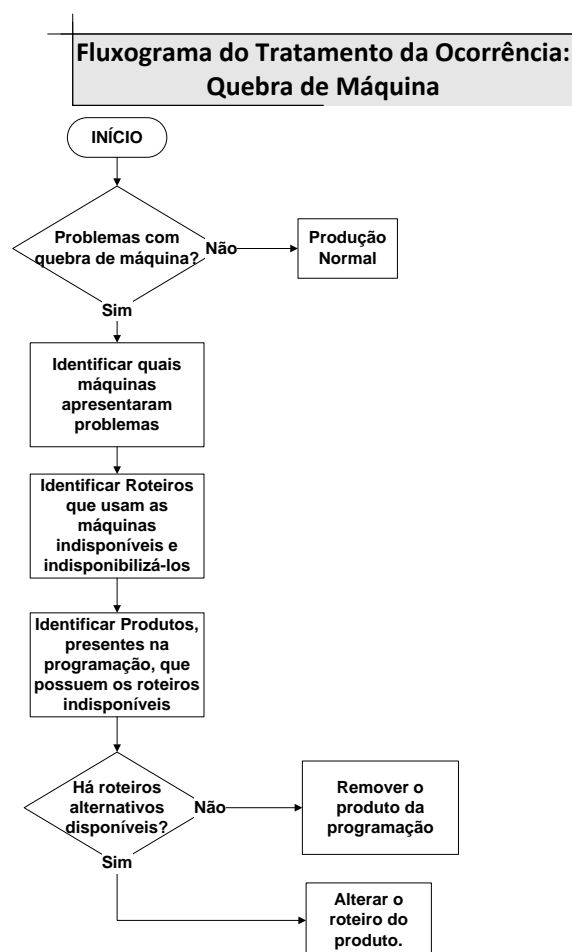


Figura 5.5 Fluxograma do tratamento da ocorrência: Quebra de máquina.

O primeiro passo é identificar quais máquinas que estão presentes na programação estão indisponíveis. Nesse caso determinou-se que as máquinas M13 e M22 estarão bloqueadas, refletindo no roteiro 2 do produto P2, roteiro 4 do produto P4, roteiro 116 do produto P6 e no roteiro 12 do produto P12.

A partir dessa informação, é possível avaliar os produtos presentes na programação e identificar que os produtos P2, P4, P6 e P12 foram afetados por conta dos roteiros que utilizam as máquinas bloqueadas.

Para os produtos identificados anteriormente, é necessário avaliar se há roteiros alternativos e no caso de ter, se eles estão disponíveis. Os produtos P2 e P4 não possuem roteiros alternativos; o produto P6 possui um roteiro alternativo que é o roteiro 16; o produto P12 também possui um roteiro alternativo que é o roteiro 112.

Depois de ter os possíveis roteiros alternativos devidamente identificados, é necessário avaliar se os mesmos encontram-se disponíveis. Para o roteiro 16 do produto P6, foi verificado o uso das máquinas: M1, M2, M3, M12; e para o roteiro 112 do produto P12, foi verificado o uso das máquinas: M1, M2, M3, M23.

Avaliou-se que nenhum dos possíveis roteiros alternativos apresentados utilizam as máquinas bloqueadas, possibilitando o uso desses para a produção dos produtos P6 e P12.

Após essas avaliações, é necessário alterar os roteiros 116 e 12 por 16 e 112 respectivamente, pertencentes aos produtos P6 e P12. No caso dos produtos P2 e P4 que não possui roteiros alternativos, serão removidos da programação.

No final do teste, obteve-se a programação reativa da produção conforme apresentado na Tabela 5.3

Tabela 5.3 Programação reativa da produção para a validação do tratamento da ocorrência: Quebra de máquina.

Programação Reativa da Produção			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P6	6	5	3
P12	112	5	2

Aplicando o mesmo cenário ao sistema computacional, deverá obter os mesmos resultados. Foi adicionada ao sistema computacional a mesma lista de programação, apresentada anteriormente, conforme apresentado na Figura 5.6.

Informações da Programação Vigente				
	Data de Inicio	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação					
	Produto	Roteiro	QTD	Prioridade	Obs
▶	P1	1	10	7	*
	P2	2	10	5	*
	P4	4	10	4	*
	P6	6	10	3	*
	P12	12	10	2	*

Figura 5.6 Programação inserida no sistema computacional.

Depois de inseridos os dados sobre a programação é necessário optar pela ocorrência desejada, nesse caso de teste é a ocorrência: “Quebra de Máquina” (Figura 5.7).

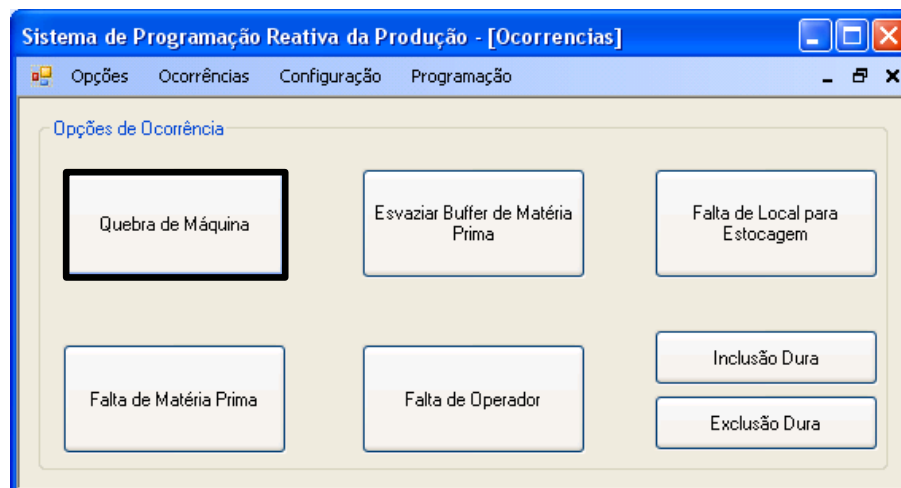


Figura 5.7 Escolha da opção "Quebra de Máquina".

Após a escolha da opção “Quebra de Máquina”, são listados todos os recursos que estão sendo utilizados na programação corrente. Neste ponto é necessário escolher quais os recursos que serão bloqueados, no caso desse teste, são os recursos M13 e M22 (Figura 5.8).

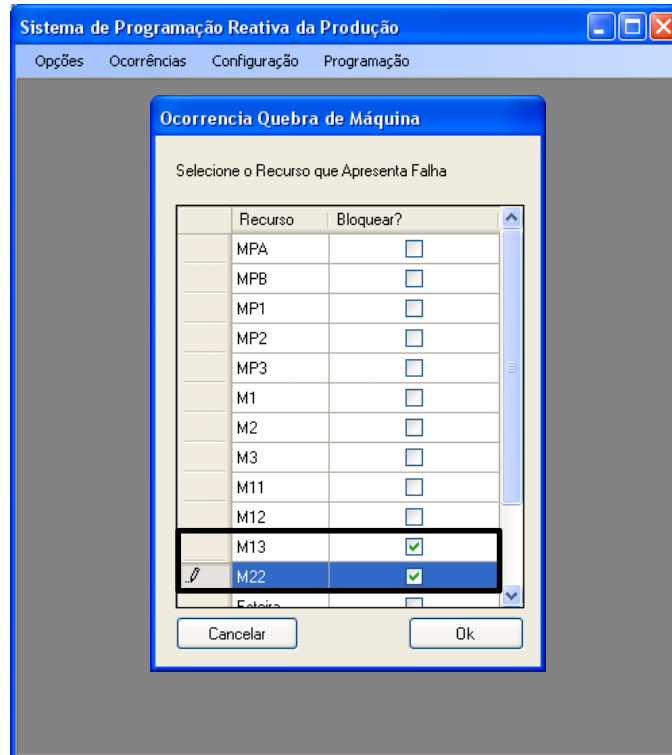


Figura 5.8 Escolha para bloquear os recursos M13 e M22.

Após serem bloqueados os recursos, a operação é executada determinando quais produtos serão excluídos e quais terão alteração de roteiros. A saída obtida é apresentada na Figura 5.9.

Informações da Programação Vigente

	Data de Inicio	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação

	Produto	Roteiro	QTD	Prioridade	Obs
▶	P1	1	10	7	*
	P6	6	10	3	*
	P12	112	10	2	*

Figura 5.9 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse teste do tratamento da ocorrência “Quebra de Máquina”.

5.2.2 Teste de validação do tratamento da ocorrência: Falta de Matéria Prima

O segundo teste de validação ao qual o modelo foi submetido utilizou o fluxograma referente ao tratamento ocorrência “Falta de matéria prima”. O teste foi realizado com a configuração apresentada na Tabela 5.4.

Tabela 5.4 Programação inicial para a validação do tratamento da ocorrência: Falta de matéria prima.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P7	7	5	4
P8	8	5	3
P13	13	5	2
P14	14	5	1

Para o teste foram escolhidos dois produtos de cada família, sendo que cada família utiliza uma matéria prima específica para a produção, sendo elas:

- Família A:
 - Produto – P1;
 - Produto – P2;
- Família B:
 - Produto – P7;
 - Produto – P8;
- Família C:
 - Produto – P13;
 - Produto – P14.

Para a produção dos produtos da família A, utiliza-se a matéria prima A (MPA); para os produtos da família B, a matéria prima B (MPB); e para os produtos da família C, a matéria prima C (MPC).

Após determinado esse cenário, no fluxograma da Figura 5.10 é apresentado o fluxograma do tratamento da ocorrência “Falta de matéria prima” que será seguido para a execução desse teste.

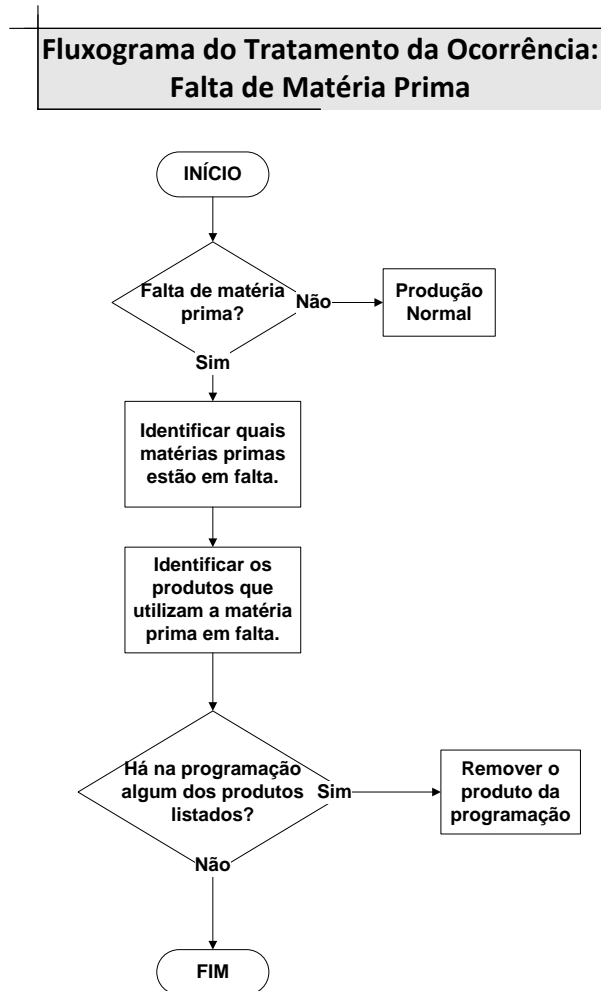


Figura 5.10 Fluxograma do tratamento da ocorrência: Falta de matéria prima.

O primeiro passo é identificar quais tipos de matéria prima estão em falta. Para esse teste determinou-se que a matéria prima que está em falta é a MPA. A partir dessa informação, é possível identificar todos os produtos que utilizam essa matéria prima em sua produção.

Com a informação dos produtos presentes na lista de programação, é possível identificar quais deles estão listados na relação de produtos que utilizam o tipo de matéria prima em falta, neste caso identificaram-se os produtos P1 e P2.

Depois de identificados os produtos, são removidos da programação atual, gerando a nova programação com os produtos que restaram na lista, conforme apresentado na Tabela 5.5.

Tabela 5.5 Programação reativa da produção para a validação do tratamento da ocorrência: Falta de matéria prima.

Programação Reativa da Produção			
Produto	Roteiro	Quantidade Programada	Prioridade
P7	7	5	4
P8	8	5	3
P13	13	5	2
P14	14	5	1

Configurando o sistema computacional com o mesmo cenário estabelecido no caso de teste, devem-se obter os mesmos resultados validando assim o modelo para esse fluxograma.

Foi adicionada a mesma lista de programação inicial definida no caso de teste, conforme apresentado na Figura 5.11.

Sistema de Programação Reativa da Produção - [Programação da Produção]				
Opções Ocorrências Configuração Programação				
Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				
Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P2	2	5	5
	P7	7	5	4
	P8	8	5	3
	P13	13	5	2
	P14	14	5	1

Figura 5.11 Programação inicial inserida no sistema computacional.

Depois de inseridos os dados sobre a programação da produção é necessário optar pela ocorrência desejada, nesse caso de teste é a ocorrência: “Falta de Matéria Prima” (Figura 5.12).

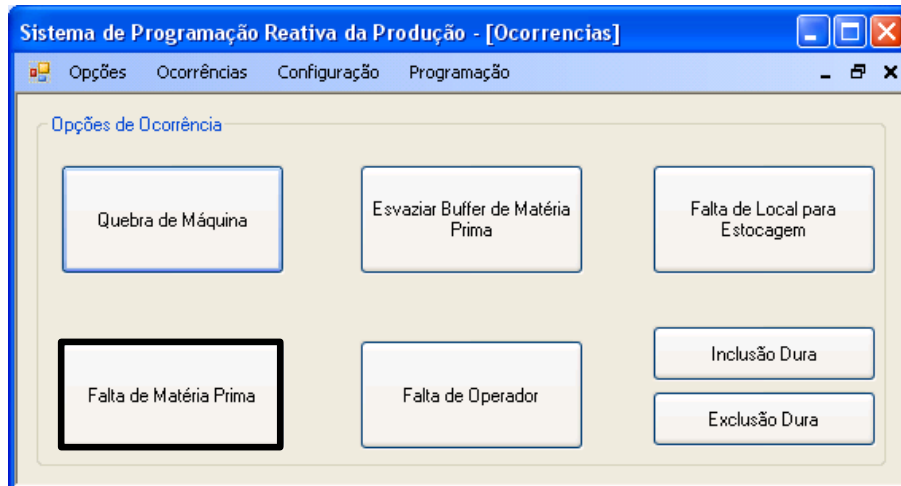


Figura 5.12 Escolha da opção "Falta de Matéria Prima".

Após a escolha da opção “Falta de Matéria Prima”, são listados todos os tipos de matéria prima que estão sendo utilizados pela programação corrente. Neste ponto é necessário escolher quais os tipos estão em falta, nesse caso optou-se somente pela matéria prima do tipo A (MPA) (Figura 5.13).

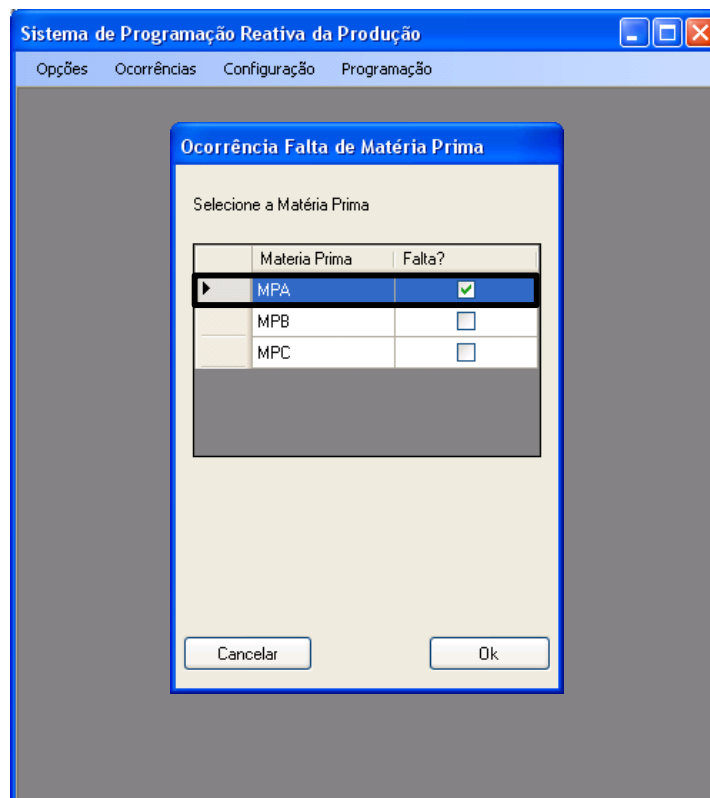


Figura 5.13 Escolha para indicar a falta da matéria prima MPA.

Depois de selecionado o tipo de matéria prima que está em falta, o sistema realiza o devido tratamento e apresenta a programação reativa da produção gerada, conforme apresentado na Figura 5.14.

Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15.07	1/4/2009 15.07	10	*
*				

Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P7	7	5	4
	P8	8	5	3
	P13	13	5	2
	P14	14	5	1

Figura 5.14 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse teste do tratamento da ocorrência “Falta de matéria Prima”.

5.2.3 Teste de validação do tratamento da ocorrência: Falta de Local para Estocagem

O terceiro teste de validação ao qual o modelo foi submetido utilizou o fluxograma referente à ocorrência “Falta de local para estocagem”. O teste foi realizado com a configuração apresentada na Tabela 5.6.

Tabela 5.6 Programação inicial para a validação do tratamento da ocorrência: Falta de local para estocagem.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P3	3	15	4
P4	4	5	3

Adotando que *a priori* que todos os produtos relacionados na lista de programação inicial tenham um espaço total para a estocagem de dez produtos e que em todos já tenham cinco produtos estocados, restando apenas espaço para armazenar mais cinco produtos de cada tipo.

Após determinado esse cenário, no fluxograma da Figura 5.15 é apresentado o caso de uso do tratamento da ocorrência “Falta de matéria prima” que será seguido para a execução desse teste.

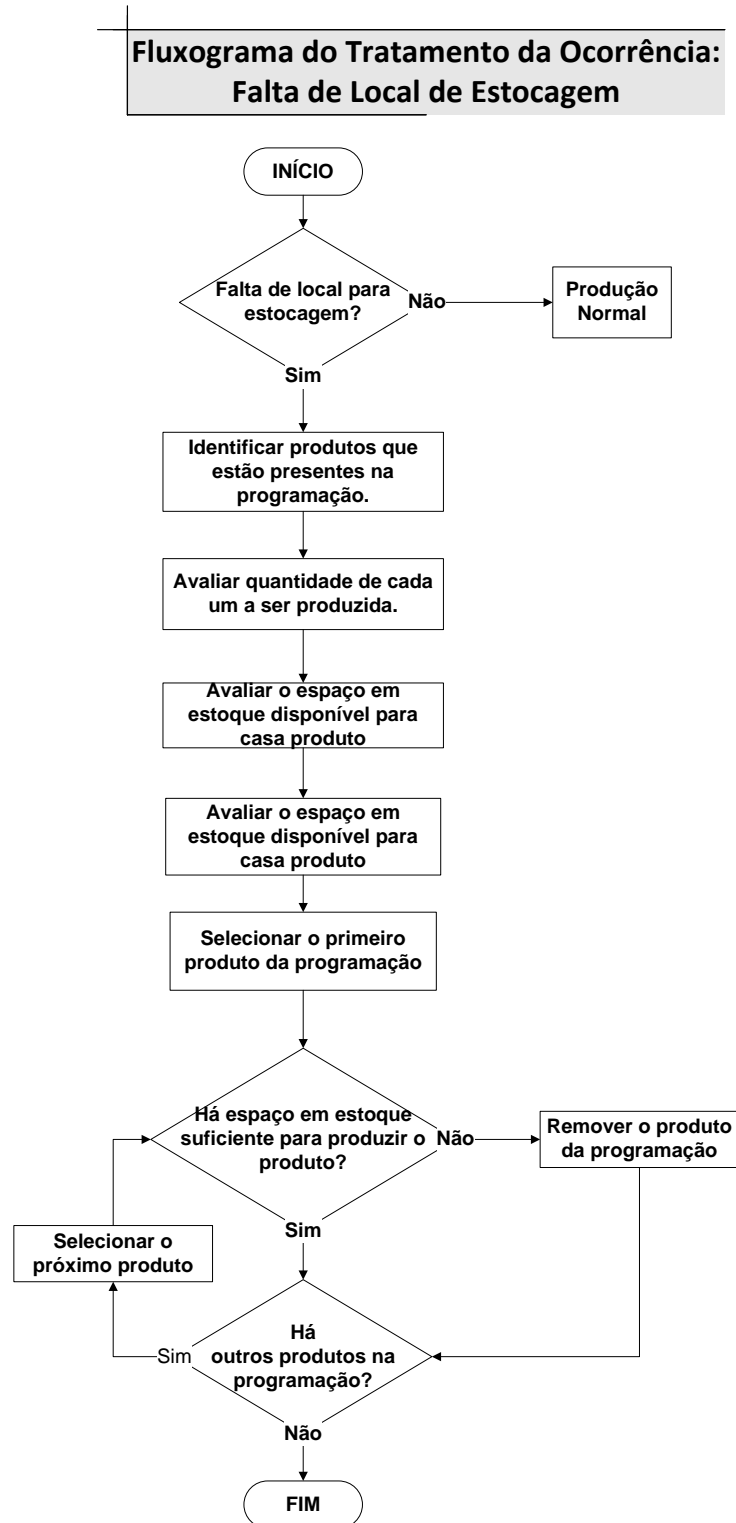


Figura 5.15 Fluxograma do tratamento da ocorrência: Falta de estocagem.

O primeiro passo é listar todos os produtos presentes na programação, nesse caso temos os seguintes produtos: P1, P2, P3 e P4. Depois de identificados, é necessário avaliar a quantidade a ser produzida de cada um deles, nesse caso temos para os produtos P1, P2 e P4, a quantidade de cinco produtos a serem produzidos; e para o produto P3 a quantidade de 15 produtos.

A seguir é necessário avaliar o espaço em estoque de cada produto, como adotado anteriormente, cada produto possui o espaço em estoque para o armazenamento de mais cinco produtos.

Após essas avaliações são identificados os produtos que estão presentes na lista de programação e que não possuem local para estocagem, e conseqüentemente retirado da lista de programação. Nesse caso foi identificado e retirado da programação o produto P3, conforme apresentando na Tabela 5.7.

Tabela 5.7 Programação reativa da produção para a validação do tratamento da ocorrência: Falta de local para estoque.

Programação Reativa da Produção			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P4	4	5	3

Configurando o sistema computacional com o mesmo cenário estabelecido no caso de teste, devem-se obter os mesmos resultados validando assim o modelo para esse fluxograma.

Foi adicionada a mesma lista de programação inicial definida no caso de teste, conforme apresentado na Figura 5.16.

Sistema de Programação Reativa da Produção - [Programação da Produção]				
Opções Ocorrências Configuração Programação				
Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				
Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P2	2	5	5
	P3	3	15	4
	P4	4	5	3

Figura 5.16 Programação inicial inserida no sistema computacional.

Depois de inseridos os dados iniciais, é necessário optar pela ocorrência desejada, nesse caso de teste é a ocorrência: “Falta de Local para Estocagem” (Figura 5.17).



Figura 5.17 Escolha da opção "Falta de Local para Estocagem".

Após a escolha da opção “Falta de Local para Estocagem”, são listados todos os produtos que estão sendo utilizados pela programação corrente. Neste ponto é necessário escolher qual o produto que está sem espaço em estoque, nesse caso optou-se somente pelo produto P3 (Figura 5.18).

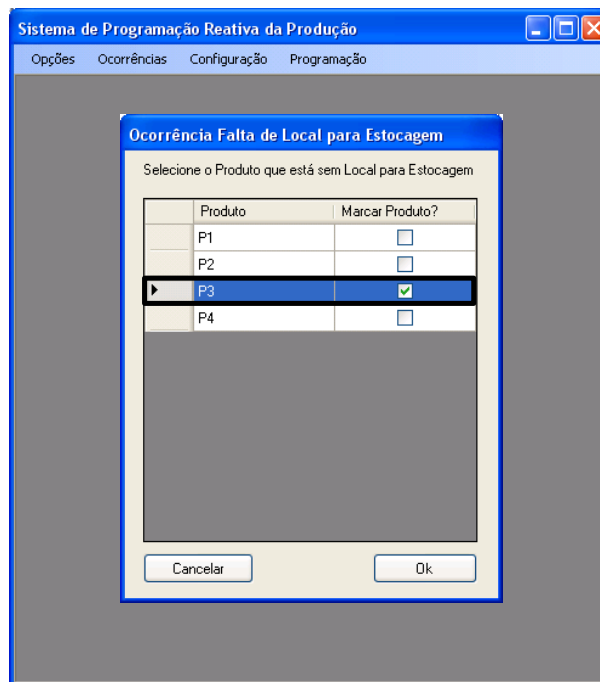


Figura 5.18 Escolha do produto que está sem local para estocagem.

Depois de selecionado o produto que está sem local para estocagem, o sistema realiza o devido tratamento e apresenta a programação reativa da produção gerada, conforme apresentado na Figura 5.19.

Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P2	2	5	5
	P4	4	5	3

Figura 5.19 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse teste do tratamento da ocorrência “Falta de local para estocagem”.

5.2.4 Teste de validação do tratamento da ocorrência: Falta de Operador

O quarto teste de validação ao qual o modelo foi submetido utilizou o fluxograma referente à ocorrência “Falta de operador”. O teste foi realizado com a configuração apresentada na Tabela 5.8.

Tabela 5.8 Programação inicial para a validação do tratamento da ocorrência: Falta de operador.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P3	3	15	4
P4	4	5	3
P5	5	5	3

Adotando que *a priori* todos os produtos necessitam para a sua produção de um único operador, e que tenha sempre cinco operadores por turno, sendo que esta programação deverá ser cumprida em um único turno.

Após determinado esse cenário, no fluxograma da Figura 5.20 é apresentado o caso de uso do tratamento da ocorrência “Falta de operador” que será seguido para a execução desse teste.

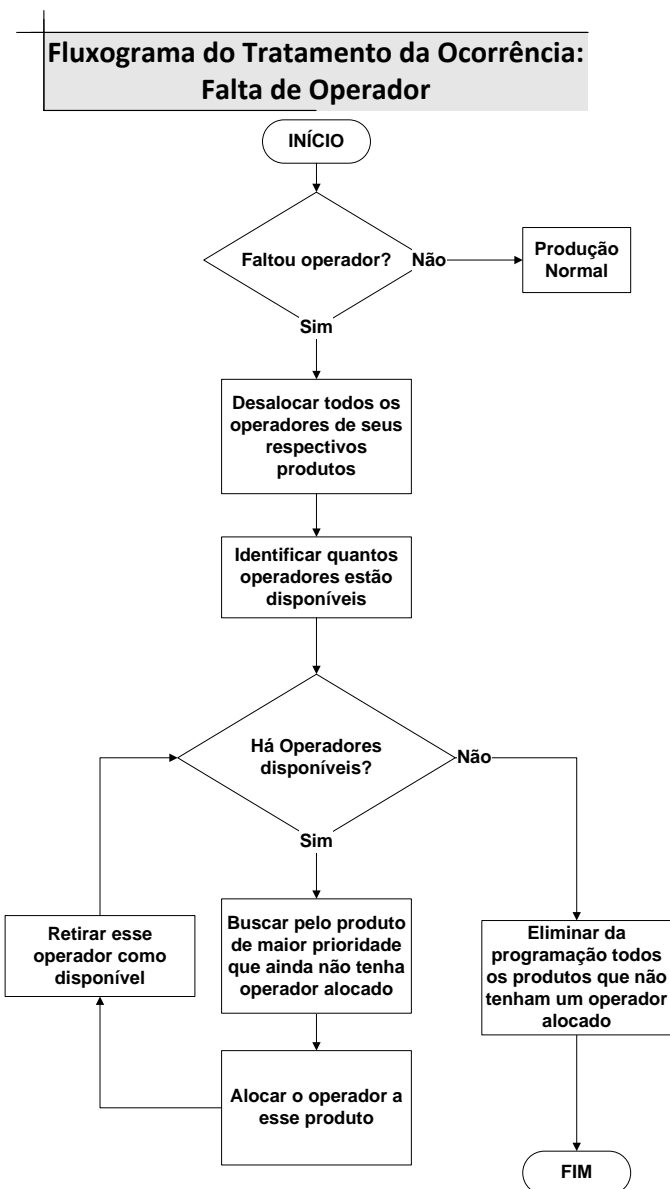


Figura 5.20 Fluxograma do tratamento da ocorrência: Falta de operador.

O primeiro passo é definir todos os operadores, que haviam sido alocados aos respectivos produtos planejados na programação inicial, como disponíveis. Depois é necessário verificar a quantidade de operadores. Nesse caso suponhamos que foi planejado ter cinco operadores disponíveis, porém houve uma ausência, ficando assim com apenas quatro operadores disponíveis.

O próximo passo é alocar cada operador disponível para os produtos de maior prioridade presentes na lista de programação, sendo que esta lista de programação inicial já está ordenada pela maior prioridade, sendo nesse caso os produtos: P1, com a prioridade 7; P2, com a prioridade 5; P3, com a prioridade 4; P4, com a prioridade 3; e P5 com a prioridade 3.

Após alocar todos os operadores aos produtos de maior prioridade, é necessário remover da programação os produtos que ficaram sem operador, nesse caso foi apenas o produto P5 e a programação reativa ficou conforme a Tabela 5.9.

Tabela 5.9 Programação reativa da produção para a validação do tratamento da ocorrência: Falta de Operador.

Programação Reativa da Produção			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P3	3	15	4
P4	4	5	3

Configurando o sistema computacional com o mesmo cenário estabelecido no caso de teste, devem-se obter os mesmos resultados validando assim o modelo para esse fluxograma.

Foi adicionada a mesma lista de programação inicial definida no caso de teste, conforme apresentado na Figura 5.21.

Sistema de Programação Reativa da Produção - [Programação da Produção]				
Opções Ocorrências Configuração Programação				
Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				
Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P2	2	5	5
	P3	3	5	4
	P5	5	5	3
	P4	4	5	3

Figura 5.21 Programação inicial inserida no sistema computacional.

Depois de inseridos os dados iniciais, é necessário optar pela ocorrência desejada, nesse caso de teste é a ocorrência: “Falta de Operador” (Figura 5.22).

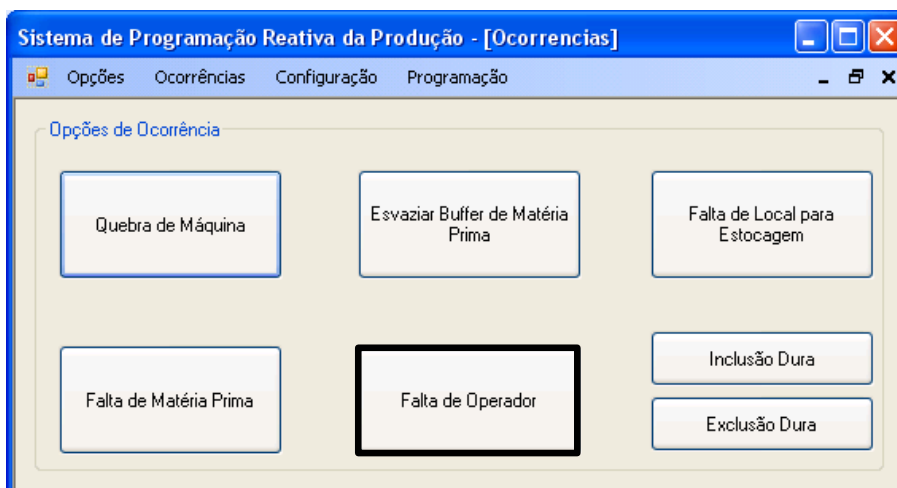


Figura 5.22 Escolha da opção "Falta de Operador".

Após a escolha da opção “Falta de Operador”, é necessário informar ao sistema quantos operadores estão disponíveis, nesse caso optou-se por quatro operadores disponíveis (Figura 5.23).

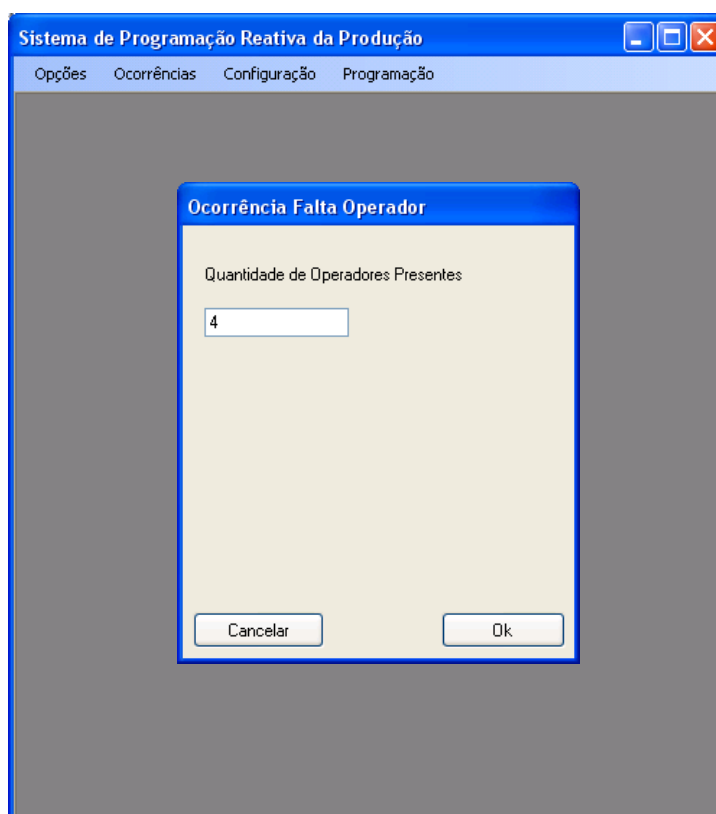


Figura 5.23 Escolhas de quantos operadores estão disponíveis.

Depois de inserido a quantidade de operadores disponível, o sistema realiza o devido tratamento e apresenta a programação reativa da produção gerada, conforme apresentado na Figura 5.24.

Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P2	2	5	5
	P3	3	5	4
	P4	4	5	3

Figura 5.24 Lista da Programação Reativa da Produção obtida após o tratamento da ocorrência.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse teste do tratamento da ocorrência “Falta de Operador”.

5.2.5 Teste de validação do tratamento da ocorrência: Esvaziar Matéria Prima do Buffer

O quinto teste de validação ao qual o modelo foi submetido utilizou o fluxograma referente à ocorrência “Esvaziar *Buffer* de Matéria Prima”. O teste foi realizado com a configuração apresentada na Tabela 5.10.

Tabela 5.10 Programação inicial para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P4	4	5	4
P6	6	5	3
P12	12	5	5

É necessário também estabelecer a quantidade de matéria prima consumida para produzir uma unidade do produto, para esse teste foi definido o seguinte:

- Quantidade matéria prima consumida para produzir uma unidade de produto:
 - Família A:
 - P1, P2, P3, P4, P5, P6 – 10 unidades;
 - Família B:

- P7, P8, P9, P10, P11, P12 – 20 unidades;
- Família C:
 - P13, P14, P15, P16, P17, P18 – 30 unidades.

Também foi estabelecida a quantidade de matéria prima no *buffer* de entrada, da forma apresentada na Tabela 5.11.

Tabela 5.11 Quantidade e tipo de matéria prima presente no *buffer* de entrada

Buffer de entrada		
MPA	MPB	MPC
400	100	0

Também é necessário saber a quantidade de espaço em estoque para armazenar os produtos produzidos, pois pode haver o caso de produzir mais produtos além dos que estão relacionados na programação inicial. Para esse caso foi considerado que todos os produtos possuem o espaço para armazenar mais cinco unidades de produtos, exceto o produto P3 que apresenta o estoque cheio e o produto P5 que possui o estoque vazio.

Outro ponto a ser definido é a lista de inclusão dos produtos preferenciais, essa lista indica produtos que o programador deseja produzir caso haja a necessidade. Essa lista foi definida da forma apresentada na Tabela 5.12.

Tabela 5.12 Lista de produtos Preferenciais para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Lista de produtos preferenciais			
Produto	Roteiro	Quantidade Programada	Prioridade
P2	2	5	7
P5	5	5	10
P15	15	5	9
P17	17	5	6

Definindo a margem de contribuição de cada produto, pois poderá vir a ser um critério de escolha de qual produto será produzido, temos o seguinte (Tabela 5.13):

Tabela 5.13 Tabela de Margem de contribuição dos produtos.

Família	Produto	Margem de contribuição
A	P1	R\$ 1,00
	P2	R\$ 2,00
	P3	R\$ 3,00
	P4	R\$ 4,00
	P5	R\$ 5,00
	P6	R\$ 6,00
	P7	R\$ 7,00
B	P8	R\$ 8,00
	P9	R\$ 9,00
	P10	R\$ 10,00
	P11	R\$ 11,00
	P12	R\$ 12,00
C	P13	R\$ 13,00
	P14	R\$ 14,00
	P15	R\$ 15,00
	P16	R\$ 16,00
	P17	R\$ 17,00
	P18	R\$ 18,00

Após determinado esse cenário, é apresentado no fluxograma da Figura 5.25 o caso de uso do tratamento da ocorrência “Esvaziar *Buffer* de Matéria Prima” que será seguido para a execução desse teste.

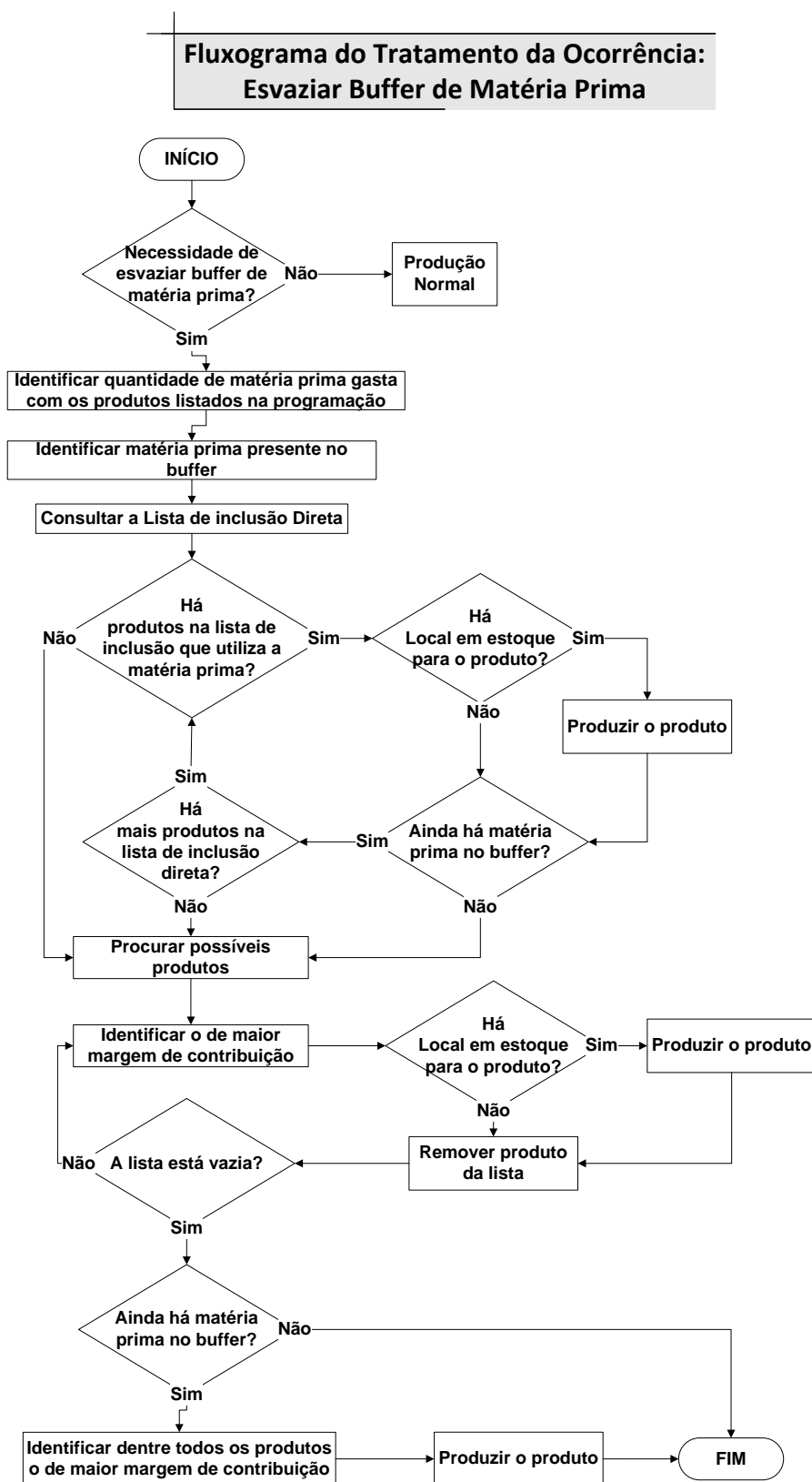


Figura 5.25 Fluxograma do tratamento da ocorrência "Esvaziar *Buffer* de entrada".

O primeiro passo é definir a quantidade de matéria prima gasta para produzir os produtos presentes na programação. Para tal atividade é necessário verificar a quantidade

que será produzida de cada produto e quanto de matéria prima é consumido para produzir um produto.

Para esse caso, temos presentes na programação os produtos: P1, P2, P4, P6, P12, sendo produzidas cinco unidades de cada um deles. Para os produtos P1, P2, P4 e P6 que pertencem à família A, são gastos 10 unidades de matéria prima para produzir uma unidade de produto. Para produzir cinco unidades de cada um dos quatro produtos serão necessárias 200 unidades de matéria prima do tipo A (MPA).

Para o produto P12, que pertence à família B, são gastos 20 unidades de matéria prima para produzir uma unidade de produto. Nesse caso, para produzir cinco unidades são necessárias 100 unidades de matéria prima do tipo B (MPB).

Subtraindo as matérias primas gastas com a produção dos produtos presentes na lista de programação, o *buffer* de entrada que continha a matéria prima MPB está vazio, pois só tinha 100 unidades. O *buffer* que continha a MPA, ainda apresenta 200 unidades de matéria prima, pois só foram gastas 200 sendo que inicialmente continham 400 unidades.

Com a produção desses produtos, o estado do estoque foi alterado, sendo que os produtos P1, P2, P3, P4, P6 e P12 apresentam o estoque cheio e os demais ainda possuem espaço para armazenar mais cinco unidades e o produto P5 continua com o estoque vazio.

Para esvaziar o *buffer* de entrada ainda é necessário consumir 200 unidades de matéria prima MPA, para isso primeiramente é necessário verificar quais produtos estão na lista produtos preferenciais, e se tem algum que utilize a matéria prima MPA em sua produção.

A lista levantada contém os produtos P2, P5, P15 e P17, sendo que apenas os produtos P2 e P5 utilizam a matéria prima MPA em sua produção. Depois de elegidos os possíveis produtos a serem produzidos, é necessário avaliar o espaço em estoque de cada um deles. Nesse caso, o produto P2 não possui local de estocagem e o produto P5 possui o estoque vazio.

Tendo em vista o espaço em estoque dos produtos candidatos, apenas o produto P5 poderá ser produzido. Produzindo as cinco unidades descritas na lista de produtos preferenciais, e levando em conta que gaste 10 unidades de MPA para produzir um produto P5, são gastas mais 50 unidades de MPA, restando ainda no *buffer* mais 150 unidades.

Nesse ponto o produto P5 é adicionado à lista de programação, deixando-a na configuração apresentada na Tabela 5.14.

Tabela 5.14 Lista de programação intermediária para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Programação intermediária			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P4	4	5	4
P6	6	5	3
P12	12	5	5
P5	5	5	10

Ainda há 150 unidades de matéria prima MPA no *buffer* de entrada, e não há mais nenhum produto da lista de produtos preferenciais que possam ser produzidos, nesse caso é necessário identificar o produto que possui a maior margem de contribuição e que possa ser produzido com a matéria prima presente no *buffer*.

Para esse caso, o produto que apresenta a maior margem de contribuição e é produzido por essa matéria prima é o produto P6, porém o estoque desse produto está cheio, impossibilitando a sua produção nessa etapa.

O próximo produto com a maior margem de contribuição e que utilize a MPA é o P5. Avaliando o estoque do produto, observa-se que é possível produzir cinco unidades de P5, levando em conta que cada unidade necessite de 10 MPA, para produzir os cinco são consumidos 50 MPA.

Subtraindo essa quantidade de matéria prima utilizada do total presente no *buffer* de entrada, ainda restam 100 unidades de matéria prima a ser consumida.

O estado do estoque dos produtos fica da seguinte maneira: todos os produtos da “família A” apresentam estoque cheio; os produtos da família B apresentam estoque de cinco unidades, exceto o produto P12 que apresenta estoque cheio; e os produtos da família C que apresentam espaço para armazenar mais cinco produtos.

A lista de programação é alterada novamente acrescentando as cinco unidades do produto P5 a serem produzidas, ficando da conforme apresentado na Tabela 5.15.

Tabela 5.15 Lista de programação intermediária para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Programação intermediária			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P4	4	5	4
P6	6	5	3
P12	12	5	5
P5	5	10	10

Como ainda há matéria prima no *buffer* e não tem mais produtos a ser produzido com espaço em estoque, o próximo passo é verificar qual dos produtos possui a maior margem de contribuição e produzi-lo até esgotar a matéria prima presente no *buffer*.

Nesse caso será produzido o produto P6, pois é o de maior margem de contribuição sendo que para esgotar as 100 unidades de MPA, é necessário produzir 10 unidades do produto, gerando a lista de programação apresentada na Tabela 5.16.

Tabela 5.16 Lista de programação intermediária para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Programação intermediária			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P4	4	5	4
P6	6	15	3
P12	12	5	5
P5	5	10	10

O próximo passo é submeter essa nova lista de programação gerada ao módulo *fuzzy*, que irá reclassificá-los de acordo com sua prioridade, quantidade em estoque, data devida e margem de contribuição, atribuindo uma nova prioridade para cada um dos produtos.

Nesse caso, as novas prioridades ficaram conforme a Tabela 5.17.

Tabela 5.17 Lista de programação intermediária após cálculo *fuzzy* para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Programação intermediária			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	0,55
P2	2	5	0,55
P4	4	5	1,11
P6	6	15	2,22
P12	12	5	3,33
P5	5	10	2,77

Após a atribuição das novas prioridades em função das variáveis estabelecidas, a lista ainda passa por um módulo de limitação da produção em função da sua capacidade. Esse módulo irá alocar os operadores disponíveis para os produtos de maiores prioridades, e garantir com que os que entraram na lista para acabar com a matéria prima sejam produzidos.

Para esse caso, determinamos cinco operadores disponíveis, considerando que queremos esvaziar o *buffer* que contém a matéria prima MPA, e temos seis produtos na lista de programação, teremos que alocar os operadores primeiramente para os produtos que utilizam a MPA e que tenham maior prioridade, para depois alocar para os outros produtos.

São alocados os operadores para a produção dos produtos na seguinte ordem: P6, P5, P4, P2 e P1. O produto P12, que utiliza a matéria prima MPB, teve que ser excluído da programação final por não ter operadores disponíveis para sua produção.

A lista gerada é apresentada na Tabela 5.18.

Tabela 5.18 Lista de programação intermediária após alocação dos operadores para a validação do tratamento da ocorrência: Esvaziar *Buffer* de matéria prima.

Programação intermediária			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	0,55
P2	2	5	0,55
P4	4	5	1,11
P6	6	15	2,22
P5	5	10	2,77

Configurando o sistema computacional com o mesmo cenário estabelecido no caso de teste, devem-se obter os mesmos resultados validando assim o modelo para esse fluxograma.

Foi adicionada a mesma lista de programação inicial definida no caso de teste, conforme apresentado na Figura 5.26.

Sistema de Programação Reativa da Produção - [Programação da Produção]

Opções Ocorrências Configuração Programação

Informações da Programação Vigente

	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação

	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P2	2	5	5
	P12	12	5	5
	P4	4	5	4
	P6	6	15	3

Figura 5.26 Programação inicial inserida no sistema computacional.

Após a inserção da lista de programação inicial, foi inserida no sistema a lista de produtos preferenciais, conforme apresentado na Figura 5.27.

Sistema de Programação Reativa da Produção - [Lista de Inclusão]

Opções Ocorrências Configuração Programação

Produtos fora da Lista

- P1
- P3
- P4
- P6
- P7
- P8
- P9
- P10
- P11
- P12
- P13
- P14
- P16
- P18

Estoque: 15

CapMaxProd: 10

CapMinMP: 10

Margem Contrib.: 1

Observações: *

Produtos Lista de Inclusão

	Produto	M. Contrib.	Estoque	Roteiro	Prioridade	Obs
▶	P5	5	10	5	10	
	P15	15	15	15	9	
	P2	2	15	2	7	
	P17	17	15	17	6	

Preencher os Campos e Incluir Produto

Prioridade: Roteiro: 1

Incluir Produto na Lista

Remover Produto na Lista

Figura 5.27 Definição da lista de produtos preferenciais.

Foi estabelecida a margem de contribuição e a quantidade armazenada em estoque para cada um dos produtos, conforme ilustrado na Figura 5.28.

Produtos Cadastrados

Produto	Estoque	CapMax	CapMinMP	M. Contribuição	Família	Obs
P1	5	10	10	1	1	*
P2	5	10	10	2	1	*
P3	10	10	10	3	1	*
P4	5	10	10	4	1	*
P5	0	10	10	5	1	*

Cadastrar Produto

Família: A

Nome Produto:

Estoque Produto:

Cap Max Prod:

Cap Min Matéria Prima:

Margem Contribuição:

OBS:

Salvar Deletar

Figura 5.28 Definição do “nível de estoque” e “margem de contribuição” de cada produto.

Também foi definida a quantidade de matéria prima presente nos *buffers* de entrada, conforme apresentado na Figura 5.29.

Recursos Cadastrados

Recurso	Operador	Matéria Prima	Família	Quantidade	Obs
MPA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	1	550	
MPB	<input type="checkbox"/>	<input checked="" type="checkbox"/>	2	200	
MPC	<input type="checkbox"/>	<input checked="" type="checkbox"/>	3	100	
MP1	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
MP2	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
MP3	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
M1	<input type="checkbox"/>	<input type="checkbox"/>	0	0	

Cadastrar Recurso

Nome Recurso:

Quantidade:

Operador Matéria prima Família: A

OBS:

Limpar Dados do Formulário

Salvar Atualizar Deletar

Figura 5.29 Definição da quantidade de matéria prima nos buffers de entrada.

E a quantidade de operadores disponíveis também foi inserida no sistema, conforme ilustrado na Figura 5.30.

The screenshot shows a software window titled "Sistema de Programação Reativa da Produção - [Cadastrar Recurso]". It has a menu bar with "Opções", "Ocorrências", "Configuração", and "Programação". Below the menu is a table titled "Recursos Cadastrados" with the following data:

Recurso	Operador	Materia Prima	Familia	Quantidade	Obs
M32	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
M33	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
Esteira	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
EstAli	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
EstDes	<input type="checkbox"/>	<input type="checkbox"/>	0	0	
Operador	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0	5	

Below the table is a "Cadastrar Recurso" form with fields for "Nome Recurso", "Quantidade", and "OBS". There are also checkboxes for "Operador" and "Matéria prima", a dropdown menu, and buttons for "Limpar Dados do Formulário", "Salvar", "Atualizar", and "Deletar".

Figura 5.30 Definição da quantidade de operadores disponíveis.

Após ter configurado o cenário a ser testado, o próximo passo é estabelecer qual tratamento de ocorrência é desejado, nesse caso é: “Esvaziar *Buffer* de Matéria Prima” (Figura 5.31).

The screenshot shows a software window titled "Sistema de Programação Reativa da Produção - [Ocorrências]". It has a menu bar with "Opções", "Ocorrências", "Configuração", and "Programação". Below the menu is a section titled "Opções de Ocorrência" containing several buttons:

- Quebra de Máquina
- Esvaziar Buffer de Matéria Prima (highlighted with a thick black border)
- Falta de Local para Estocagem
- Falta de Matéria Prima
- Falta de Operador
- Inclusão Dura
- Exclusão Dura

Figura 5.31 Escolha do tratamento da ocorrência: Esvaziar *buffer* de matéria prima.

Após o tratamento da ocorrência é apresentada a programação reativa da produção, conforme ilustrada na Figura 5.32.

Sistema de Programação Reativa da Produção - [Programação da Produção]

Opções Ocorrências Configuração Programação

Informações da Programação Vigente

	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação

	Produto	Roteiro	QTD	Prioridade
▶	P6	6	15	3,3333
	P5	5	10	2,7778
	P4	4	5	2,2222
	P2	2	5	1,1111
	P1	1	5	0,5556

Figura 5.32 Programação reativa da produção, após tratamento da ocorrência: “Esvaziar *buffer* de entrada”.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse teste do tratamento da funcionalidade “Esvaziar *buffer* de entrada”.

5.2.6 Teste de validação da funcionalidade: Exclusão Dura

O sexto teste de validação ao qual o modelo foi submetido utilizou o fluxograma referente à “Exclusão Dura”. O teste foi realizado com a configuração apresentada na Tabela 5.19.

Tabela 5.19 Programação inicial para a validação do tratamento da ocorrência: Exclusão dura.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P7	7	5	4
P13	13	5	3
P14	14	5	5

Após determinada a lista de programação inicial, é apresentada no fluxograma da Figura 5.33 o caso de uso do tratamento da ocorrência “Exclusão Dura” que será seguido para a execução desse teste.

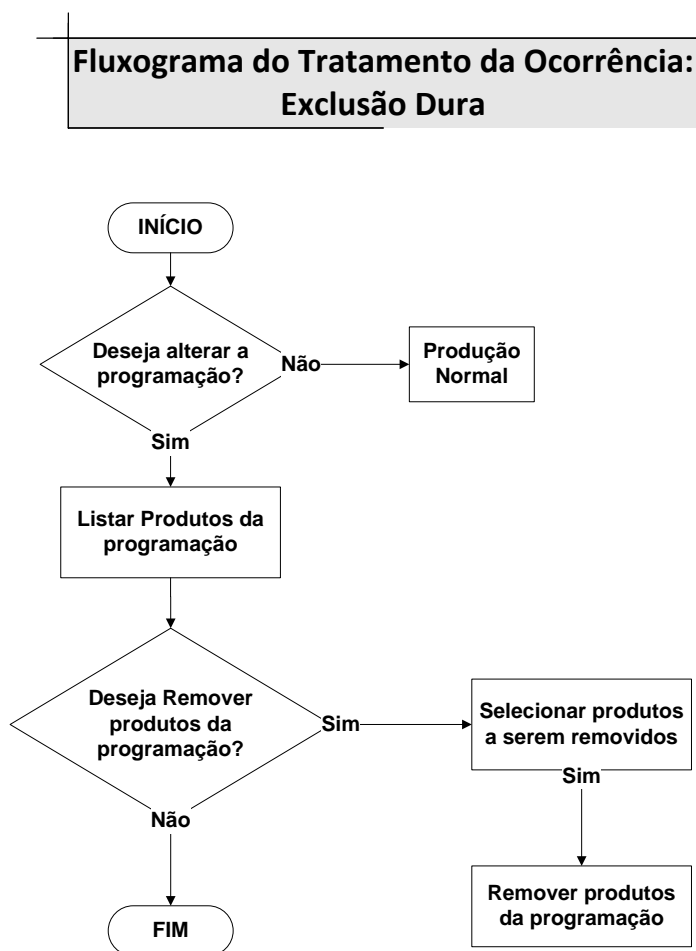


Figura 5.33 Fluxograma da funcionalidade: "Exclusão Dura".

Caso o usuário decida excluir algum produto sugerido na programação ou inserido na programação inicial, primeiramente ele precisa visualizar os produtos presentes na programação e definir qual ou quais deles serão removidos.

Para esse teste, iremos alterar a programação inicial removendo os produtos P7 e P13. Para tal tarefa, simplesmente remove-se os produtos desejados da programação, apresentando a nova programação ao usuário. Essa programação ficará de acordo com a Tabela 5.20.

Tabela 5.20 Programação reativa para a validação do tratamento da ocorrência: Exclusão dura.

Programação Reativa			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P14	14	5	5

Aplicando esse caso de teste no sistema computacional, é inserida no sistema a mesma programação inicial apresentada anteriormente, conforme apresentado na Figura 5.34.

	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

	Produto	Roteiro	QTD	Prioridade
▶	P1	1	5	7
	P14	14	5	5
	P2	2	5	5
	P7	7	5	4
	P13	13	5	3

Figura 5.34 Programação inicial inserida no sistema computacional.

Após a inserção da programação inicial, é necessário optar pela ocorrência desejada, nesse caso: “Exclusão Dura” (Figura 5.35).

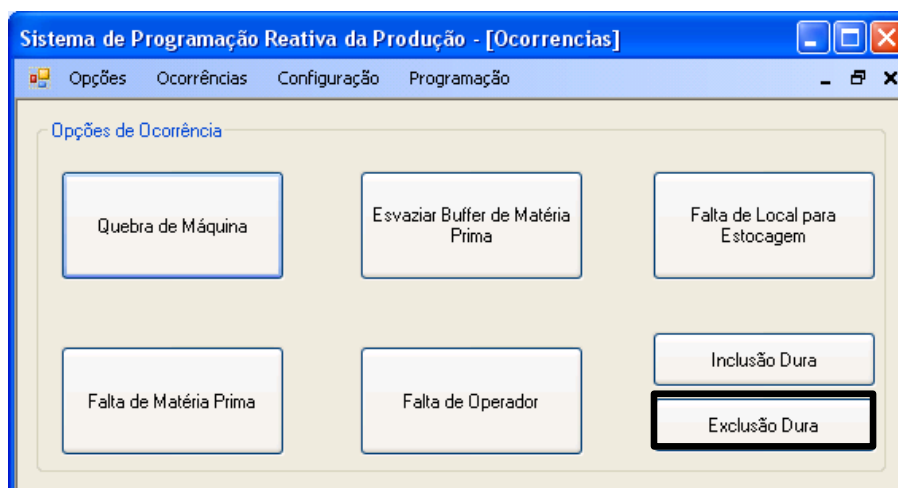


Figura 5.35 Escolha da ocorrência: "Exclusão Dura".

Após a escolha da ocorrência “Exclusão Dura”, todos os produtos presentes na programação corrente são listados. Nessa listagem, devem-se selecionar os produtos que deseja excluir da programação (Figura 5.36).

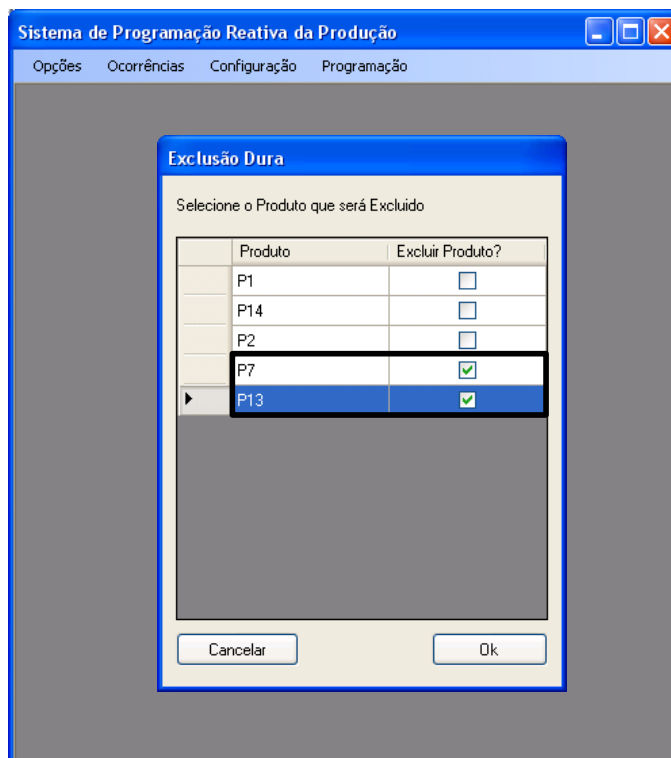


Figura 5.36 Seleção dos produtos a serem removidos da programação.

Após essa seleção dos produtos, o sistema realizará as devidas tarefas apresentando ao usuário a nova programação, nesse caso, conforme ilustrada na Figura 5.37.

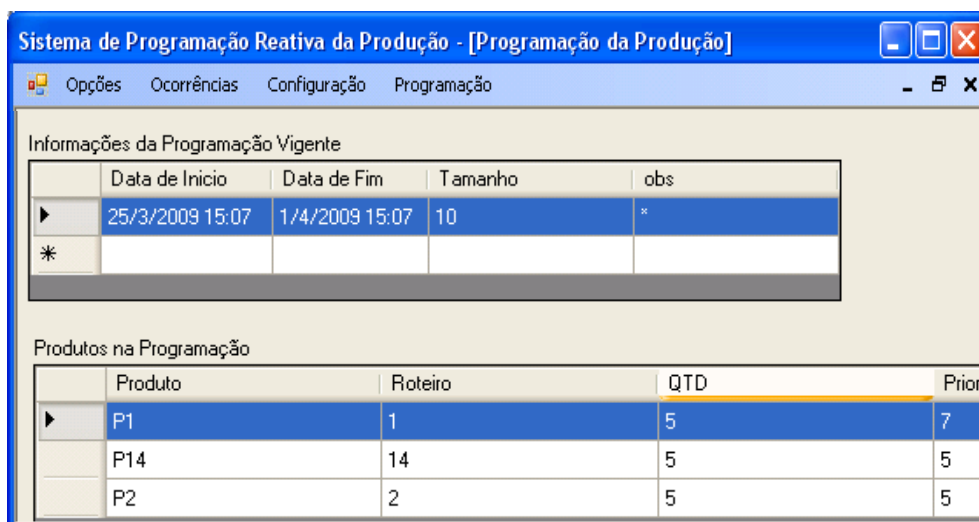


Figura 5.37 Apresentação da programação reativa da produção.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse teste do tratamento da funcionalidade “Exclusão Dura”.

5.2.7 Teste de validação da funcionalidade: Inclusão Dura

O sétimo e último teste de validação ao qual o modelo foi submetido utilizou o fluxograma referente à “Inclusão Dura”. O teste foi realizado com a configuração apresentada na Tabela 5.21.

Tabela 5.21 Programação inicial para a validação do tratamento da ocorrência: Inclusão dura.

Programação Inicial			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	8
P2	2	5	5
P13	13	5	2

Após determinada a lista de programação inicial, é apresentada no fluxograma da Figura 5.38 o caso de uso do tratamento da ocorrência “Inclusão Dura” que será seguido para a execução desse teste.

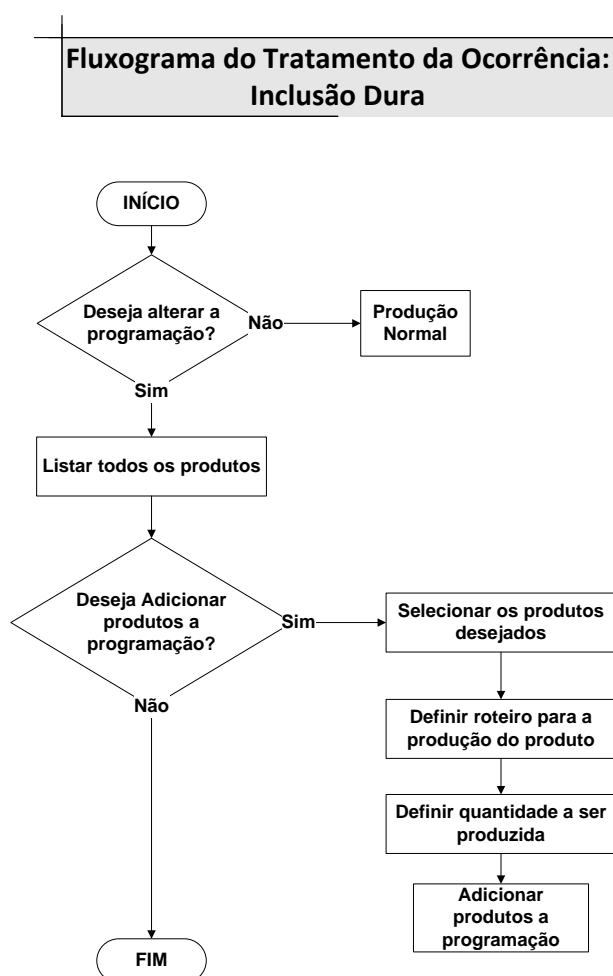


Figura 5.38 Fluxograma da funcionalidade: "Inclusão Dura".

Caso o usuário decida incluir algum produto na programação reativa ou na programação inicial, primeiramente ele precisa visualizar todos os produtos e definir qual ou quais deles serão incluídos, informando roteiro e a quantidade a ser produzida de cada um.

Para esse teste, iremos alterar a programação inicial adicionando o produto P6 com roteiro 6 e com a quantidade a ser produzida de 5 unidades. Para tal tarefa, simplesmente adiciona-se o produto desejado na programação, apresentando a nova programação ao usuário, conforme apresentado na Tabela 5.22.

Tabela 5.22 Programação reativa para a validação do tratamento da ocorrência: Inclusão dura.

Programação Reativa			
Produto	Roteiro	Quantidade Programada	Prioridade
P1	1	5	7
P2	2	5	5
P6	6	5	10
P13	13	5	5

Aplicando esse caso de teste no sistema computacional, é inserida no sistema a mesma programação inicial apresentada anteriormente, conforme ilustrado na Figura 5.39.

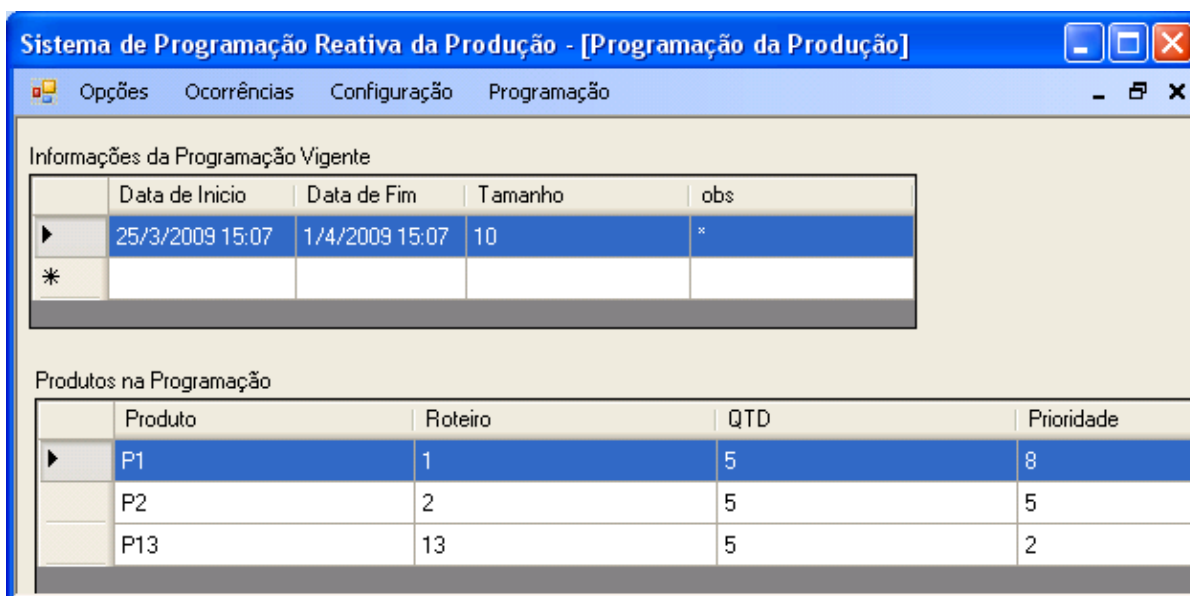


Figura 5.39 Programação inicial inserida no sistema computacional.

Após a inserção da programação inicial, é necessário optar pela ocorrência desejada, nesse caso: “Inclusão Dura” (Figura 5.40).

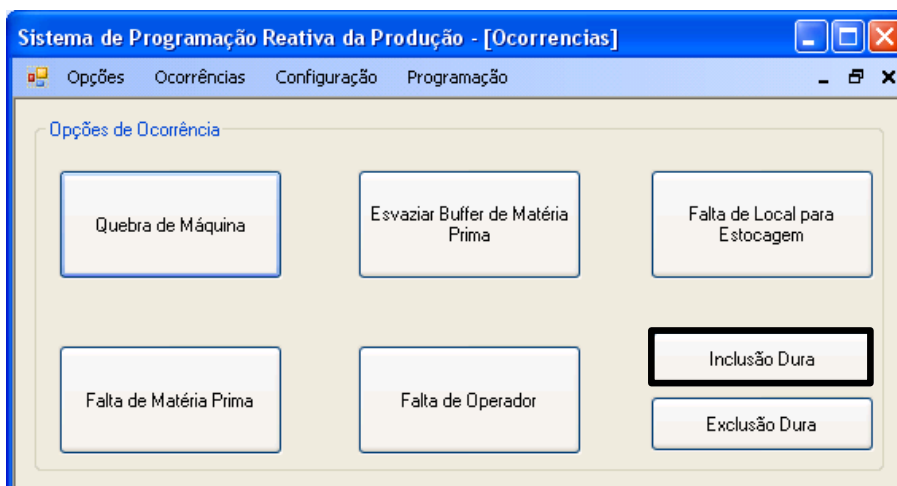


Figura 5.40 Escolha da ocorrência: "Inclusão Dura".

Após a escolha da ocorrência “Inclusão Dura”, todos os produtos são listados. Dessa listagem, devem-se selecionar os produtos que deseja incluir da programação informando seu roteiro e sua quantidade a ser produzida.

Para esse teste, será inserido o produto P6, com o roteiro 6 e 5 unidades a serem produzidas (Figura 5.41).

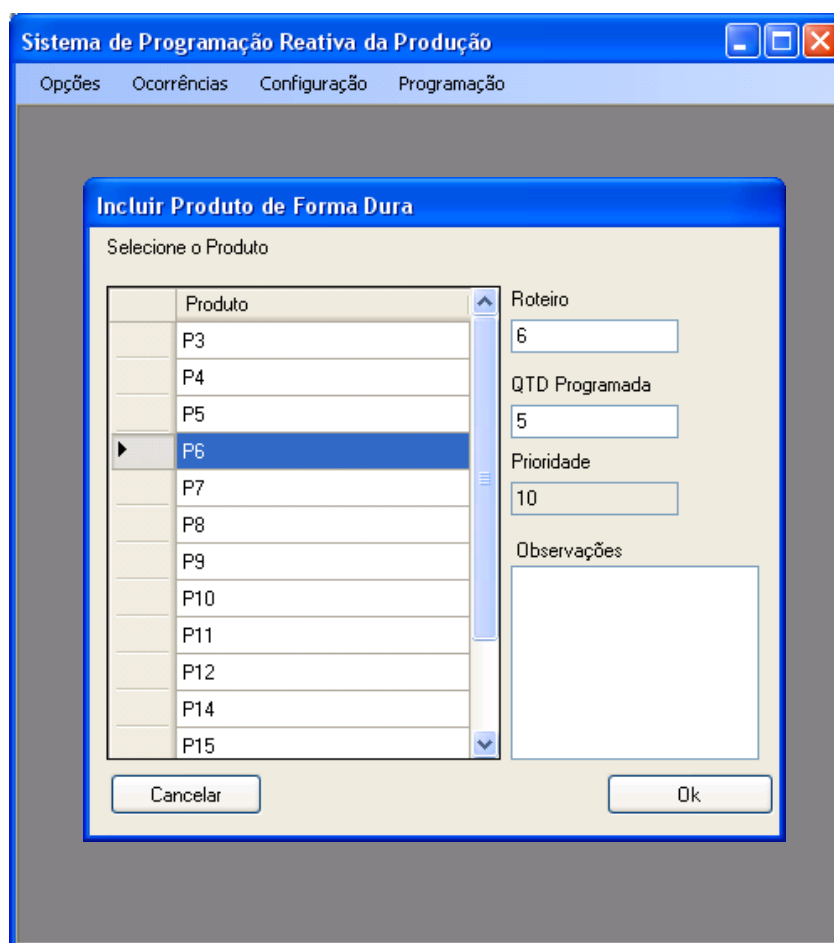


Figura 5.41 Inserção do produto P6.

Após a execução da funcionalidade, a programação reativa é apresentada para o usuário conforme ilustrado na Figura 5.42.

The screenshot shows a software window with the following data:

Informações da Programação Vigente				
	Data de Início	Data de Fim	Tamanho	obs
▶	25/3/2009 15:07	1/4/2009 15:07	10	*
*				

Produtos na Programação				
	Produto	Roteiro	QTD	Prioridade
▶	P6	6	5	10
	P1	1	5	8
	P2	2	5	5
	P13	13	5	2

Figura 5.42 Apresentação da programação reativa da produção.

Comparando a saída obtida pelo sistema computacional com a saída obtida pelo caso de teste, é possível avaliar que se obteve a mesma lista de programação reativa, confirmando o sucesso nesse último teste da funcionalidade “Inclusão Dura”.

Após o sucesso em todos os testes de validação, começando pela validação unitária dos módulos; passando pela validação da integração entre os módulos; pela validação da interface em conjunto com o usuário; depois pelas validações de funcionalidades de configuração do modelo e de sua lógica de tratamento das ocorrências; observou-se que o modelo cumpre com o seu objetivo, atingindo o resultado esperado.

6 Conclusão

No intuito de definir uma estratégia de modelagem para a programação reativa da produção, contemplando os aspectos da engenharia do conhecimento, esse trabalho apresentou algumas técnicas para a modelagem tanto do conhecimento implícito quanto o explícito, ambos importantes para a representação do modelo em relação ao domínio real.

Esse modelo foi direcionado para tratar problemas não planejados que possam vir a ocorrer durante a execução da programação. O conhecimento sobre esses problemas foram mapeados por meio de casos de uso e modelados em fluxogramas para representar o fluxo do tratamento do problema.

Também foram levantados os roteiros de fabricação com a ajuda de tabelas de aquisição do conhecimento, sendo esses roteiros modelados usando redes de Petri. A partir do estudo sobre o domínio em conjunto com os conhecimentos adquiridos e modelados, foi proposto um modelo das funcionalidades para representar o domínio desejado.

Esse modelo de funcionalidades agrega funções para o tratamento das ocorrências levantadas: uma modelagem *fuzzy* para definir novas prioridades em função de algumas variáveis pertencentes aos produtos e uma modelagem para limitar a produção em função de sua capacidade.

A partir do modelo de funcionalidade, foi proposta uma arquitetura para a construção de um sistema computacional que contemple a tratativa de todos os problemas previamente identificados. Por meio desse sistema foi realizada a validação da proposta.

A validação foi realizada primeiramente sobre duas óticas: validação do sistema e validação do modelo. A primeira, validação do sistema, foi realizada em duas etapas, validação dos módulos individualmente por meio de testes unitários, e validação da integração dos módulos para verificar a comunicação entre eles.

Sob a ótica de validação do modelo, após ter validado o sistema computacional, foram utilizados os casos de uso para verificar a consistência do modelo proposto. Foram percorridos todos os casos de uso e verificado se o modelo tem o mesmo comportamento do planejado.

6.1 Análise dos Resultados

Para a análise dos resultados, pode-se pensar em dois pontos: a aderência do modelo a realidade do domínio desejado, e o entendimento e aprovação das modelagens pelos envolvidos no domínio.

Para a verificação da aderência foram realizados os testes de validação no *software* confirmando o sucesso de acordo com o objetivo específico inicialmente estabelecido. Também foi verificado por meio da validação do modelo, o quanto ele se aproxima da realidade do domínio desejado, assegurando que o modelo cumpre com as funcionalidades previamente estabelecidas no objetivo geral.

O outro ponto, o entendimento e a aprovação das modelagens pelos envolvidos, foram verificados por meio do uso das mesmas técnicas de aquisição e representação do conhecimento usadas no projeto existente do laboratório TEAR e uma indústria.

Para essa verificação, foi envolvida uma grande quantidade de pessoas de várias áreas avaliando a facilidade com que os modelos foram entendidos, obtendo-se uma grande aceitação das técnicas utilizadas para a modelagem.

6.2 Trabalhos Futuros

Com base nesse trabalho, diversas extensões podem ser desenvolvidas, entre as quais se encontram descritas a seguir:

Aplicação de modelagem *fuzzy* em outros pontos que possam ter avaliações qualitativas envolvidas, como por exemplo, na limitação da produção em função de sua capacidade. Também é possível avaliar outras variáveis para determinar a nova prioridade do produto, como por exemplo, *makespan*.

Desenvolvimento de um sistema baseado em algoritmo genético para otimizar a resposta obtida pelo modelo em função de algum objetivo. Essa otimização pode ser, como por exemplo, melhorar o *makespan* da resposta obtida.

Incorporar sistemas de transporte ao domínio e criar um modelo para tratar possíveis ocorrências que esse novo elemento possa trazer, assim como aplicar algumas heurísticas para resolução dos conflitos.

Utilizar outras técnicas de representação e verificar a eficiência entre elas pelo ponto de vista do grau de representação, e também pelo impacto causado na compreensão dos modelos de representação por parte dos especialistas.

Ampliar os estudos de caso avaliando situações em que exista em espectro maior ou diferente do número de ocorrências que possam acontecer, impedindo a execução da programação e tornando o resultado do modelo mais próximo do real.

Criação de uma base de conhecimento e heurísticas apropriadas, permitindo assim o aprendizado com experiências passadas.

7 Referências

ALAVI, M.; LEIDNER, D. E. Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues. **MIS Quart Manage Inf Syst**, v. 25, n. 1, p. 107-136, March 2001.

BENINCASA, A. X.; MORANDIN, O. JR.; KATO, E. R. R., Reactive fuzzy dispatching rule for automated guided vehicles. In: IEEE Int Conf Syst Man Cybern, 2003, Washington. **Proceedings...** IEEE Inc, 2003, v. 5, P. 4375-4380.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **The Unified Modeling Language user guide**. 2º ed. Addison Wesley Longman Publishing, 2005.

BRACHMAN, R.; LEVESQUE, H. **Knowledge Representation and Reasoning**. San Francisco, USA: Morgan Kaufman, 2004.

CARVALHO, V. O.; MORANDIN, O. JR; KATO, E. R. R., Similarity measure between scenarios through fuzzy inference. In: IEEE Int Conf Syst Man Cybern, 2002, Yasmine Hammamet – Tunisia. **Proceedings...** IEEE Inc, 2002, v. 5, P. 135-140.

CASTRO, P. A. D.; PIRES, M. G.; CAMARGO, H. A.; MORANDIN, O. JR.; KATO, E. R. R., Genetic learning of fuzzy rules applied to sequencing problem of FMS. In: IEEE Int Conf Syst Man Cybern, 2004, The Hague - Holanda. **Proceedings...** IEEE Inc, 2004, v. 5, P. 4336-4341.

CHORAFAS, D. **Knowledge engineering**. New York, USA: Van Nostrand Reinhold Co. 1990.

COCHRANE, S.; YOUNG, R.; CASE, K.; HARDING, J.; GAO, J.; DANI, S.; BAXTER, D. Knowledge Reuse in Manufacturability Analysis. **Rob Comput Integr Manuf**, Oxford, v. 24, n. 4, p. 508-513, August 2008.

DERIZ, A. C. Um método de busca usando algoritmos genéticos para a programação da produção em sistemas de manufatura com recursos compartilhados. **Dissertação de Mestrado**. Universidade Federal de São Carlos – Departamento de Computação. São Carlos, 2007.

DOKAS, I.; PANAGIOTAKOPOULOS, D. A knowledge acquisition process to analyze operational problems in solid waste management facilities. **Waste Management & Research**, Northampton, v. 24, n. 4, p. 332, August 2006.

FUGATE, B. S.; STANK, T. P.; MENTZER, J. T. Linking improved knowledge management to operational and organizational performance. **Journal of Operations Management**, Amsterdam, v. 27, n. 3, p. 247-264, June 2009.

GIARRATANO, J.; RILEY, G. **Expert systems: principles and programming**. México: PWS Publishing Company, 1998.

GRUNDSPENKIS, J. Agent Based Approach for Organization and Personal Knowledge Modelling: Knowledge Management Perspective. **Journal of Intelligent Manufacturing**, Dordrecht, v. 18, n. 4, p. 451-457, August 2007.

GUEDES, G. **Guia de consulta rápida UML 2**. São Paulo: Novatec, 2005.

GUERRA, D. A.; YOUNG, R. A manufacturing model to enable knowledge maintenance in decision support systems. In: North American Manufacturing Research Conference, 33, 2005, New York. **Proceedings...** Dearborn, 2005. P. 203-210.

HALEI, G.; WANG, K. Knowledge based manufacturing system (KBMS). **J Intell Manuf**, Dordrecht, v. 18, n. 4, p. 467-474, August 2007.

HENNING, G. P.; CERDÁ, J. Knowledge-based predictive and reactive scheduling in industrial environments. **Comput. Chem. Eng**, v. 24, n. 9-10, p. 2315-2338, October 2000.

HJELMERVIK, O. R.; WANG, K. ICT – supported knowledge representation for development of routines in industry. **J Intell Manuf**, Dordrecht, v. 18, n.4, p. 479-485, August 2007.

JANAK, S. L.; FLOUDAS, C. A.; KALLRATH, J.; VORMBROCK, N. Production scheduling of a large-scale industrial batch plant. II. Reactive scheduling. **Industrial and Engineering Chemistry Research**, Columbus, v. 45, n. 25, p. 8253-8269, December 2006.

JINSONG, Z.; QIFU, W.; LI, W.; YIFANG, Z. Configuration-Oriented product modelling and knowledge management for made-to-order manufacturing enterprises. **Int J Adv Manuf Technol**, London, v. 25, n. 1-2, p. 41-52, January 2005.

KASABOV, N. **Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering**. London, UK: Bradford Books, 1996.

KENDAL, S.; CREEN, M. **An Introduction to Knowledge Engineering**. London, UK: Springer-Verlag, 2007.

KLIR, G.; YUAN, B. *Fuzzy sets and fuzzy logic*. Upper Saddle River, NJ: Prentice Hall, 1995.

LI, Z.; IERAPETRITOU, M. Process scheduling under uncertainty: Review and challenges. **Computers and Chemical Engineering**, Oxford, v. 32, n. 4-5, p. 715-727, April 2008.

MAIA, J.L.; DESCO, M.B.; MORANDIN, O. JR.; KATO, E.R.R., Expanding the production planning system based on simulation (PPSS) to dynamically reschedule production in a FMS. In: IEEE Int Conf Syst Man Cybern, 2002, Yasmine Hammamet – Tunisia. **Proceedings...** IEEE Inc, 2002, v. 5, P. 685-690.

MANAF, N. A. A.; BEIKZADEH, M. R. Representation and reasoning of *fuzzy* temporal knowledge. In: IEEE Conference on Cybernetics and Intelligent Systems, 2006, Bangkok. **Proceedings...** Piscataway, 2006. n. 4017811.

METAXIOTIS, K. RECOT: An expert system for the reduction of environmental cost in the textile industry. **Inf Manage Comput Secur**, v.12, n.3, p.218-227, 2004.

MILTON, N. R., **Knowledge Acquisition in Practice: A Step-by-step Guide**. London, UK: Springer-Verlag, 2007.

MITCHELL, T. **Machine Learning**. Boston, USA: Mac Graw Hill, 1997.

MORANDIN, O. JR.; KATO, E.R.R.; POLITANO, P.R.; CAMARGO, H.A.; PORTO, A.J.V.; INAMASU, R.Y., Modular modeling approach for automated manufacturing systems based on shared resources and process planning using Petri Nets. In: IEEE Int Conf Syst Man Cybern, 2000, Nashville – USA. **Proceedings...** Piscataway, 2000, v. 4, P. 3057-3062.

MORANDIN, O. JR.; KATO, E. R.R., Virtual Petri nets as a modular modeling method for planning and control tasks of FMS. **Int J Computer Integr Manuf**, Taylor and Francis Ltd, v. 18, n. 2-3, p. 100-106, March/ May 2005.

MORANDIN, O. JR.; CASTRO, DALBEM, P., A.; KATO, E. R. R.; CAMARGO, H. A., A genetic fuzzy system for defining a reactive dispatching rule for AGVs. In: IEEE Int Conf Syst Man Cybern, 2006, Taipei – Taiwan. **Proceedings...** Piscataway, 2006, v. 1, P. 56-61.

MORANDIN, O. JR.; KATO, E. R. R.; ARAÚJO, R. G.; SASSO, L. V., A modeling strategy for control and interlocking of an ams using virtual petri nets. In: IEEE Int Conf Syst Man Cybern, 2007, Montreal – Canada. **Proceedings...** IEEE Inc, 2007, p. 3493-3498.

MORANDIN, O JR.; KATO, E. R. R.; MONTORO, F. A.; OLIVEIRA, V. C., A production sequencing model for a decision support system. In: IEEE Int. Conf. Comput. Intell. Model. Control. Autom., 2008, Vienna – Austria. **Proceedings**... IEEE Inc, 2008, p. 814-819.

MORENO, M. D. R.; BORRAJO, D.; CESTA, A.; ODDI, A. Integrating Planning and Scheduling in Workflow domains. **Expert Systems with Applications**, Oxford, v. 33, n. 2, p. 389-406, August 2007.

MURATA, T. Petri nets: Properties, analysis and applications. In: Proceedings of the IEEE, vol.77, no.4, pp.541-580, April 1989. Disponível em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=24143&isnumber=911>. Acesso em: 28/04/2009.

MUSZYNSKI, W.; BANASZAK, Z.; TOMCZUK-PIRÓG, I. Automated vehicles' work planning in flexible manufacturing systems. In: IEEE Conference on Emerging Technologies and Factory Automation, 11., 2006, Hamburg. **Proceedings**... Piscataway, 2006. P. 813-818.

O'KANE, J. F. A knowledge-based system for reactive scheduling decision-making in FMS. **Journal of Intelligent Manufacturing**, Dordrecht, v. 11, n. 5, p. 461-474, October 2000.

POLITANO, P. R.; CAMARGO, H. A.; KATO E. R. R.; MORANDIN, O. JR. A reactive programming procedure for flexible manufacturing system. In: IEEE Int Conf Syst Man Cybern, 2000, Nashville – USA. **Proceedings**... Piscataway, 2000, v. 3, P. 2156-2161.

REZENDE, S. O. **Sistemas inteligentes: Fundamentos e Aplicação**. Barueri, São Paulo: Manole, 2003.

ROBINSON, S.; ALIFANTIS, T.; EDWARDS JS.; LADBROOK, J.; WALLER, A. Knowledge-based improvement: simulation and artificial intelligence for identifying and improving human decision-making in an operations system. **J Oper Res Soc**, v.56, n. 8, p. 912-921, 2005. Disponível em: <http://dx.doi.org/10.1057/palgrave.jors.2601915>. Acesso em: 28/04/2009.

SABUNCUOGLO, I.; KIZILISIK, O. B. Reactive scheduling in a dynamic and stochastic FMS environment. **Int J Prod Res**, v. 41, n. 17, p. 4211-4231, November 2003.

STEFANOVITZ, J. P.; NAGANO, M. S. Aquisição e Criação de Conhecimento na Indústria de Alta Tecnologia. **Revista Produção Online**. v. 6, n. 1, 2006.

SUN, J.; XUE D. A dynamic reactive scheduling mechanism for responding to changes of production orders and manufacturing resources. **Computers in Industry**, v. 46, n. 2, p. 189-207, September 2001.

TANG, L.; WANG X. A predictive reactive scheduling method for color-coating production in steel industry. **Int J Adv Manuf Technol**, Guildford, v. 35, n. 7-8, p. 633-645, January 2008.

TIAN, Y.; WANG Y. A knowledge representation tool based on concept Algebra. In: IEEE International Conference on Cognitive Informatics, 6., 2007, Lake Tahoe. **Proceedings...** Piscataway, 2007. p. 294-301.

WANG, Y. On concept algebra and knowledge representation. In: IEEE International Conference on Cognitive Informatics, 5., 2006, Beijing. **Proceedings...** Piscataway, 2006. p. 320-331.

WANG, Y. The OAR model for knowledge representation. In: Canadian Conference on Electrical and Computer Engineering, 6., 2007, Ottawa. **Proceedings...** Piscataway, 2006. p. 1727-1730.

WOLFF, V.; LEFEBVRE, A.; RENAUD, J. Maps of Dispersions for Machining Process. **Concurrent Engineering**, v. 14, n. 2 p. 129-139, 2006.

YANG, J.-B. Developing a knowledge map for construction scheduling using a novel approach. **Autom Constr**, Netherlands, v. 16, n. 6, p. 806-815, September 2007.

YIM N.; KIM S.; KIM H.; KWAHK K. Knowledge based decision making on higher level strategic concerns: system dynamics approach. **Expert Sys Appl**, v. 27, n. 1, p. 143-158, July 2004.

8 Apêndice A – Aquisição e Representação dos Roteiros de Fabricação

Para cada produto foi criada uma tabela apresentando todos os passos para a produção de cada um deles. Essa tabela contém todos os elementos necessários para determinar o roteiro de fabricação.

Foram preenchidas 18 tabelas, correspondentes aos 18 produtos existentes, listando os processos atrelados aos produtos e os recursos atrelados aos processos.

Serão apresentadas as tabelas dividindo os produtos de acordo com sua família (A, B ou C). Os produtos da família A, que consiste dos produtos P1 ao P6, sendo que o produto P6 apresenta dois roteiros de fabricação distintos, são apresentados a seguir.

Tabela 8.1 Tabela de aquisição do roteiro de fabricação do produto P1.

Família	Produto	Roteiro
A	P1	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 11 / Operador
5	A5	Máquina 12 / Operador

Tabela 8.2 Tabela de aquisição do roteiro de fabricação do produto P2.

Família	Produto	Roteiro
A	P2	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 12 / Operador
5	A5	Máquina 13 / Operador

Tabela 8.3 Tabela de aquisição do roteiro de fabricação do produto P3.

Família	Produto	Roteiro
A	P3	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 11 / Operador
5	A5	Máquina 13 / Operador

Tabela 8.4 Tabela de aquisição do roteiro de fabricação do produto P4.

Família	Produto	Roteiro
A	P4	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 11 / Operador
5	A5	Máquina 12 / Operador
6	A6	Máquina 13 / Operador

Tabela 8.5 Tabela de aquisição do roteiro de fabricação do produto P5.

Família	Produto	Roteiro
A	P5	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 11 / Operador

Tabela 8.6 Tabela de aquisição do roteiro de fabricação do produto P6 roteiro 1.

Família	Produto	Roteiro
A	P6	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 12 / Operador

Tabela 8.7 Tabela de aquisição do roteiro de fabricação do produto P6 roteiro 2.

Família	Produto	Roteiro
A	P6	2

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima A
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 13 / Operador

Os produtos da família B, que consiste do produto P7 ao P12, sendo que o produto P12 apresenta dois roteiros de fabricação distintos, é apresentado a seguir.

Tabela 8.8 Tabela de aquisição do roteiro de fabricação do produto P7.

Família	Produto	Roteiro
B	P7	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 21 / Operador
5	A5	Máquina 22 / Operador

Tabela 8.9 Tabela de aquisição do roteiro de fabricação do produto P8.

Família	Produto	Roteiro
B	P8	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 22 / Operador
5	A5	Máquina 23 / Operador

Tabela 8.10 Tabela de aquisição do roteiro de fabricação do produto P9.

Família	Produto	Roteiro
B	P9	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 21 / Operador
5	A5	Máquina 23 / Operador

Tabela 8.11 Tabela de aquisição do roteiro de fabricação do produto P10.

Família	Produto	Roteiro
B	P10	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 21 / Operador
5	A5	Máquina 22 / Operador
6	A6	Máquina 23 / Operador

Tabela 8.12 Tabela de aquisição do roteiro de fabricação do produto P11.

Família	Produto	Roteiro
B	P11	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 21 / Operador

Tabela 8.13 Tabela de aquisição do roteiro de fabricação do produto P12 roteiro 1.

Família	Produto	Roteiro
B	P12	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 22 / Operador

Tabela 8.14 Tabela de aquisição do roteiro de fabricação do produto P12 roteiro 2.

Família	Produto	Roteiro
B	P12	2

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima B
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 23 / Operador

Os produtos da família C, que consiste do produto P13 ao P18, sendo que o produto P18 apresenta dois roteiros de fabricação distintos, é apresentado a seguir.

Tabela 8.15 Tabela de aquisição do roteiro de fabricação do produto P13.

Família	Produto	Roteiro
C	P13	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 31 / Operador
5	A5	Máquina 32 / Operador

Tabela 8.16 Tabela de aquisição do roteiro de fabricação do produto P14.

Família	Produto	Roteiro
C	P14	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 32 / Operador
5	A5	Máquina 33 / Operador

Tabela 8.17 Tabela de aquisição do roteiro de fabricação do produto P15.

Família	Produto	Roteiro
C	P15	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 31 / Operador
5	A5	Máquina 33 / Operador

Tabela 8.18 Tabela de aquisição do roteiro de fabricação do produto P16.

Família	Produto	Roteiro
C	P16	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 31 / Operador
5	A5	Máquina 32 / Operador
6	A6	Máquina 33 / Operador

Tabela 8.19 Tabela de aquisição do roteiro de fabricação do produto P17.

Família	Produto	Roteiro
C	P17	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 31 / Operador

Tabela 8.20 Tabela de aquisição do roteiro de fabricação do produto P17 roteiro 1.

Família	Produto	Roteiro
C	P18	1

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 32 / Operador

Tabela 8.21 Tabela de aquisição do roteiro de fabricação do produto P17 roteiro 2.

Família	Produto	Roteiro
C	P18	2

	Processo	Recurso
1	A1	Esteira / Máquina 1 / Peça Triangular / Esteira de Alimentação / Matéria Prima C
2	A2	Esteira / Máquina 2 / Peça Quadrada
3	A3	Esteira / Máquina 3 / Peça Retangular / Esteira de Descarga
4	A4	Máquina 33 / Operador

Depois de preenchidas todas as tabelas, as informações contidas nelas foram modeladas em Redes de Petri. Essa modelagem representa o conhecimento relativo aos roteiros de produção, apresenta o fluxo de produção do produto, mostrando desde a entrada da matéria prima até o produto finalizado, indicando o momento em que os recursos são utilizados.

Foram gerados 18 modelos em Redes de Petri, sendo cada modelo representativo de um produto. Os produtos que apresentaram mais de um roteiro foram modelados por apenas um único modelo em Redes de Petri contendo os dois roteiros.

A seguir são apresentados os modelos, sendo divididos pelas famílias dos produtos. Os primeiros modelos de produtos apresentados são os produtos pertencentes a família A, que contempla do P1 ao produto P6.

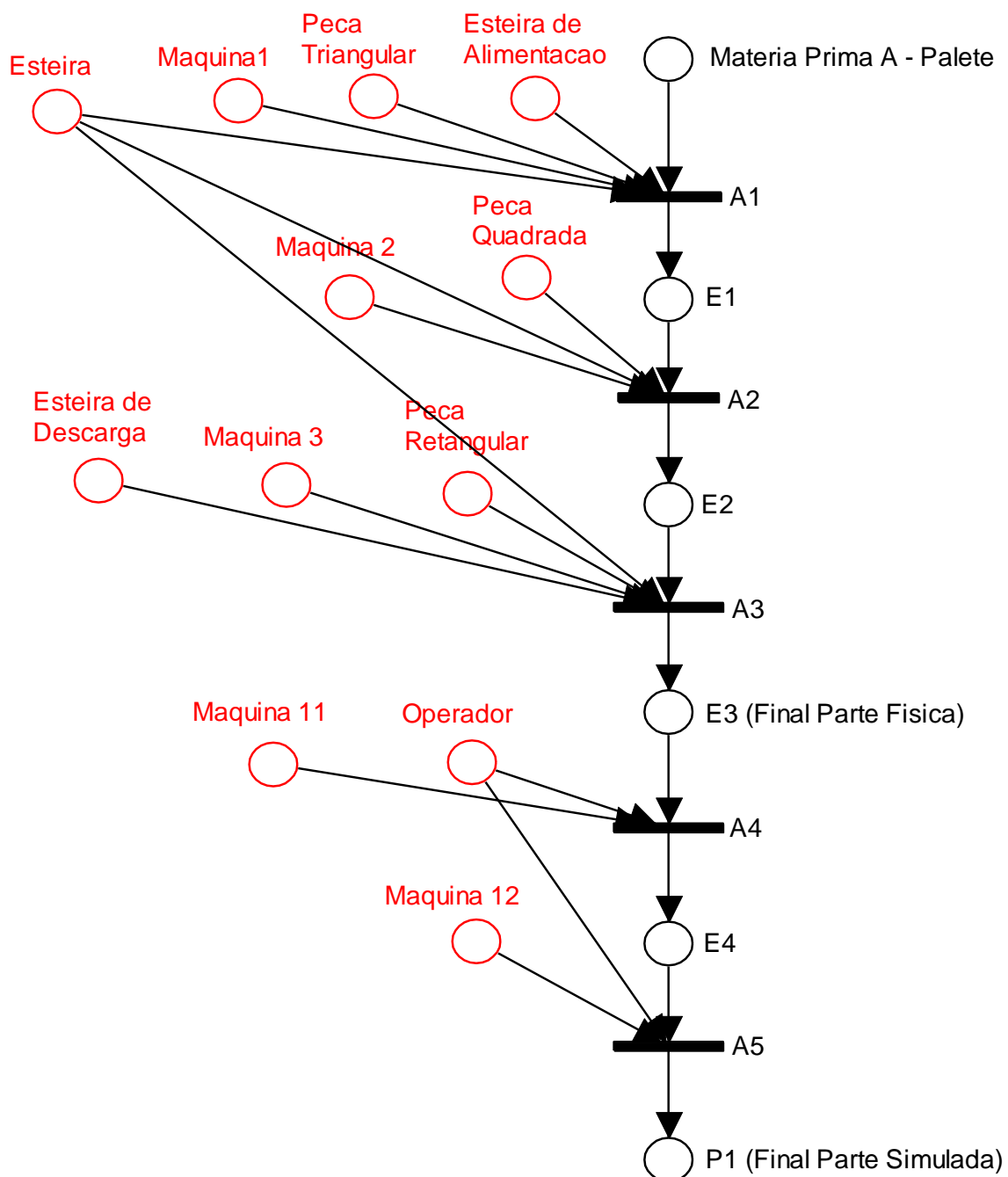


Figura 8.1 Modelo do roteiro de fabricação do produto P1.

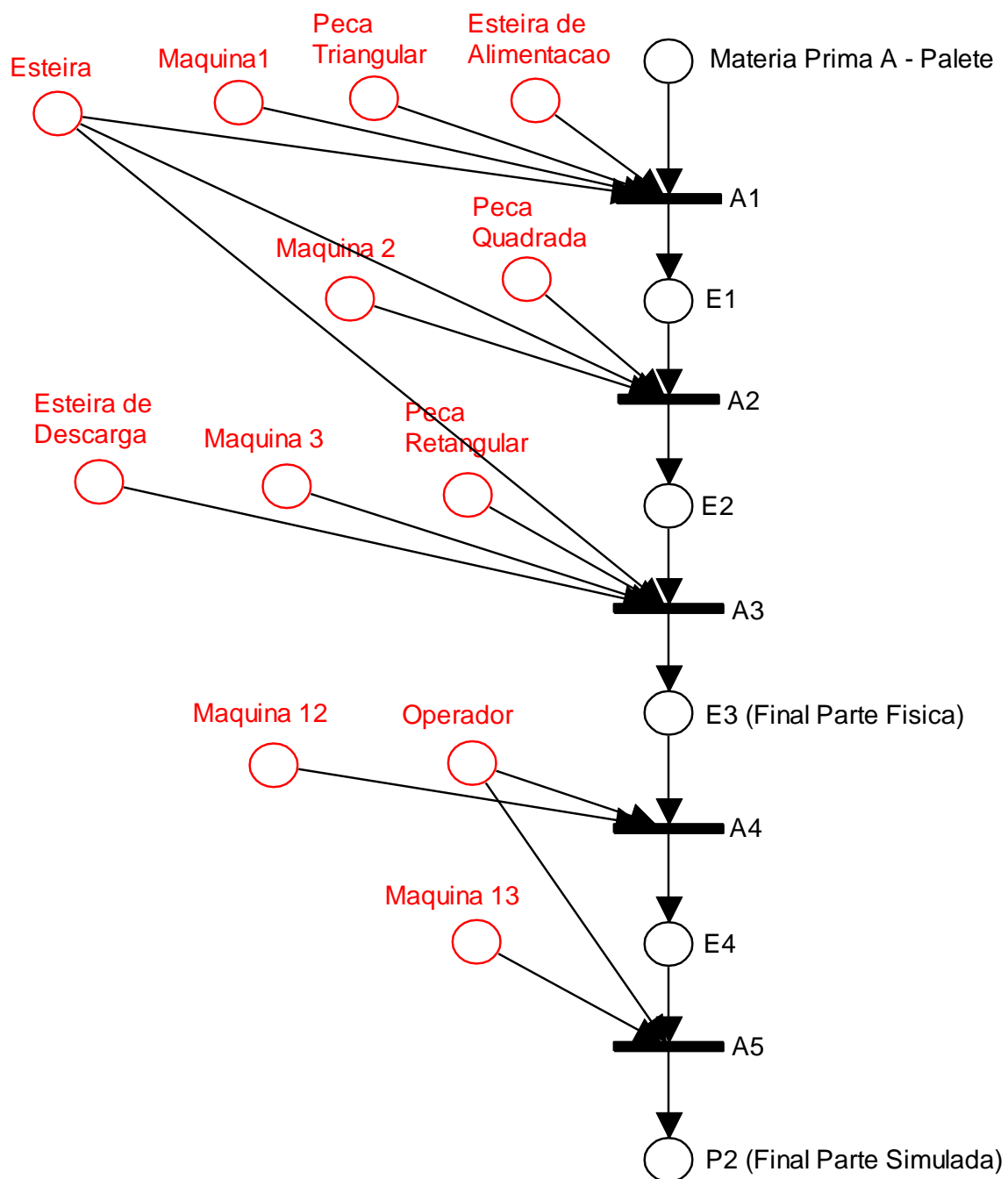


Figura 8.2 Modelo do roteiro de fabricação do produto P2.

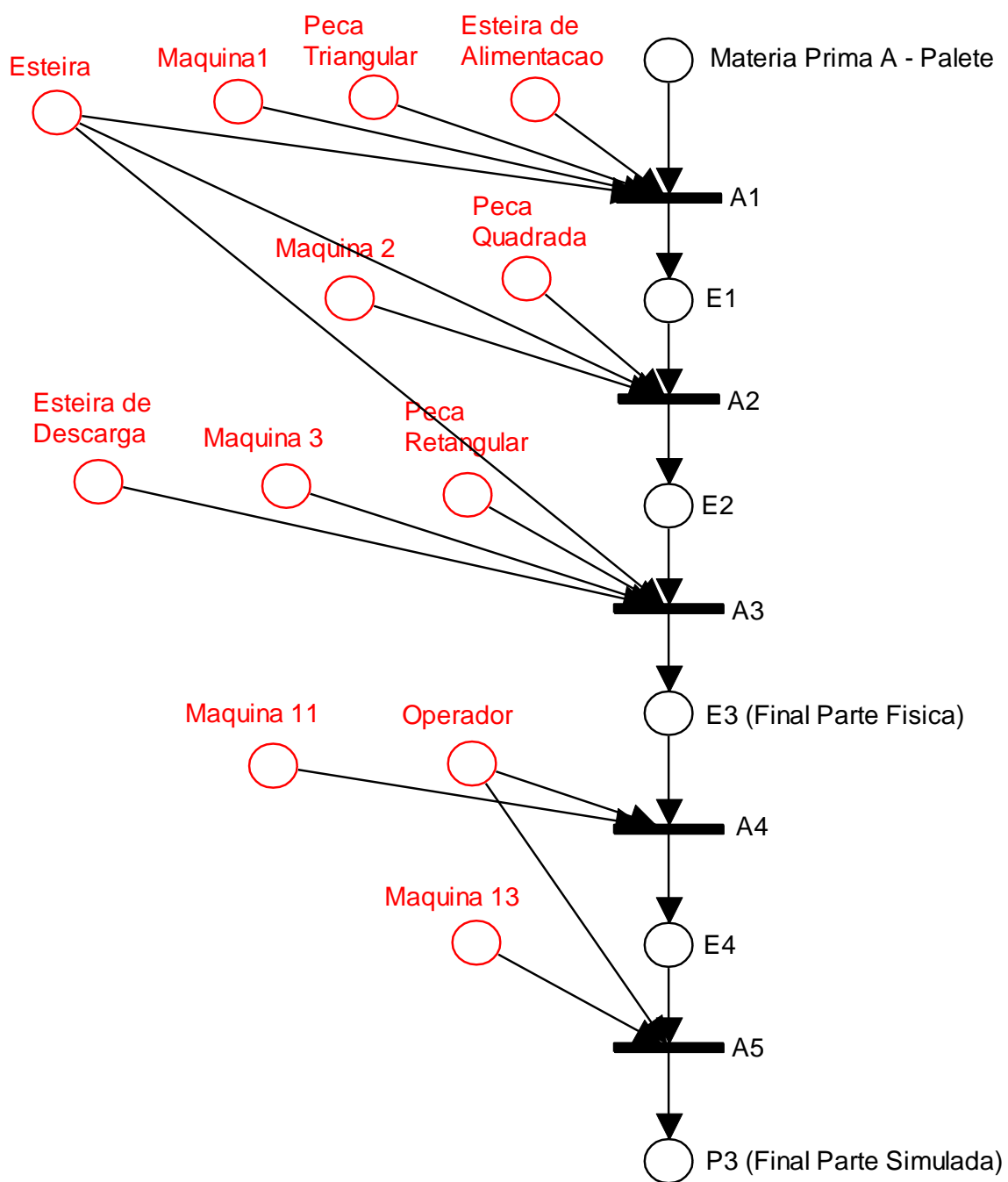


Figura 8.3 Modelo do roteiro de fabricação do produto P3.

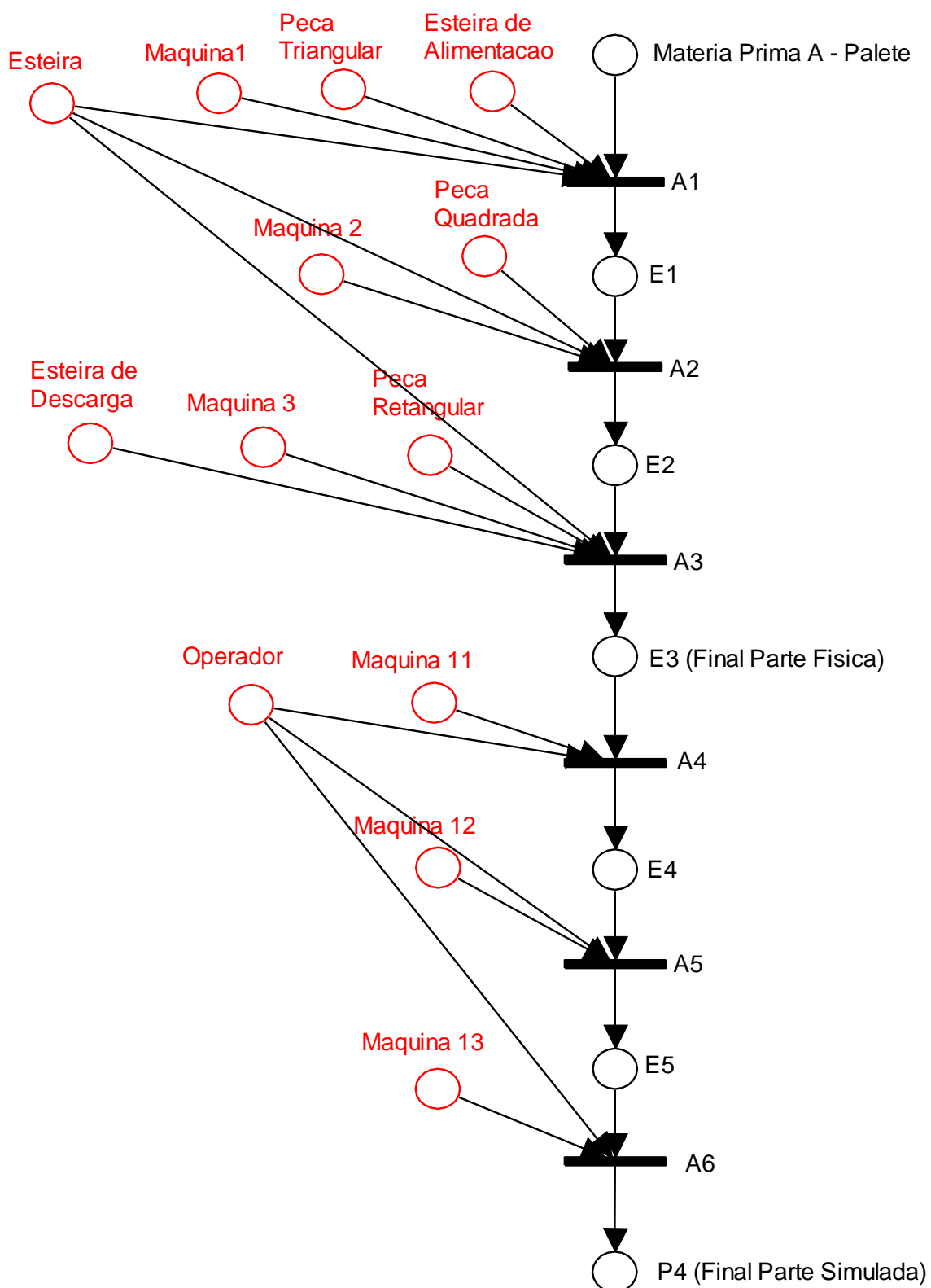


Figura 8.4 Modelo do roteiro de fabricação do produto P4.

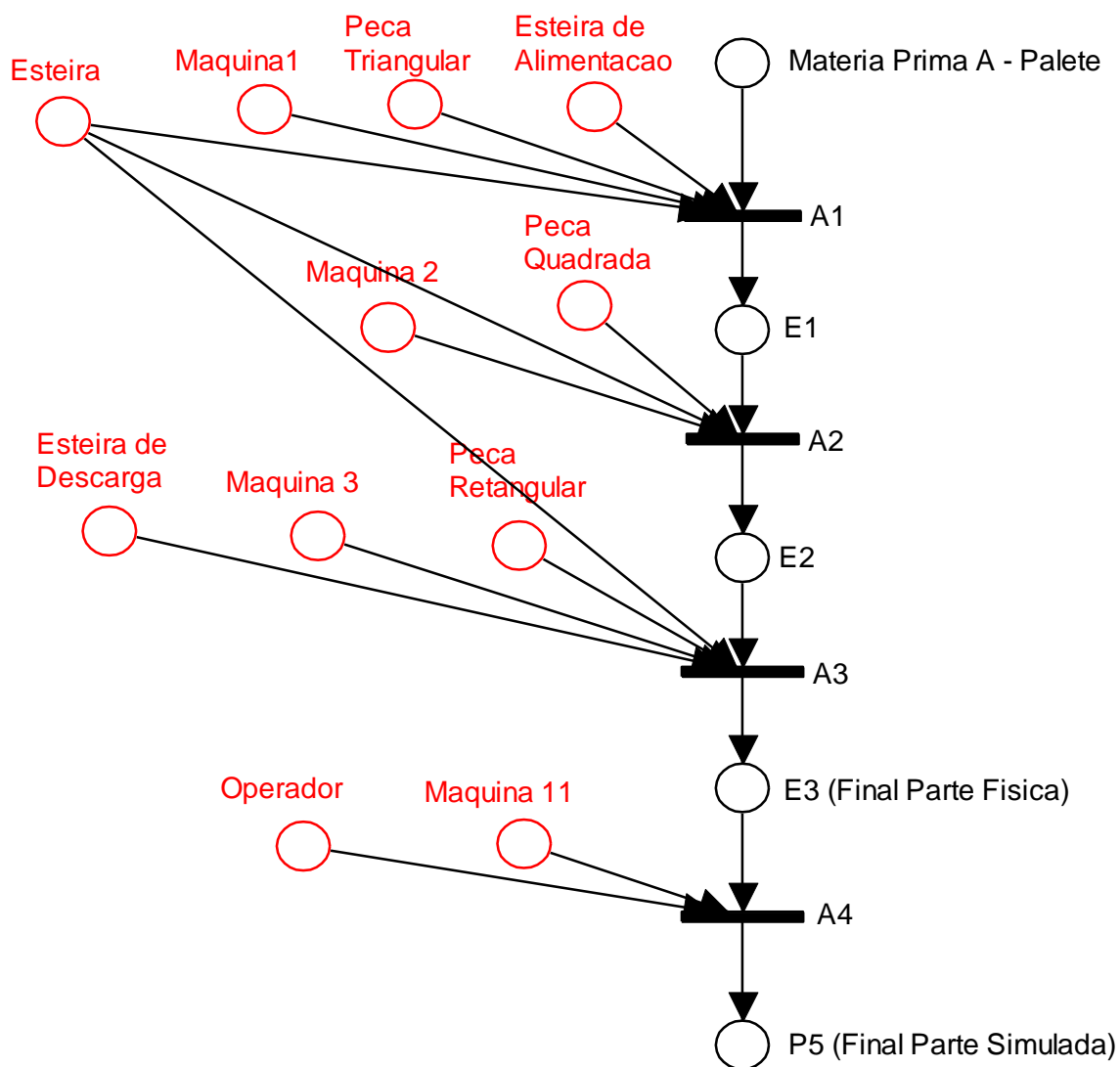


Figura 8.5 Modelo do roteiro de fabricação do produto P5.

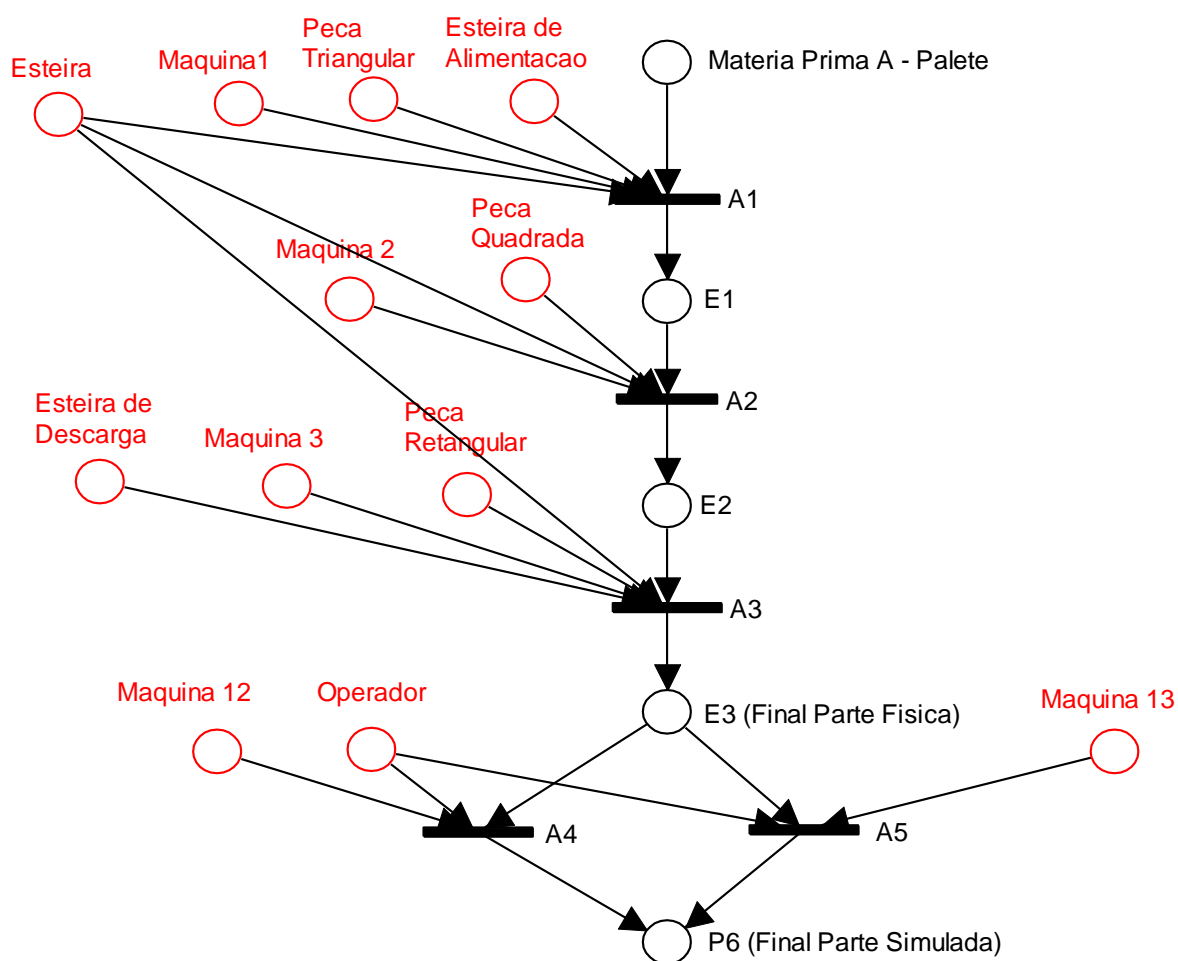


Figura 8.6 Modelo do roteiro de fabricação do produto P6.

Os produtos da família B, que consiste do produto P7 ao P12, são apresentados a seguir:

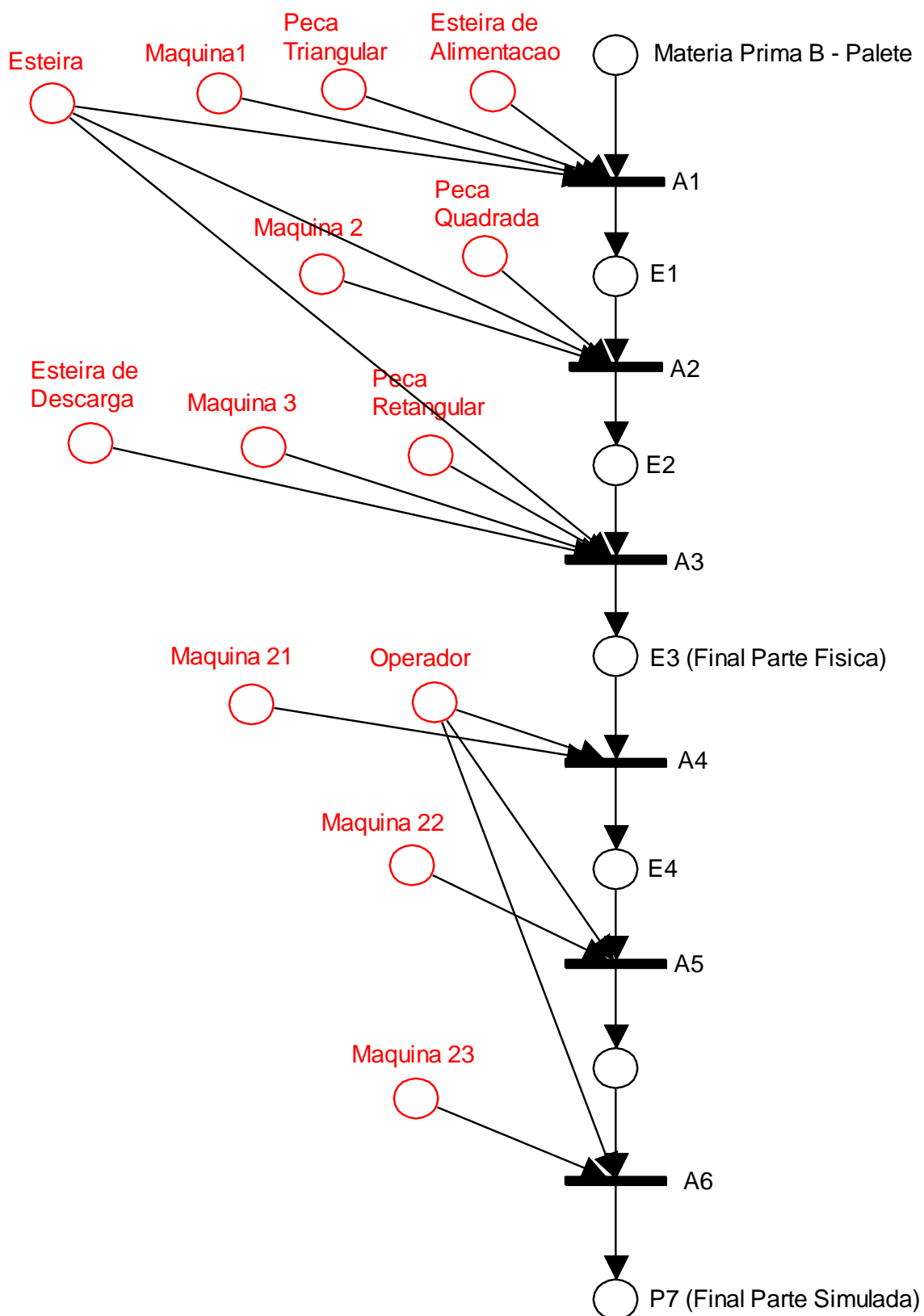


Figura 8.7 Modelo do roteiro de fabricação do produto P7.

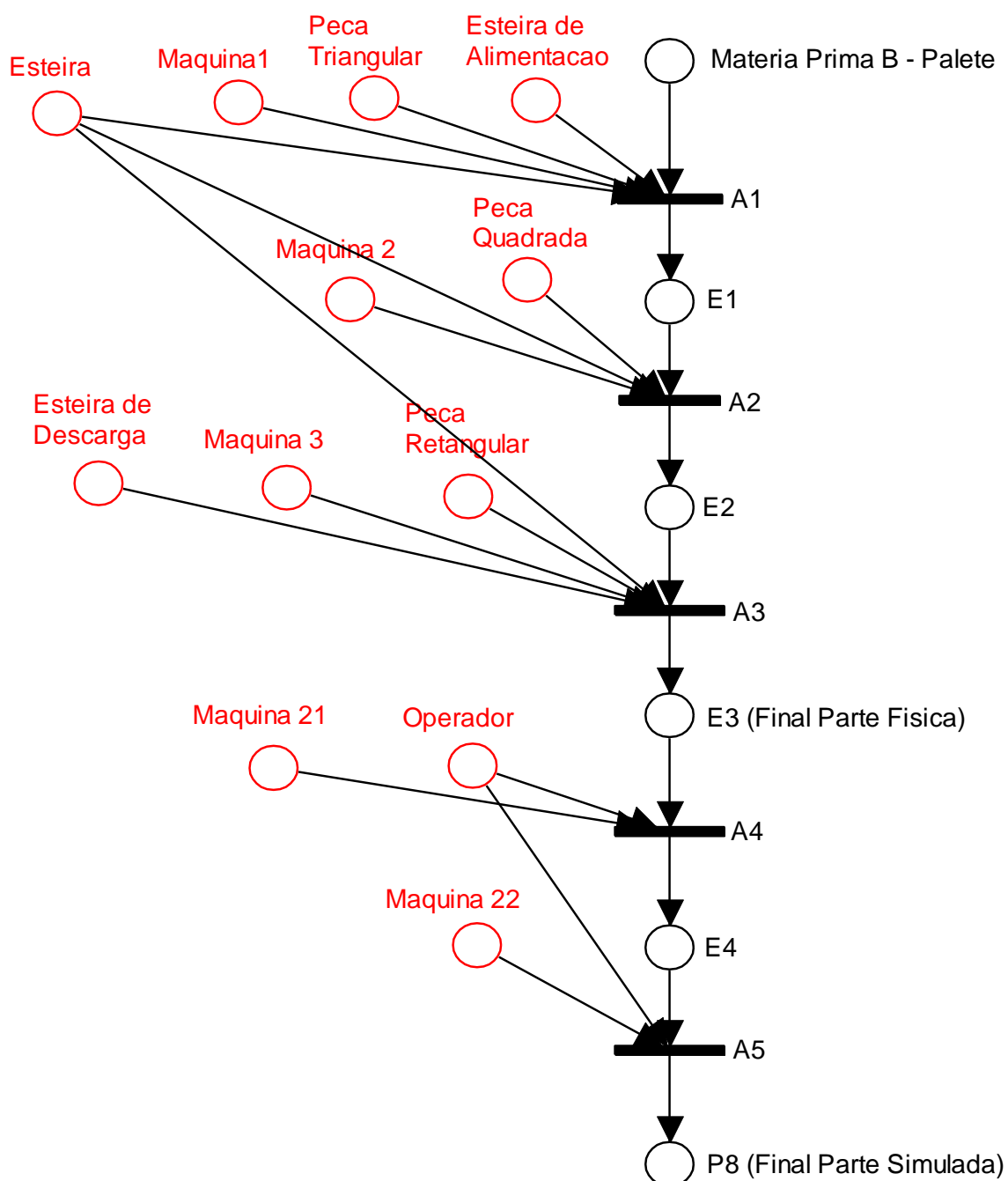


Figura 8.8 Modelo do roteiro de fabricação do produto P8.

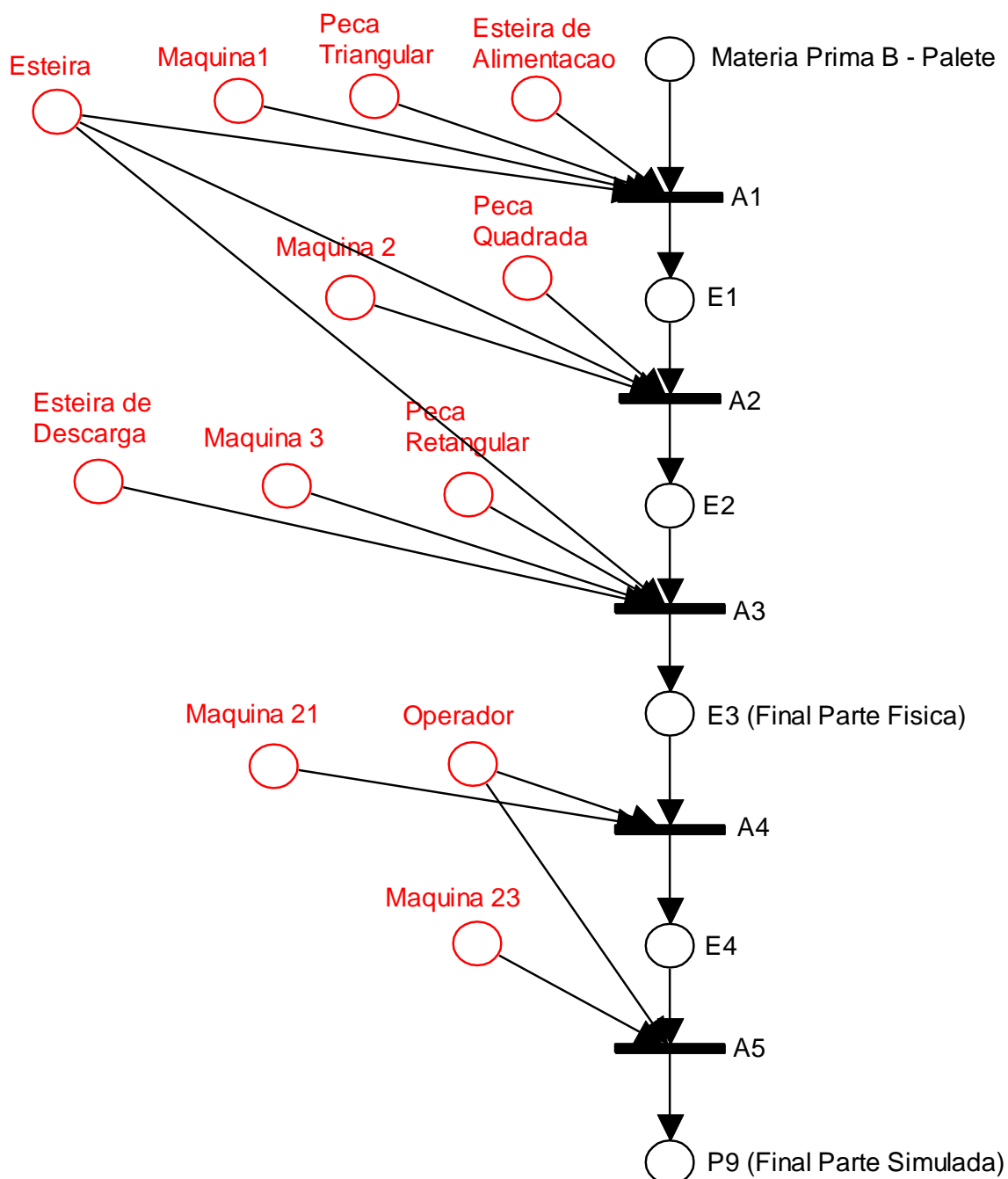


Figura 8.9 Modelo do roteiro de fabricação do produto P9.

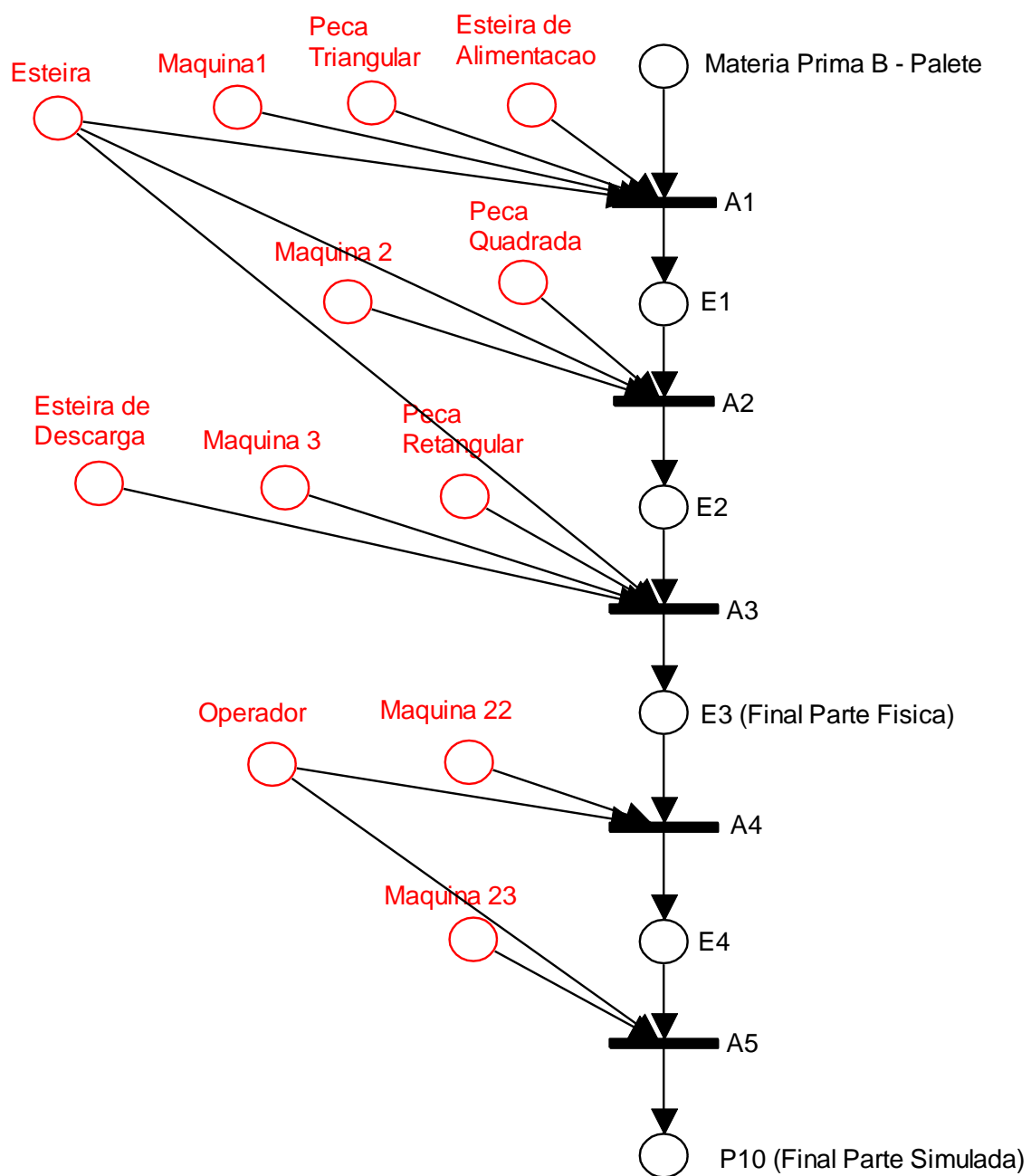


Figura 8.10 Modelo do roteiro de fabricação do produto P10.

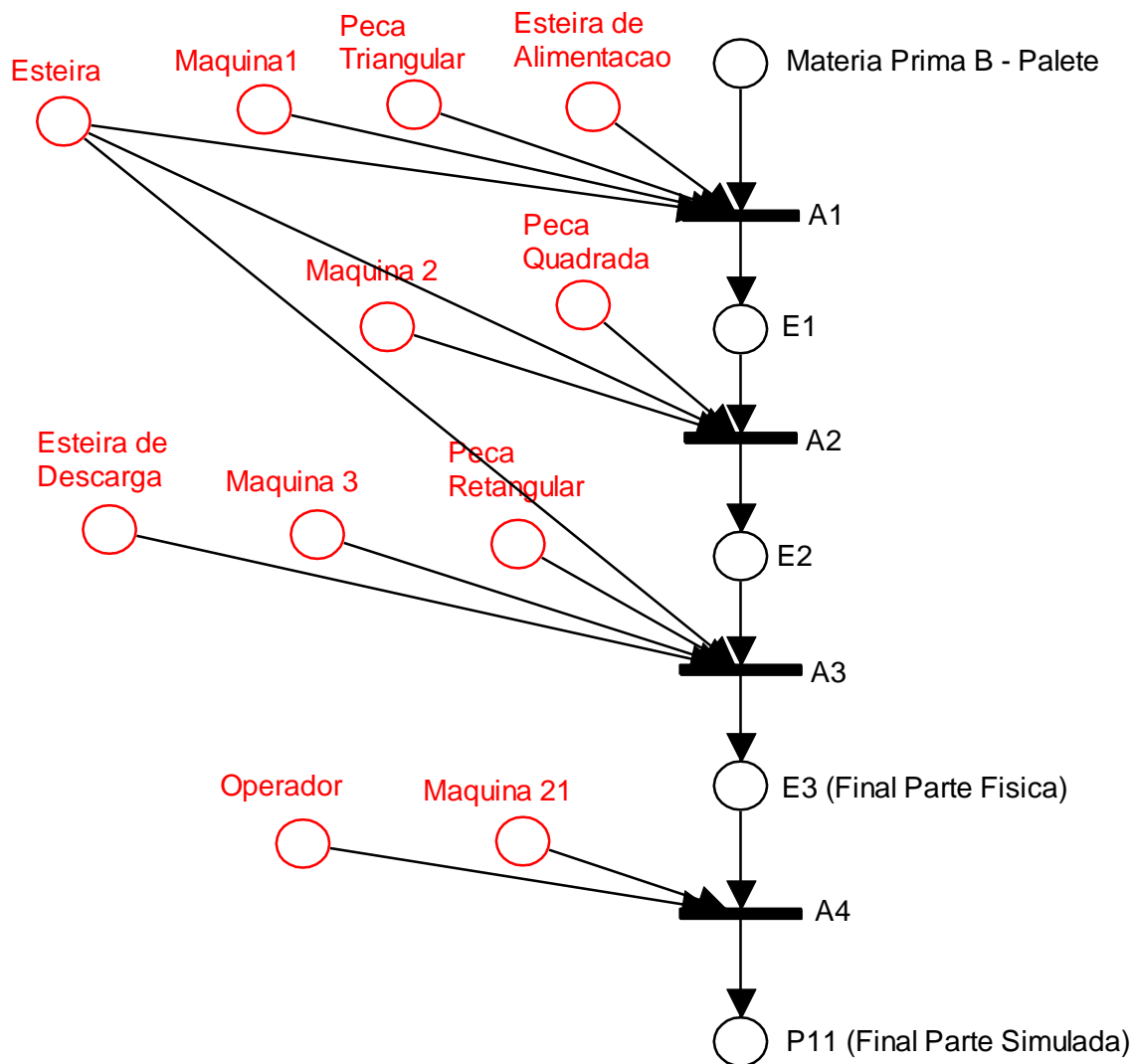


Figura 8.11 Modelo do roteiro de fabricação do produto P11.

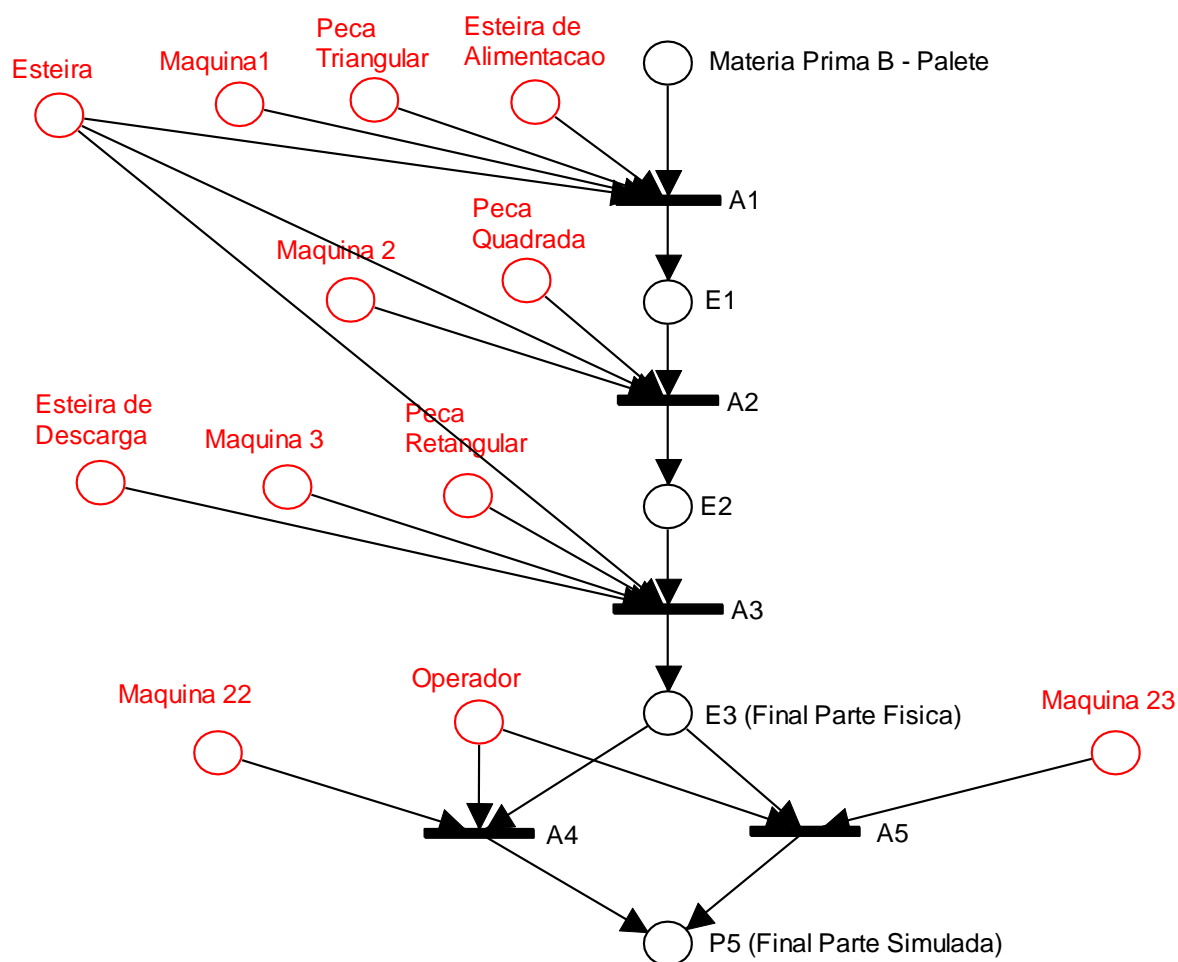


Figura 8.12 Modelo do roteiro de fabricação do produto P12.

Já os produtos da família C, que consiste do produto P13 ao P18, são apresentados a seguir:

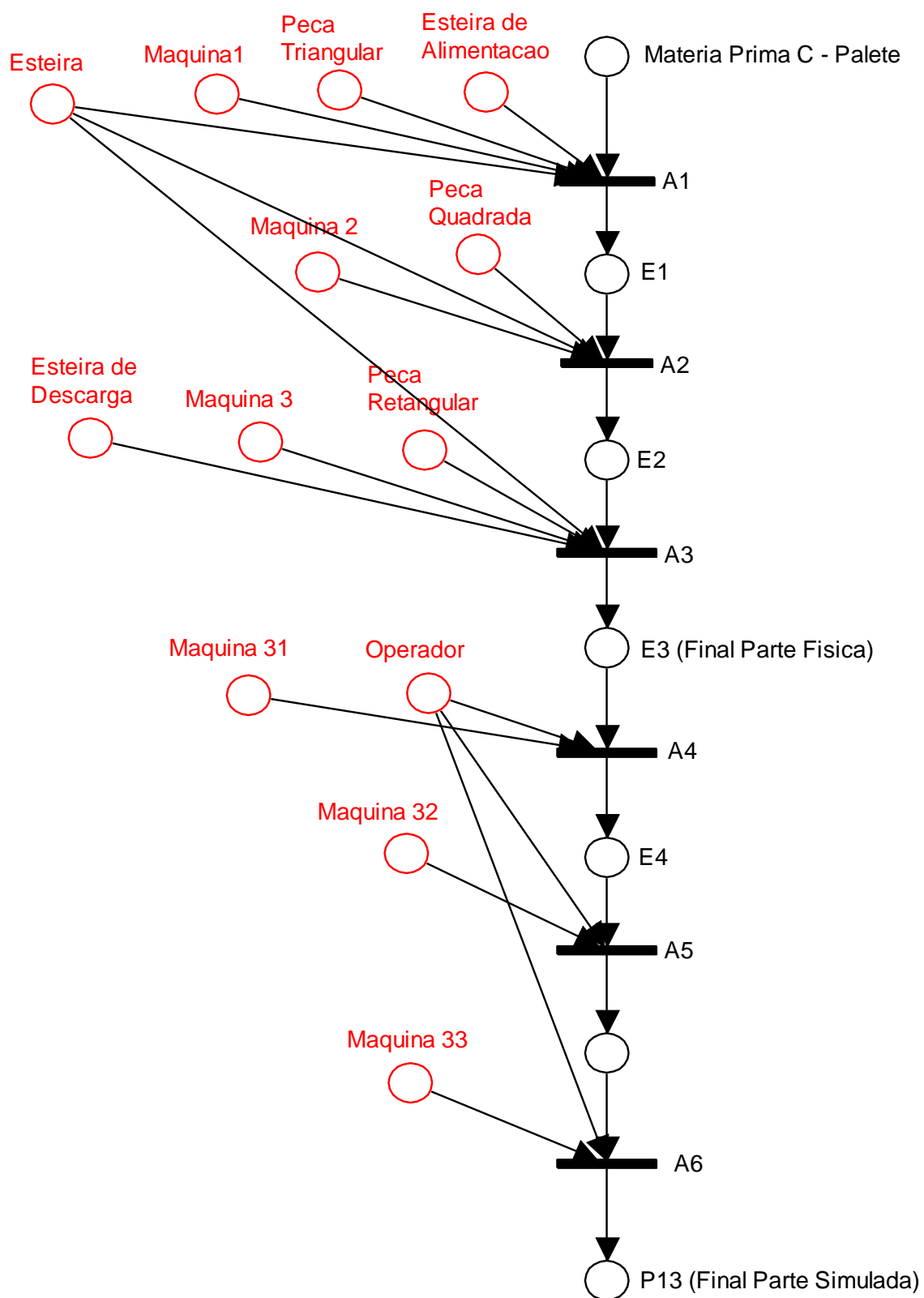


Figura 8.13 Modelo do roteiro de fabricação do produto P13.

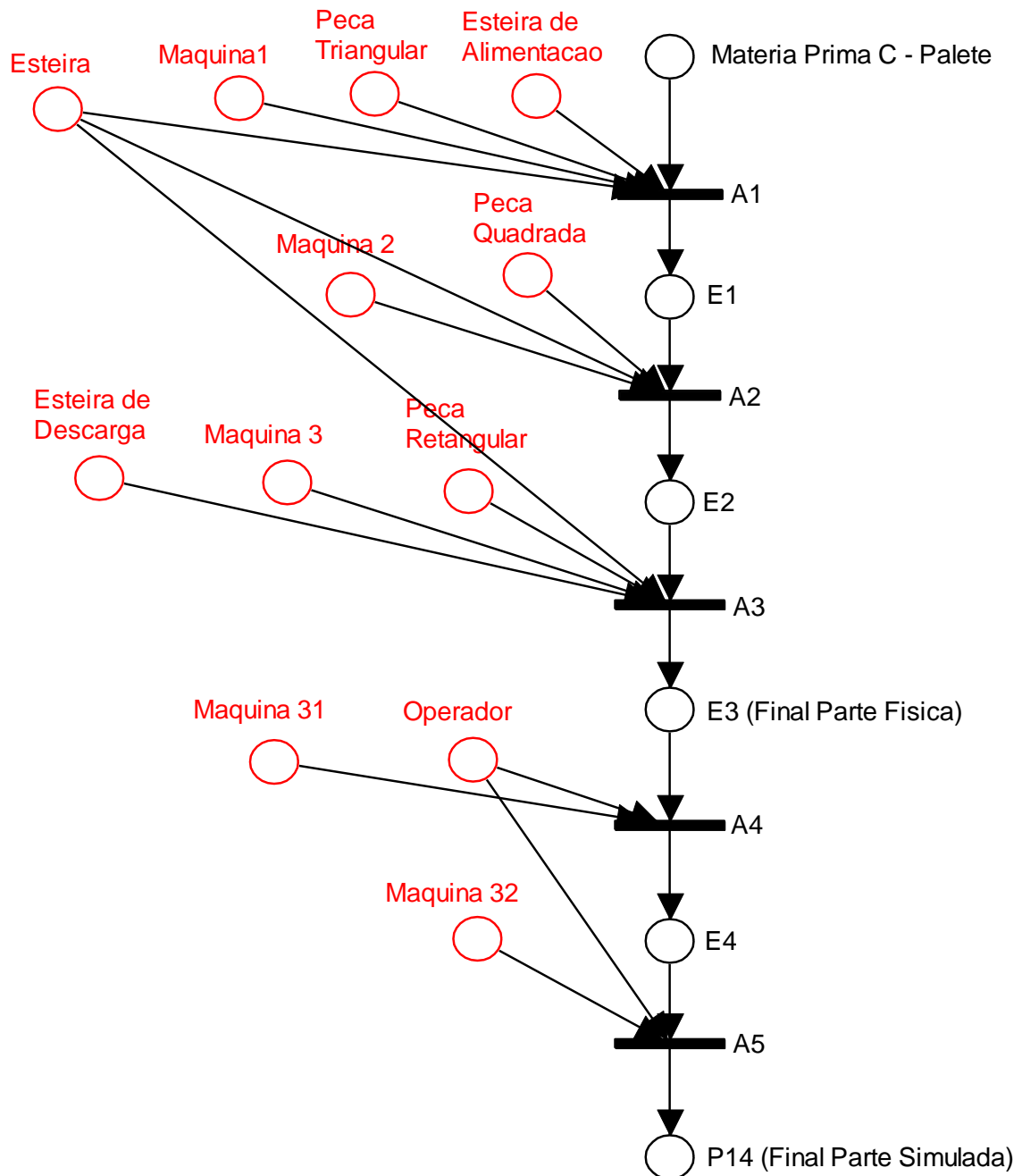


Figura 8.14 Modelo do roteiro de fabricação do produto P14.

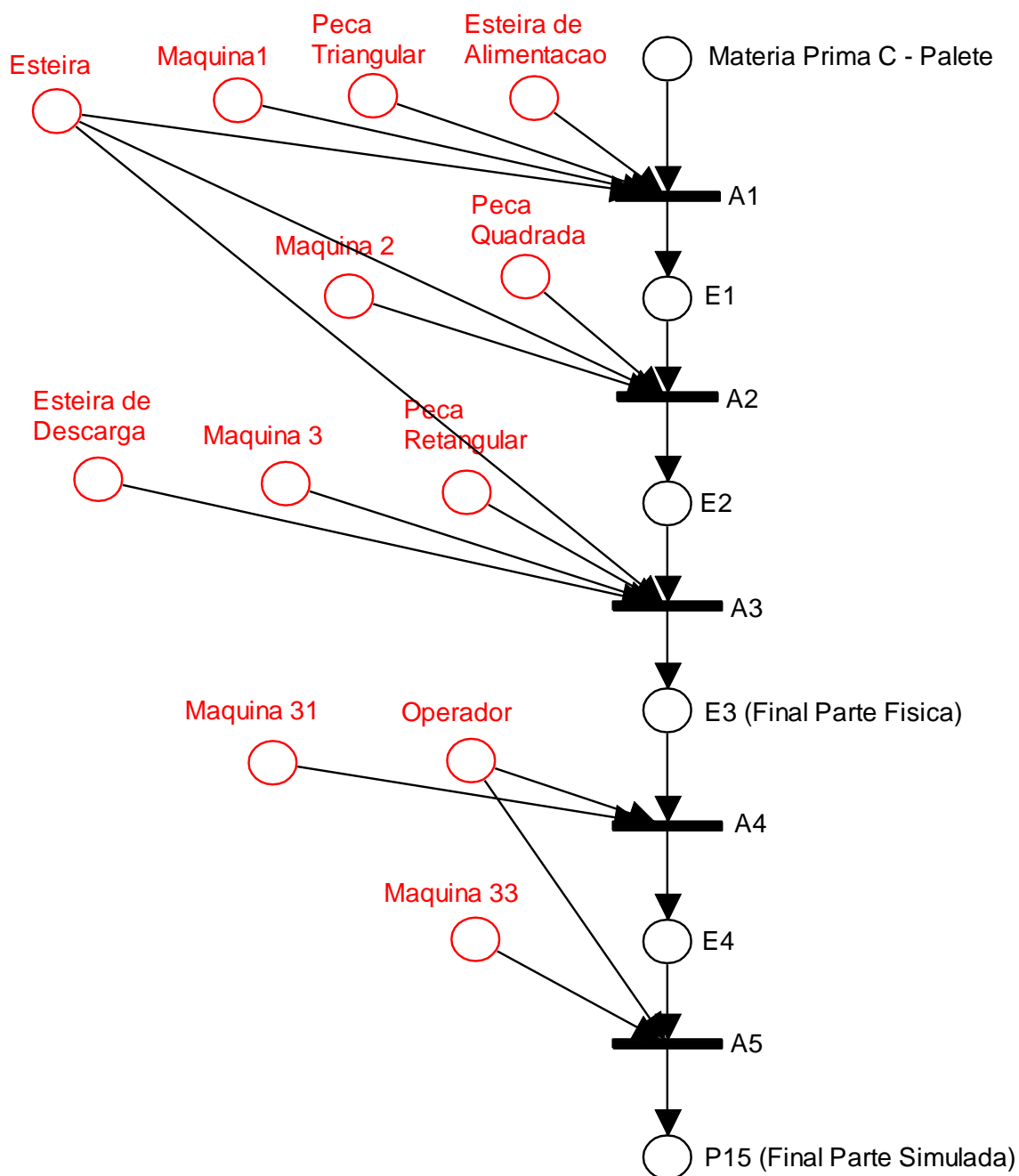


Figura 8.15 Modelo do roteiro de fabricação do produto P15.

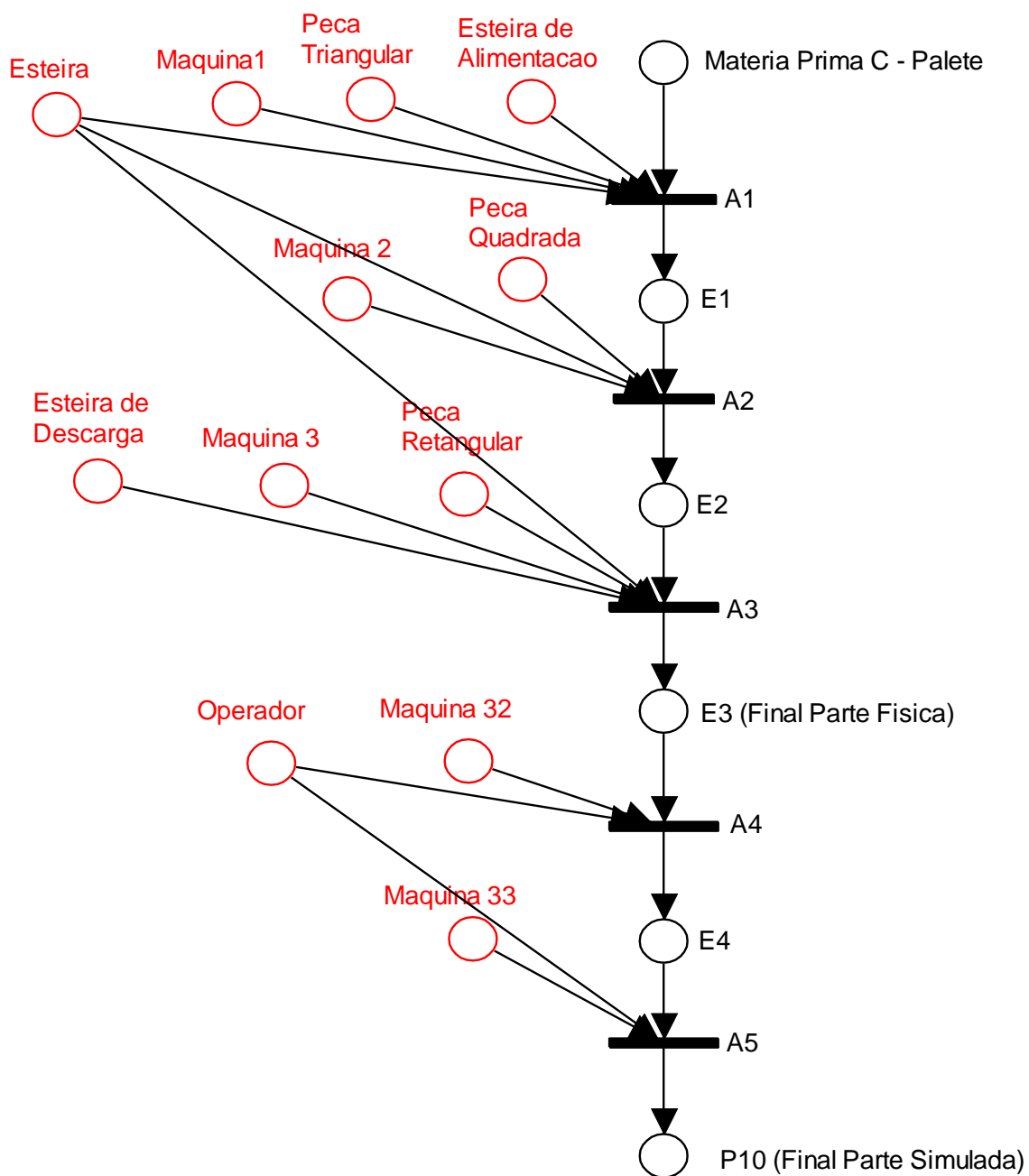


Figura 8.16 Modelo do roteiro de fabricação do produto P16.

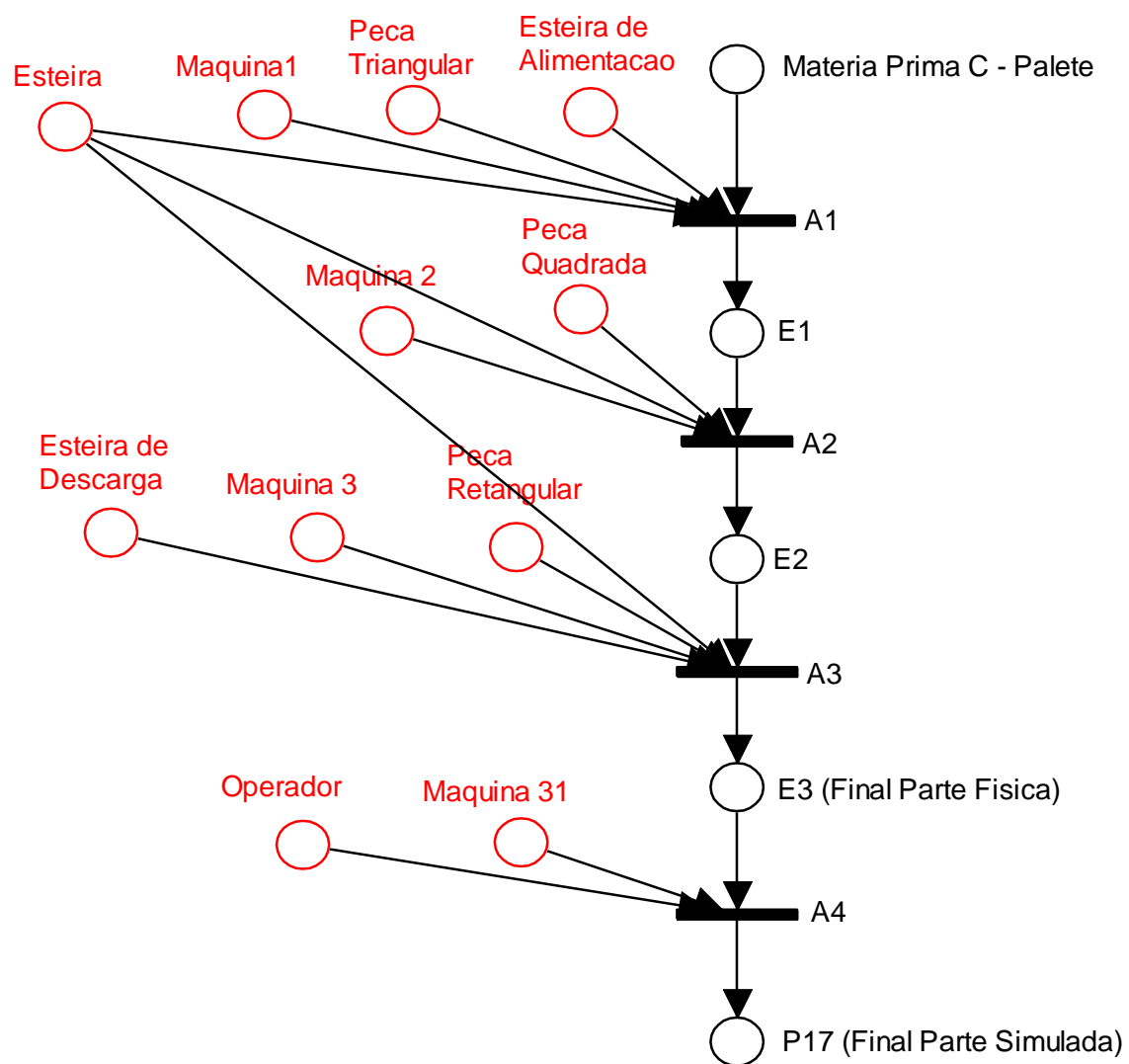


Figura 8.17 Modelo do roteiro de fabricação do produto P17.

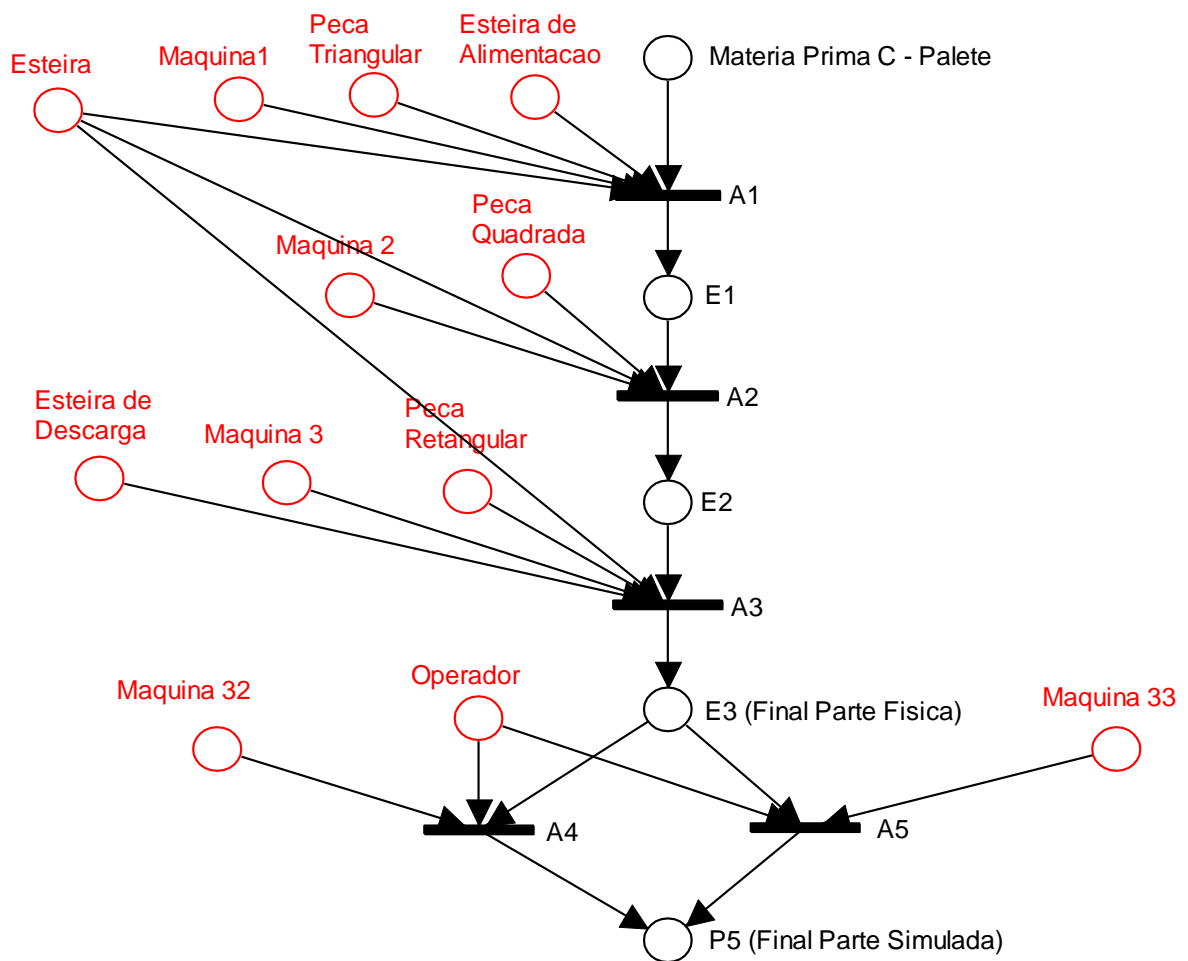


Figura 8.18 Modelo do roteiro de fabricação do produto P18.