Um framework para simulação do fluxo de peças em sistemas flexíveis de fabricação

Ernesto Massaroppi Junior

Orientador: ***Prof. Dr. Paulo Cesar Masiero***

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional.

"VERSÃO REVISADA APÓS A DEFESA"

Data da Defesa:	17/02/2003
Visto do Orientador:	

USP – São Carlos
Abril/2003

Um framework para simulação do fluxo de peças em sistemas flexíveis de fabricação

Ernesto Massaroppi Junior

A meu pai Ernesto
que não pode presenciar a conclusão deste trabalho
A minha esposa Marli e minhas filhas Larissa e Raquel
que tantas vezes tiveram um marido/pai ausente

Agradecimentos

Ao meu orientador Prof. Dr. Paulo Cesar Masiero pelas diretrizes que me guiaram no desenvolvimento desta tese e, principalmente, pelo incentivo e amizade.

Aos membros da banca do exame de qualificação, Prof. Dr. José C. Maldonado, Profa. Dra. Selma S.S. Melnnikoff e Prof. Dr. Antonio F. Prado, e da defesa da proposta de tese, Profa. Dra. Solange O. Resende, Prof. Dr. Ivan L.M. Ricarte e Prof. Dr. Fernão S.R. Germano, pelas críticas e sugestões que enriqueceram este trabalho.

Ao diretor da Escola de Engenharia de São Carlos, Prof. Dr. Eugenio Foresti, e aos chefes do Departamento de Engenharia Mecânica, Prof. Dr. Rosalvo T. Ruffino, Prof. Dr. Mario F. Mucheroni e Prof. Dr. Eduardo Vila G. F°, pelo apoio institucional que possibilitou uma maior dedicação de minha parte nos momentos mais difíceis.

A amiga Rosana T.V. Braga pelo auxílio no levantamento bibliográfico e pelas discussões em que muito aprendi sobre temas abordados nesta tese.

Aos professores e funcionários do Departamento de Engenharia Mecânica pela colaboração, amizade e incentivo.

A minha esposa e filhas, por tudo ao longo desses anos.

A todos que direta ou indiretamente colaboraram na execução deste trabalho.

ÍNDICE

Lista de Figuras ..... **vi**

Lista de Tabelas ..... **ix**

Resumo ..... **x**

Abstract ..... **xi**

1 Introdução **1**

1.1 Contextualização 1
1.2 Motivação 3
1.3 Objetivos 5
1.4 Organização 5

2 Padrões e *Frameworks* **7**

2.1 Considerações Iniciais 7
2.2 Padrões 7
2.3 Subsistema de Ponto Variável 11
2.4 *Frameworks* 12
2.4.1 Histórico 13
2.4.2 Classificação 14
2.4.2.1 Classificação de Acordo com o Escopo 14
2.4.2.2 Classificação de Acordo com a Forma de Reuso 15
2.4.3 Desenvolvimento de Aplicações Baseada em *Frameworks* 17
2.4.3.1 Desenvolvimento Baseado em Pontos Variáveis 18
2.4.3.2 Desenvolvimento Baseado em Generalização Sistemática 18
2.4.3.3 Desenvolvimento Proposto por Bosch 19
2.4.4 Uso e Aprendizagem de *Frameworks* 20
2.4.5 Evolução e Manutenção de *Frameworks* 21
2.5 Comparação entre Formas de Reuso 21
2.6 Considerações Finais 22

3 O Domínio da Manufatura **23**

3.1 Considerações Iniciais 23
3.2 Histórico 26
3.3 Conceitos Básicos 27
3.3.1 Fluxo de Material 27
3.3.2 Manipulação como Função do Fluxo de Material 28
3.3.3 Subfunções da Manipulação 29
3.3.3.1 Armazenar 30
3.3.3.2 Movimentar 30
3.3.3.3 Assegurar 31
3.4 Equipamentos e Dispositivos de Manipulação 31
3.4.1 Dispositivos para Armazenar 32
3.4.2 Equipamentos para Movimentar 34
3.4.3 Dispositivos para Assegurar 34
3.4.3.1 Placa de Fixação 34

3.4.3.2 Garra ... 36
3.5 Simulação da Função Manipulação ... 38
3.5.1 Simulação da Subfunção Armazenar ... 40
3.5.2 Simulação da Subfunção Movimentar ... 40
3.5.3 Simulação da Subfunção Assegurar ... 41
3.5.3.1 Simulação da Fixação com Placas ... 41
3.5.3.2 Simulação da Fixação com Garras ... 42
3.5.4 Resumo Crítico da Simulação da Manipulação ... 43
3.6 *Frameworks* para Manufatura ... 44
3.6.1 *Framework* OSEFA ... 46
3.6.2 *Framework* Baseado na Linguagem de Padrões G++ ... 52
3.6.3 Análise Comparativa dos *Frameworks* ... 57
3.7 Considerações Finais ... 58

4 Um Método para Simulação do Fluxo de Peças em Sistemas Produtivos ... 60

4.1 Considerações Iniciais ... 60
4.2 Informações Básicas ... 61
4.2.1 Descrição dos Equipamentos ... 61
4.2.2 Descrição da Interação entre os Equipamentos ... 63
4.2.3 Descrição dos Dispositivos ... 64
4.2.3.1 Placa de Fixação ... 64
4.2.3.2 Garra ... 66
4.2.3.3 Palete de Transporte ... 68
4.2.4 Descrição dos Materiais para a Produção da Peça ... 71
4.2.5 Relacionamento entre Equipamentos e Dispositivos ... 73
4.2.6 Relacionamento entre Equipamentos e Materiais ... 74
4.3 Procedimento para Simulação do Fluxo da Peça ... 74
4.3.1 Determinação das Rotas ... 75
4.3.2 Simulação das Rotas ... 79
4.3.3 Simulação dos Equipamentos ... 81
4.3.3.1 Depósito de Material em Bruto ... 81
4.3.3.2 Depósito de Peça em Bruto ... 82
4.3.3.3 Depósito de Palete ... 82
4.3.3.4 Serra ... 84
4.3.3.5 Robô ... 85
4.3.3.6 Torno ... 87
4.3.3.7 Transportadores ... 88
4.3.4 Simulação das Transferências entre Equipamentos ... 89
4.3.4.1 Depósito de Palete para Transportador de Palete ... 89
4.3.4.2 Transferências com o Transportador de Material em Bruto ... 89
4.3.4.3 Robô para Torno ... 89
4.3.4.4 Robô para Depósito de Palete ... 91
4.3.4.5 Outras Transferências com o Robô ... 93
4.3.5 Simulação das Conexões ... 93
4.3.5.1 Depósito de Material em Bruto para Serra ... 93
4.3.5.2 Serra para Depósito de Palete ... 94
4.3.5.3 Depósito de Peça em Bruto para Depósito de Palete ... 94
4.3.5.4 Depósito de Palete para Depósito de Palete ... 94
4.3.5.5 Depósito de Palete para Torno ... 95
4.3.5.6 Outras Conexões para o Torno ... 96

4.3.5.7 Torno para Depósito de Palete ... 96
4.3.5.8 Conexões Consecutivas ... 97
4.3.6 Simulação Inversa das Rotas ... 97
4.4 Análise e Escolha dos Resultados ... 98
4.5 Considerações Finais ... 100

5 Construção de um *Framework* para Simulação do Fluxo de Peças ... 101

5.1 Considerações Iniciais ... 101
5.2 Arquitetura do *Framework* ... 102
5.2.1 Arquitetura Básica ... 102
5.2.2 Características do Usuário ... 103
5.2.3 Arquitetura Detalhada ... 106
5.3 Integração entre Módulos ... 108
5.4 Persistência de Pontos Variáveis ... 110
5.4.1 Generalização de Tratamento da Chave ... 113
5.4.2 Implementação Flexível de Classes Específicas ... 114
5.4.3 Criação Flexível de Objetos da Aplicação ... 115
5.5 Projeto do *Framework* ... 117
5.5.1 Aspectos Gerais do Projeto ... 118
5.5.2 Transformações dos Relacionamentos de Agregação ... 119
5.5.3 Transformações para Generalização dos Dispositivos ... 120
5.5.4 Transformações para Generalização das Rotas, Seções e Conexões ... 122
5.6 Comparação entre este *Framework* e aqueles para controle de SFF ... 124
5.7 Instanciação do *Framework* ... 126
5.7.1 Instanciação de uma aplicação a partir do *framework* ... 126
5.7.2 Instanciação de uma aplicação semelhante a outra já realizada ... 130
5.7.3 Instanciação de uma aplicação distinta de outras já realizadas ... 132
5.8 Implementação ... 132
5.9 Considerações Finais ... 134

6 Avaliação do Sistema Proposto ... 135

6.1 Considerações Iniciais ... 135
6.2 Uso do *Framework* ... 135
6.2.1 Descrição do Sistema Flexível de Fabricação ... 135
6.2.2 Instanciação de uma Aplicação ... 136
6.2.3 Dados de Entrada para a Simulação ... 138
6.2.4 Resultados da Simulação ... 138
6.3 Avaliação do Método de Simulação e do *Framework* ... 154
6.4 Considerações Finais ... 155

7 Conclusões ... 156

7.1 Considerações Iniciais ... 156
7.2 Contribuições ... 156
7.3 Trabalhos Futuros ... 158

Anexo ... 159

Referências Bibliográficas ... 162

Lista de Figuras

Figura 2.1 – Padrão Herança de Interface (Schmid, 1997; 1999) 9
Figura 2.2 – Padrão Cadeia de Responsabilidade (Gamma *et al*., 1995) 10
Figura 2.3 – Padrão Método de Fabricação (Gamma *et al*., 1995) 10
Figura 2.4 – Subsistema de ponto variável (Schmid, 1997; 1999)..... 12
Figura 2.5 – Classificação de pontos variáveis de acordo com a forma de reuso 16
Figura 2.6 – Desenvolvimento baseado em pontos variáveis (Pree, 1995) 18
Figura 2.7 – Desenvolvimento baseado em generalização sistemática (Schmid, 1999) 19
Figura 3.1 – Domínio da manufatura..... 23
Figura 3.2 – Grau de orientação e posicionamento de um corpo 29
Figura 3.3 – Divisões do fluxo de material 32
Figura 3.4 – Equipamentos para movimentar (VDI2860, 1990)..... 35
Figura 3.5 – Equipamentos e dispositivos de manipulação 39
Figura 3.6 – Diagrama de classes de uma célula de manufatura (Schmid, 1996b)..... 47
Figura 3.7 – Lógica de uma aplicação específica (Schmid, 1996b)..... 47
Figura 3.8 – Diagrama de pontos variáveis estruturado de acordo com o nível de abstração (Schmid, 2000a) 48
Figura 3.9 – Estrutura de camadas do *framework* OSEFA (Schmid, 2000a) 49
Figura 3.10 – Diagrama de classes do *framework* OSEFA..... 51
Figura 3.11 – Linguagem de padrões G++ (Aarsten *et al*., 2000)..... 53
Figura 3.12 – Modelo de referência da NBS para sistemas flexíveis de fabricação (McLean *et al*. *apud* Brugali *et al*., 2000) 55
Figura 3.13 – Diagrama de classes do *framework* (Brugali *et al*., 2000) 57
Figura 4.1 – Relacionamento entre o Sff, a célula e o equipamento 62
Figura 4.2 – Detalhamento dos equipamentos 63
Figura 4.3 – Descrição da conexão de manipulação e da conexão de transporte..... 64
Figura 4.4 – Representação esquemática de uma placa de três castanhas 65
Figura 4.5 – Descrição geométrica da placa de três castanhas..... 65
Figura 4.6 – Modelo da placa de três castanhas 66
Figura 4.7 – Representação esquemática de uma garra de duas pinças..... 67
Figura 4.8 – Descrição geométrica da garra de duas pinças 68
Figura 4.9 – Modelo da garra de duas pinças..... 68
Figura 4.10 – Representação esquemática do apoio sobre longarinas..... 69
Figura 4.11 – Descrição geométrica da longarina para apoio horizontal 70
Figura 4.12 – Descrição geométrica da longarina para apoio vertical 71

Figura 4.13 – Modelo do palete de longarinas 72
Figura 4.14 – Modelo da peça em bruto e do material em bruto 73
Figura 4.15 – Relacionamento entre equipamento e dispositivo..... 74
Figura 4.16 – Relacionamento entre materiais e depósitos..... 74
Figura 4.17 – Fluxos principais de peças com simetria rotacional..... 75
Figura 4.18 – Subdivisão das seções principais para equipamentos em células distintas 76
Figura 4.19 – Subdivisão das seções e subseções em conexões 78
Figura 4.20 – Relacionamento entre rota, seção e conexão 79
Figura 4.21 – Simulação genérica do fluxo de peças 80
Figura 4.22 – Esquema de uso dos equipamentos, transferências entre eles e processos 81
Figura 4.23 – Simulação do depósito de material em bruto..... 82
Figura 4.24 – Simulação do depósito de peça em bruto 83
Figura 4.25 – Dados da armazenagem em longarinas 83
Figura 4.26 – Simulação do depósito de palete..... 85
Figura 4.27 – Simulação da serra 85
Figura 4.28 – Dados da fixação com a garra..... 86
Figura 4.29 – Simulação do robô 87
Figura 4.30 – Dados da fixação com a placa 87
Figura 4.31 – Simulação do torno 88
Figura 4.32 – Região ocupada pela placa durante a fixação 90
Figura 4.33 – Simulação da transferência entre o robô e o torno..... 91
Figura 4.34 – Simulação da transferência entre o robô e o depósito de palete 92
Figura 5.1 – Arquitetura básica do *framework*..... 102
Figura 5.2 – Menu principal do Módulo de Configuração 104
Figura 5.3 – Janela de diálogo da longarina prismática para apoio de peças na horizontal .. 105
Figura 5.4 – Informações gráficas do Módulo de Escolha 107
Figura 5.5 – Arquitetura detalhada do *framework*..... 108
Figura 5.6 – Comunicação com o banco de dados 110
Figura 5.7 – Modelo básico de uma aplicação com generalização 111
Figura 5.8 – Generalização da chave..... 114
Figura 5.9 – Implementação de classes específicas..... 115
Figura 5.10 – Criação flexível de objetos da aplicação 116
Figura 5.11 – Generalização dos tipos de células..... 119
Figura 5.12 – Generalização dos tipos de placas de fixação..... 120
Figura 5.13 – Generalização dos tipos de garras e pinças 121

Figura 5.14 – Generalização dos tipos de paletes e longarinas 121
Figura 5.15 – Generalização da identificação das rotas..... 123
Figura 6.1 – Sistema Integrado de Fabricação e Montagem (IFMS) 137
Figura 6.2 – Parafuso especial..... 138
Figura 6.3 – Rota de produção da peça..... 139
Figura 6.4 – Disposição da peça em bruto na mesa da Serra Kasto 141
Figura 6.5 – Fixação da peça em bruto com a garra do robô 142
Figura 6.6 – Armazenagem da peça em bruto no palete de longarinas 143
Figura 6.7 – Processo de descarga da serra Kasto..... 143
Figura 6.8 – Fixação da peça em bruto com a garra do robô Kuka..... 145
Figura 6.9 – Fixação da peça em bruto com a placa do torno Monforts 146
Figura 6.10 – Processo de alimentação do torno Monforts..... 146
Figura 6.11 – Geometria da peça semipronta..... 147
Figura 6.12 – Fixação da peça semipronta com a placa do torno Monforts após a primeira usinagem 148
Figura 6.13 – Fixação da peça semipronta com a garra do robô Kuka 148
Figura 6.14 – Segunda fixação da peça semipronta com a placa do torno Monforts..... 149
Figura 6.15 – Processo de refixação da peça semipronta no torno Monforts 149
Figura 6.16 – Fixação da peça pronta com a placa do torno Monforts 150
Figura 6.17 – Fixação da peça pronta com a garra do robô Kuka..... 152
Figura 6.18 – Armazenagem da peça pronta sobre o palete de longarinas..... 152
Figura 6.19 – Processo de descarga do torno Monforts..... 153

Lista de Tabelas

Tabela 2.1 – Classificação dos padrões (Schmid, 1999) 12
Tabela 3.1 – Transformações do diagrama de classes do *framework* OSEFA 52
Tabela 3.2 – Linguagem de padrões G++ (Aarsten *et al.*, 2000) 54
Tabela 5.1 – Tamanho do *framework* e da aplicação 134

Resumo

Um processo de modelagem e o modelo de classes de um *framework* para a simulação do fluxo de peças em sistemas flexíveis de fabricação são apresentados. O método de simulação implementado no *framework* é baseado na modelagem geométrica e funcional dos dispositivos que entram em contato direto com a peça durante o seu fluxo pelo sistema produtivo. Um índice que qualifica a utilização de cada tipo de dispositivo foi definido, o qual é utilizado para a escolha automática da melhor forma de realização do fluxo da peça no sistema de fabricação. No processo de desenvolvimento do *framework* propôs-se uma associação de padrões para efetuar a persistência em um banco de dados relacional das informações relativas aos pontos variáveis do *framework*. Essa associação de padrões é reutilizada em outras partes do *framework* e poderia ser útil em outros domínios, como um novo padrão de projeto para persistência de dados. O *framework* proposto é instanciado para simular um sistema produtivo real e sua utilização resulta no planejamento macroscópico das operações que compõem a folha de processo da peça.

Abstract

A modeling process and the class model of a framework for the simulation of the part flow in flexible manufacturing systems are presented. The simulation method implemented in the framework is based on the geometric and functional modeling of the devices that enter in direct contact with the part during its flow through the production system. An index that qualifies the use of each device type was defined and is used as the basis for choosing automatically the best way of conveying the part through the production system. In the framework development process an association of patterns to make the persistence in a relational database of informations related to the framework hot spots is proposed. This association of patterns is reused in other parts of the framework and could be reused in other domains as a new design pattern for data persistency. The framework presented is instantiated to simulate a real production system and its use results in the macroscopic planning of the operations that comprise the part's work sheet.

1 Introdução

1.1 Contextualização

O reuso sistemático de software foi introduzido com o primeiro compilador COBOL há cerca de quarenta anos, quando o conceito de bibliotecas foi desenvolvido. As bibliotecas permitiam que várias subrotinas pré-compiladas pudessem ser reutilizadas ao serem ligadas a um programa. A visão dos pesquisadores em COBOL era que em poucos anos bibliotecas-padrão seriam desenvolvidas para diferentes domínios de aplicações, possibilitando que os desenvolvedores escrevessem programas de alto nível que reutilizassem em alto grau a lógica e a funcionalidade disponíveis nas bibliotecas. Entretanto, isto não aconteceu com o grau de intensidade imaginado pelos pesquisadores da época. Poucos domínios, em que o reuso era baseado em bibliotecas, tiveram sucesso. Entre eles destacam-se a análise numérica, no qual um grande número de bibliotecas com rotinas escritas em FORTRAN estão disponíveis e são utilizadas em muitos projetos, e interfaces gráficas com o usuário. O motivo principal para o fracasso do reuso baseado em bibliotecas é a dificuldade de encapsular funcionalidade de alto nível em subrotinas. Bibliotecas de subrotinas impõem a separação entre o código e os dados que compõem um programa (Brugali, 1998).

O paradigma da orientação a objetos, iniciado com o surgimento da linguagem Simula em 1967 e popularizado com o desenvolvimento da linguagem Smalltalk (Goldberg & Robson, 1983) nos anos oitenta, introduziu o conceito de bibliotecas de classes como uma nova técnica de reuso. Programas orientados a objetos são vistos como objetos que encapsulam o estado e o comportamento e que se comunicam por meio de mensagens. O comportamento do sistema no todo é o resultado da interação e da colaboração entre os objetos (Lippman & Lajoie, 1998). O desenvolvedor implementa classes que são descrições do estado e do comportamento de muitos objetos similares e cria objetos como instâncias dessas classes.

A abordagem de biblioteca de classes não teve um sucesso maior que as bibliotecas de subrotinas. A maioria das bibliotecas de classes que obtiveram sucesso consiste de classes simples do tipo *container* ou coleção, que são nada mais do que implementações orientadas a objetos de estruturas básicas de dados (Brugali, 1998). O fracasso do reuso baseado em biblioteca de classes deve-se parcialmente ao fato de que elas normalmente restringem o polimorfismo a trabalhar sobre objetos especializados de uma classe da biblioteca e de que cada biblioteca disponibiliza uma versão diferente e incompatível da classe a ser

especializada. Outro fator importante é que o tamanho e a complexidade dos projetos cresceram demasiadamente desde o surgimento das bibliotecas de subrotinas e, conseqüentemente, o projeto e a arquitetura do software passaram a representar uma parcela maior do esforço de desenvolvimento.

Apesar do surgimento de diversos métodos e notações para o desenvolvimento e documentação de projetos e arquiteturas, a reutilização do projeto e da arquitetura de forma sistemática não foi possível. Essa reutilização é mais difícil que o reuso de código, uma vez que um projeto não é por si só um produto diretamente reutilizável (Bosch *et al*., 1999). O projeto precisa ser compreendido por aquele que pretende reutilizá-lo para que as adaptações necessárias possam ser realizadas. O desafio era encontrar um meio de representar não só o projeto em si, mas também o conhecimento e a experiência nele embutidos.

Isso se tornou possível com a introdução de padrões de software no início dos anos noventa (Gamma *et al*., 1995) e, mais recentemente, de subsistemas de pontos variáveis (Schmid, 1997; 1999), utilizados para captar e documentar a experiência dos desenvolvedores de softwares. Com o aparecimento dos *frameworks* de software orientados a objetos[1] tornou-se possível a reutilização de grandes estruturas de um determinado domínio (Fayad & Schmidt, 1997). Com isso, famílias de aplicações semelhantes, mas não iguais, puderam ser desenvolvidas a partir de um mesmo *framework*.

Primeiramente, surgiram os *frameworks* gerais, desenvolvidos para domínios de software básico como, por exemplo, interface com o usuário (Bosch *et al*., 1999). Nos últimos anos existe uma tendência cada vez maior de desenvolver-se *frameworks* para domínios específicos, como sistemas bancários, sistemas de telecomunicação, gestão de negócios, medicina e aviação, entre outros (Fayad & Johnson, 2000). Em particular, o ambiente de manufatura necessita reagir mais rapidamente às mudanças do mercado consumidor, principalmente após a globalização. Se por um lado, as empresas manufatureiras necessitam de investimentos em equipamentos cada vez mais sofisticados, por outro, esses equipamentos estão cada vez mais integrados com a eletrônica digital e necessitam de softwares de controle. Os *frameworks* têm-se mostrado uma forma eficiente e rápida de desenvolvimento e manutenção de aplicações também na área de integração da manufatura (Aarsten *et al*., 1995; Schmid, 1995).

---

[1] No escopo deste trabalho será utilizado apenas o termo *framework* como sinônimo de *framework* de software orientado a objetos.

Em contrapartida, *frameworks* são complexos para construir, entender e utilizar. Os métodos para o desenvolvimento de *frameworks* são ainda incipientes (Bosch *et al*., 1999). A sua documentação enfatiza o projeto do *framework* e não a sua utilização para o desenvolvimento de aplicações.

A tendência futura é o desenvolvimento de pesquisas que estudem a relação entre *frameworks* e linguagens de padrões (Brugali & Menga, 1999) e a integração de *frameworks* para o desenvolvimento de aplicações em fronteiras entre domínios (Mattsson *et al*.,1999).

1.2 Motivação

Uma das motivações para o desenvolvimento deste projeto de pesquisa é proveniente de uma série de atividades, desencadeadas uma pela outra, realizadas pelo autor ao longo do tempo, em grande parte antes do início da pesquisa aqui proposta. Essas atividades são aqui descritas sucintamente, com o objetivo de mostrar como se identificou a necessidade de realização desta pesquisa.

O autor iniciou suas atividades no domínio da manufatura realizando a integração entre um sistema CAD (*Computer Aided Design*) e um sistema de Programação CN (Comando Numérico). Os dados geométricos, tanto da peça a ser usinada como dos componentes da máquina-ferramenta (sistema de fixação da peça, ferramentas, etc.) foram criados no sistema CAD e posteriormente utilizados no sistema de programação CN para a apresentação gráfica e simulação da usinagem.

Posteriormente, o autor desenvolveu um sistema simples, baseado na atividade anterior, para geração de programas CN, que leva em consideração restrições dos sistemas periféricos da máquina-ferramenta. Com essa atividade procurou-se poupar um trabalho demorado e de alto custo, que é realizado após a geração dos programas CN, para evitar incompatibilidades entre a opção realizada para a usinagem da peça e para o emprego dos equipamentos periféricos à máquina, que também realizam operações sobre a peça.

Ao término da atividade, o autor verificou que a abordagem, só do ponto de vista do programador CN não era a melhor forma de tratamento do problema. Então, ampliou-se a proposta para o desenvolvimento de um sistema capaz de simular todo o fluxo de material desde a escolha do material em bruto em um depósito de matéria prima até a entrega da peça pronta no depósito correspondente. No início do desenvolvimento verificou-se que a simples produção de código computacional não levaria ao desenvolvimento de um sistema satisfatório. Percebeu-se a interdisciplinaridade do tema escolhido e a necessidade de

interação entre as técnicas provenientes da Engenharia Mecânica e da Engenharia de Software.

Simultaneamente, o autor iniciou o programa de doutoramento. Após a conclusão dos créditos e início da pesquisa bibliográfica para escolha de um artigo objeto do Exame de Qualificação, o autor entrou em contato com o trabalho desenvolvido por Schmid (1995; 1996a; 1996b), um *framework* para controle de uma célula flexível de fabricação. Com o maior contato do autor com padrões de software e *frameworks* durante sua preparação para o exame, surgiu a idéia do desenvolvimento de um *framework* para simular o fluxo de peças em sistemas flexíveis de fabricação, desenvolvido com base em padrões de software. Padrões de software e subsistemas de pontos variáveis são temas atuais de pesquisa na Engenharia de Software (Gamma *et al*., 1995; Pree, 1995; Schmid, 1999) e a sua utilização para o desenvolvimento de *frameworks* deve ser explorada (Fayad & Johnson, 2000). Desta forma surgiu a motivação para o desenvolvimento desta tese.

Além das motivações pessoais, o desafio do desenvolvimento de um *framework*, um produto concebido para ser reutilizado por outros desenvolvedores e pesquisadores para aplicações muitas vezes não previstas durante a concepção do *framework*, é, acredita o autor, a maior motivação para o desenvolvimento desta tese.

O desenvolvimento de um *framework* está associado a um grande esforço na compreensão e na sistematização das informações que caracterizam um domínio. Nos últimos anos, o desenvolvimento de *frameworks* nas diversas áreas do conhecimento tem sido tema de pesquisas. Em particular, no domínio da manufatura encontram-se trabalhos em vários de seus subdomínios e também diversos subdomínios ainda não explorados. A possibilidade de ser o primeiro a desenvolver um *framework* em um destes nichos é também uma motivação para os pesquisadores.

O desenvolvimento de *frameworks* é uma tendência atual não só em pesquisas mas também em aplicações comerciais. O aumento da produtividade e da eficiência no desenvolvimento de aplicações a partir de um *framework* pode ser financeiramente contabilizado pelos seus usuários, que passaram a disponibilizar recursos para a sua aquisição. As vantagens econômicas obtidas com o desenvolvimento de *frameworks* tem sido umas das motivações para o aumento do número desses produtos.

1.3 Objetivos

Um dos objetivos desta tese é o desenvolvimento de um método que simule a produção de uma peça típica da indústria metal-mecânica em sistemas flexíveis de fabricação com base no emprego dos dispositivos utilizados para a manipulação da peça durante o seu fluxo pelo sistema. O objetivo da simulação é determinar as operações a que a peça deve ser submetida durante a sua produção e como ela é fixada no equipamento que realiza a operação. O método de simulação deve garantir a compatibilidade entre todas as operações realizadas sobre a peça durante o processo produtivo.

Outro objetivo desta pesquisa é a generalização de parte do domínio da manufatura e do método de simulação proposto para a criação de um modelo conceitual orientado a objetos do subdomínio de fluxo de material em sistemas de fabricação que possa ser utilizado para a implementação computacional do método proposto. A generalização do modelo é importante para que ele possa ser reutilizado em estudos semelhantes do domínio e em desenvolvimentos de aplicações similares, mas não idênticas, com maior rapidez e eficiência.

Também é objetivo deste trabalho o desenvolvimento um *framework* baseado em um método de generalização sistemática que abstraia os aspectos flexíveis do subdomínio de fluxo de material. Para avaliar o método e o *framework* desenvolvido deve-se instanciar uma aplicação a partir do *framework* que modele um sistema flexível de fabricação específico. Os resultados da simulação devem ser criticamente comparados com os resultados provenientes da análise de especialistas do domínio.

1.4 Organização

Este trabalho está organizado em sete capítulos. No capítulo 2 são definidos os conceitos básicos sobre padrões de software, subsistemas de pontos variáveis e *frameworks*. O objetivo é proporcionar o embasamento teórico necessário para compreensão do trabalho de doutorado. São incluídos nesse capítulo definições e termos pertinentes ao assunto. No fim do capítulo é feita uma comparação entre os diversos conceitos apresentados, com o propósito de esclarecer as semelhanças e diferenças entre eles.

No capítulo 3 é apresentado sucintamente o domínio da manufatura e em maior detalhes os conceitos relacionados aos subdomínios do planejamento do processo e dos sistemas flexíveis de fabricação. São apresentadas com maior destaque as definições e os conceitos relacionados com o fluxo de material que ocorre nesses sistemas. Apresentam-se também alguns sistemas computacionais desenvolvidos para simular o fluxo de material nos sistemas

flexíveis de fabricação e, com destaque, dois *frameworks* para controle de sistemas flexíveis de fabricação.

No Capítulo 4 apresenta-se a proposta de um método que simula o fluxo de peça em sistemas flexíveis de fabricação. Em conjunto com o método mostra-se também o processo de modelagem e o modelo orientado a objetos resultante, em um nível de detalhamento suficiente para uma implementação computacional.

No Capítulo 5 transforma-se o modelo conceitual apresentado no Capítulo 4 no modelo de um *framework* para simular o fluxo da peça, com base em um método de generalização sistemática. Como o *framework* é desenvolvido de forma modular, apresentam-se os problemas ocorridos e as soluções adotadas para integrar os seus módulos utilizando um banco de dados relacional. Detalha-se o projeto das partes mais relevantes do *framework* para poder compará-lo com dois *frameworks* para controle de sistemas de fabricação freqüentemente presentes na literatura específica. No fim do Capítulo mostra-se como instanciar o *framework* para a geração de aplicações e a implementação realizada no escopo deste trabalho.

No Capítulo 6 mostra-se a instanciação de uma aplicação para simular um sistema flexível de fabricação real a partir do *framework*. Utiliza-se esse exemplo de uso do *framework* para avaliar tanto o método proposto como o *framework* desenvolvido para implementá-lo.

No Capítulo 7 apresentam-se as conclusões desta tese, com um resumo do trabalho realizado, as contribuições ao estado da arte e as sugestões para trabalhos futuros.

2 Padrões e *Frameworks*

2.1 Considerações Iniciais

Neste capítulo apresenta-se uma revisão bibliográfica sobre as formas de reuso de software de interesse no escopo deste trabalho, procurando mostrar o estado da arte em que elas se encontram. Inicialmente, mostram-se definições, classificações e fontes de coletâneas dos padrões de software. Em seguida, introduzem-se os conceitos de subsistemas de pontos variáveis e mostra-se o mecanismo utilizado para atingir a flexibilidade necessária. Posteriormente, é feita uma introdução aos *frameworks*, seu histórico, suas classificações e os principais conceitos referentes ao desenvolvimento, implementação, uso, aprendizagem e evolução dos *frameworks* para o desenvolvimento de aplicações. Finalmente, comparam-se as três formas de reuso com a finalidade de evidenciar suas semelhanças e suas diferenças.

Na seção 2.2 são apresentados os conceitos básicos sobre padrões. Os subsistemas de pontos variáveis que, normalmente, são implementados por meio de padrões, são apresentados na seção 2.3. A seção 2.4 mostra os conceitos relativos aos *frameworks*. Na seção 2.5 efetua-se uma comparação entre padrões, subsistemas de pontos variáveis e *frameworks*. Na seção 2.6 são apresentadas as considerações finais sobre este capítulo.

2.2 Padrões

O conceito de padrão de software é baseado no trabalho desenvolvido por Alexander *et al*. (1977) na área de arquitetura. Segundo eles "Cada padrão descreve um problema que ocorre inúmeras vezes em nosso ambiente e, então, descreve o núcleo da solução para esse problema, de tal modo que você pode usá-la milhões de vezes, sem que o faça duas vezes do mesmo modo". De forma semelhante, os padrões de software representam soluções para problemas que ocorrem inúmeras vezes no desenvolvimento de sistemas de software, podendo ser considerado com um par problema/solução (Buschmann *et al*., 1996). Um padrão é um conjunto essencial de informações instrutivas que possui um nome e que capta a estrutura essencial e o raciocínio de uma família de soluções, comprovadamente bem sucedidas, para um problema repetido que ocorre sob um determinado contexto e sob um conjunto de forças (Appleton, 1997). Ou seja, padrões descrevem soluções freqüentes que passaram pelo teste do tempo (Fayad *et al*., 1999).

O uso de padrões proporciona um vocabulário comum para comunicação entre desenvolvedores e projetistas, criando abstrações em um nível superior ao de classes e

garantindo uniformidade na estrutura do software (Gall *et al*., 1996), pois fornecem exemplos de problemas e soluções a serem diretamente utilizados, refinados ou estendidos. Além disso, os padrões atuam como blocos construtores, a partir dos quais projetos mais complexos podem ser construídos (Gamma *et al*., 1995).

Padrões de software podem referir-se a diferentes níveis de abstração no desenvolvimento de sistemas orientados a objetos. A fim de facilitar a reutilização dos padrões de software, deve-se classificá-los. Entretanto, essa classificação não é única, pois devido à natureza de alguns padrões eles podem ser diferentemente classificados. Conferências internacionais para apresentação e discussão de padrões têm ocorrido desde 1994. Uma série de livros reúne os melhores artigos apresentados nessas conferências (Coplien & Schmidt, 1995; Martin *et al*., 1998; Vlissides *et al*., 1996), classificando-os de acordo com o propósito de cada padrão. Assim, padrões arquiteturais auxiliam a especificar a estrutura fundamental dos sistemas de software. Padrões de distribuição são importantes para desenvolver sistemas distribuídos compreensíveis, gerenciáveis, adaptáveis e confiáveis. Padrões de persistência auxiliam na armazenagem e recuperação de informações e são bastante utilizados com sistema de gerenciamento de banco de dados. Padrões de interface com o usuário auxiliam o desenvolvimento de interfaces gráficas. Padrões de programação descrevem como implementar e como tomar decisões táticas e estratégicas durante o desenvolvimento de um sistema. Padrões de análise englobam o conhecimento do domínio específico da aplicação e auxiliam a solucionar os problemas típicos de análise. Padrões de processo dão soluções para os problemas da engenharia de software: desenvolvimento, controle de configuração, testes, etc. Padrões de padrão descrevem como os padrões devem ser apresentados. Padrões de projeto apresentam as soluções para as situações comuns em projeto de software. Gamma *et al*. (1995) é um bom exemplo de como a classificação dos padrões não deve ser rígida. Eles dividem os padrões de projeto em padrões de criação, estruturais e de comportamento.

A **Figura 2.1** mostra a estrutura de classes do padrão de projeto Herança de Interface (do inglês *Interface Inheritance*) apresentada por Schmid (1997; 1999). Esse padrão é o mais simples de todos, composto apenas da classe base abstrata e de suas subclasses concretas, que implementam a flexibilidade requerida. Não existe nenhuma outra classe, abstrata ou concreta, nem nenhum outro relacionamento entre as classes. A classe Cliente requisita um serviço oferecido pelo UmMétodo da ClasseBase abstrata. Pode existir uma implementação padrão (do inglês *default*) do método requisitado na ClasseBase mas, normalmente, utiliza-se

a implementação existente na subclasse concreta. Para cada objeto, de subclasses distintas, a resposta à requisição será, em princípio, diferente, proporcionando assim a flexibilidade necessária.

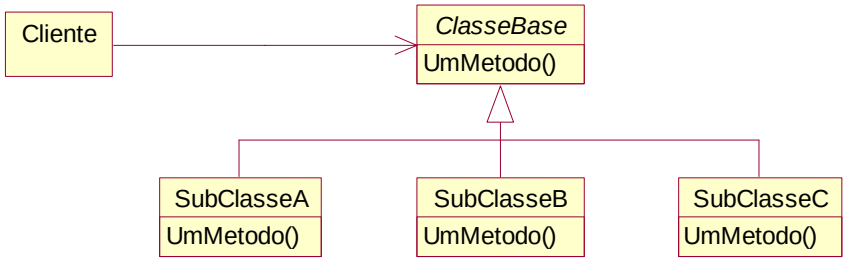


Figura 2.1 – Padrão Herança de Interface (Schmid, 1997; 1999)

A **Figura 2.2** apresenta o padrão Cadeia de Responsabilidade (do inglês *Chain of Responsibility*), utilizando a notação proposta por Gamma *et al*. (1995).

O diagrama de classes do padrão Método de Fabricação (do inglês *Factory Method*) pode ser visto na **Figura 2.3**. O padrão define uma interface para criação de objetos, mas deixa as subclasses decidirem qual classe deve ser instanciada. Ele é útil quando uma classe cliente não pode antecipar a qual classe pertence um objeto que deve ser criado ou quando hierarquias paralelas de classes devem ser conectadas (Gamma *et al*., 1995).

Neste trabalho optou-se por padronizar a apresentação dos diagramas de classe utilizando a notação UML (Booch *et al*., 2000; Eriksson & Penker, 1998; Fowler & Scott, 1998) e, assim, os diagramas originais foram adaptados para essa notação, utilizando-se a ferramenta Rational Rose (Rational, 2000). As representações da notação UML utilizadas no escopo deste trabalho estão descritas em anexo.

Fowler (1997) apresenta um grupo de padrões de análise que são aplicáveis no domínio de modelos de negócios. Ele caracteriza os padrões de análise como um “grupo de conceitos que representam uma construção comum em modelos de negócios”. Coad (1992) propôs sete padrões de análise utilizáveis para o desenvolvimento de sistemas de informação. Posteriormente, ele reuniu em um livro um total de trinta e um padrões (Coad *et al*., 1995; 1997). Em muitos aspectos os padrões de análise de Fowler são uma forma mais pura de padrões de modelos que os padrões de Coad, que lidam explicitamente com a modelagem conceitual de um domínio (Jacobson & Nowack,1999).

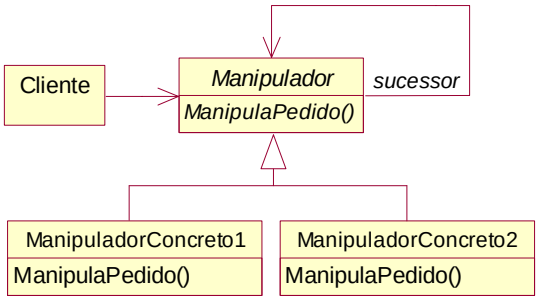
CADEIA DE RESPONSABILIDADE

Intenção:
Acoplar aquele que envia um pedido aos seus receptores, dando a mais de um objeto a chance de manipular o pedido. Encadeia os objetos receptores e passa o pedido por toda a cadeia até que um objeto possa manipulá-lo.

Motivação:
A idéia deste padrão é não relacionar diretamente aquele que envia um pedido àquele que o recebe, dando a oportunidade a diversos objetos de manipular a requisição.
. . .

Aplicabilidade:
Utilize a cadeia de responsabilidade quando:
- mais de um objeto pode manipular o pedido e o manipulador não é conhecido a priori. O manipulador precisa ser averiguado automaticamente;
- quer-se emitir um pedido a um objeto, entre vários, sem especificá-lo explicitamente;
- o grupo de objetos que pode manipular um pedido deve ser especificado dinamicamente.

Estrutura:



Participantes:
Manipulador:
- define uma interface para manipular pedidos;
- implementa (opcional) o acesso ao próximo elemento da cadeia (sucessor).

ManipuladorConcreto:
- manipula pedidos, pelos quais ele é responsável;
- pode acessar o seu sucessor;
- se ele puder manipular o pedido, ele o faz, caso contrário propaga o pedido ao seu sucessor.

Cliente:
- envia o pedido para o primeiro objeto da cadeia.

Colaborações:
Quando um cliente emite um pedido, ele é propagado através da cadeia até que um objeto ManipuladorConcreto assuma a responsabilidade de manipulá-lo.

Conseqüências:
- Reduz o acoplamento entre Cliente e ManipuladorConcreto.
- Adiciona flexibilidade ao atribuir responsabilidades a objetos.
- Não garante a manipulação do pedido.

Implementação:
. . .

Exemplo de Código:
. . .

Usos Conhecidos:
Várias bibliotecas de classes utilizam o padrão Cadeia de Responsabilidade para manipular eventos provenientes do usuário. Cita-se: MacApp, ET++, Symantec's TCL library e NeXT's AppKit, que implementam a mesma funcionalidade, porém com nomes distintos.

Padrões Relacionados:
. . .

Figura 2.2 – Padrão Cadeia de Responsabilidade (Gamma *et al.*, 1995)

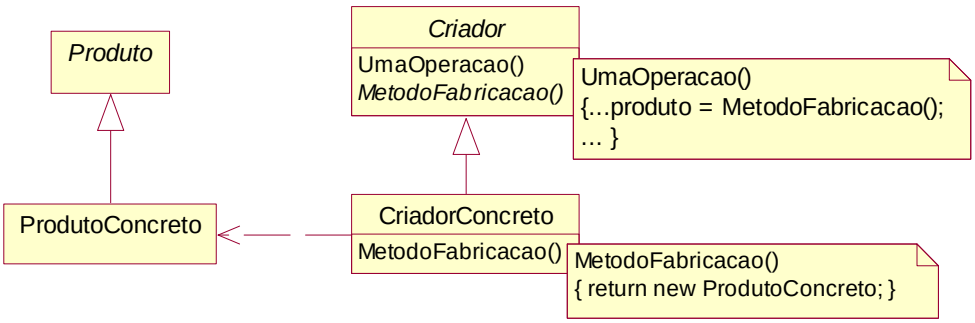


Figura 2.3 – Padrão Método de Fabricação (Gamma *et al.*, 1995)

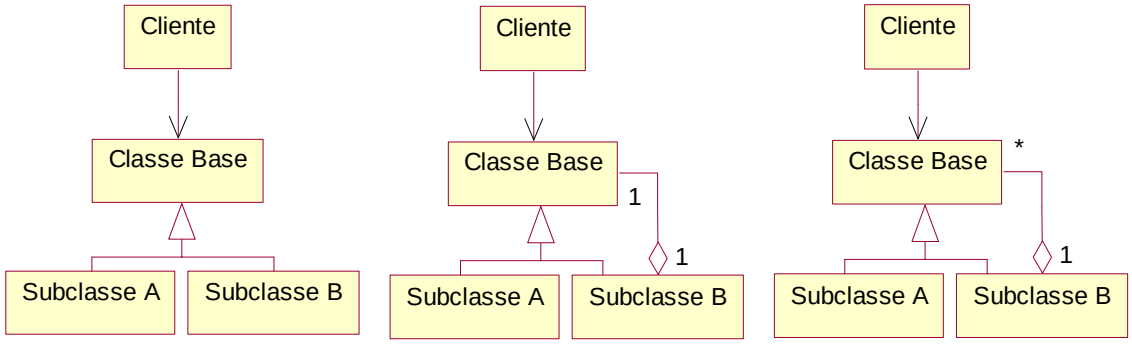
Com o surgimento de conferências internacionais para apresentação e discussão de novos padrões, com os melhores artigos sendo publicados em livros (Coplien & Schmidt, 1995; Martin *et al*., 1998; Vlissides *et al*., 1996), a disponibilidade de padrões dos diversos tipos aumentou consideravelmente. Entretanto, quando se recorre aos padrões de projeto, o livro de Gamma *et al*. (1995) tornou-se clássico pois contém os mais utilizados.

2.3 Subsistema de Ponto Variável

Um subsistema de ponto variável (do inglês *hot-spot subsystem*) é, normalmente, composto por uma classe base (geralmente abstrata), por subclasses concretas e, muitas vezes, por classes e relações adicionais. A classe base define a interface para todas as alternativas do ponto variável. A subclasse concreta representa uma das instâncias dos aspectos variáveis (Schmid, 1999). Um subsistema de ponto variável sem classes e relações adicionais é implementado, normalmente, pelo mecanismo de Herança de Interface, ou por padrões como Método de Fabricação ou Método Genérico (do inglês *Template Method*), entre outros. Quando o subsistema contém classes e relações adicionais, sua estrutura é implementada por padrões mais complexos.

Um subsistema de ponto variável introduz uma variabilidade, seja por herança, seja por composição, sem que um cliente fora do subsistema seja afetado. O cliente requisita um serviço oferecido por um método genérico concreto da classe base abstrata. O método genérico requisita serviços de métodos componentes de sua própria classe, os quais são redefinidos nas subclasses concretas da classe base. A variabilidade é alcançada pela ligação dinâmica do método componente da subclasse, que é realmente executado, com o método componente da classe base devido à referência polimórfica existente entre eles (Pree, 1995). Os termos genérico (*template*) e componente (*hook*) são traduções utilizadas por Quadros (1997) e Quadros e Rubira (1997) dos termos correspondentes na língua inglesa utilizados por Pree (1995). Existem casos em que o método componente da subclasse requisita serviços do próprio subsistema de ponto variável. Neste caso os subsistemas são chamados de recursivos.

Com base nesses aspectos, Schmid (1997; 1999) classifica os subsistemas de ponto variável em três categorias: não recursivos, quando o serviço requisitado pela classe Cliente é executado por uma subclasse; recursivo 1:1 (estrutura em cadeia), quando o serviço pode ser atendido por um dos n objetos das subclasses que estão estruturados em cadeia; e recursivo 1:n (estrutura em árvore), quando o serviço pode ser executado por uma árvore de n objetos das subclasses. Os diagramas de classes das três categorias podem ser vistos na **Figura 2.4**.



a) não recursivo b) recursivo 1:1 c) recursivo 1:n

Figura 2.4 – Subsistema de ponto variável (Schmid, 1997; 1999)

Schmid (1999) também aloca os padrões de projeto propostos por Gamma *et al*. (1995) dentro da sua classificação (**Tabela 2.1**).

Tabela 2.1 – Classificação dos padrões (Schmid, 1999)

Subsistema de ponto variável	Padrão de projeto
não recursivo	herança de interface, fabricação abstrata, construtor, método de fabricação, protótipo, adaptador, ponte, representante, comando, iterador, mediador, observador, estado, estratégia, método genérico, visita
recursivo 1:1	cadeia de responsabilidade, decorador
recursivo 1:n	composição, interpretador

2.4 *Frameworks*

Muitos autores concordam que *framework* é uma arquitetura de software reutilizável compreendendo tanto o projeto quanto o código, mas normalmente não aceitam uma definição de *framework* e de suas partes constituintes (Bosch *et al*., 1999). Muitos autores entendem por reuso não só o aproveitamento do projeto e do código, mas sim de todas as fases do desenvolvimento de software, desde a análise de requisitos até os testes.

A definição mais aceita, provavelmente, é aquela dada por Johnson e Foote (1988): “Um *framework* é um conjunto de classes que engloba um projeto abstrato para uma família de problemas relacionados”.

Pree (1995) afirma que um *framework* é uma coleção de classes abstratas e concretas e a interface entre elas e, também, é o projeto de um subsistema. As classes abstratas formam a base de um *framework*. Outras classes concretas ou abstratas podem ser implementadas com base nas interfaces oferecidas pelas classes abstratas.

2.4.1 Histórico

Os primeiros exemplos do conceito de *framework* encontrados na literatura estão relacionados com o ambiente Smalltalk (Goldberg & Robson[2] *apud* Bosch *et al*., 1999) e com a Apple Incorporation (Schmucker[3] *apud* Bosch *et al*., 1999). O *framework* de interface com o usuário Model-View-Controler (MVC) desenvolvido na linguagem Smalltalk-80, foi o primeiro *framework* usado em grande escala. A Apple Incorporation desenvolveu o *framework* de interface com o usuário MacApp para dar suporte à implementação de aplicações para o Macintosh. Eles atraíram maior atenção quando os *frameworks* de interface com o usuário InterViews (Linton *et al*.[4] *apud* Bosch *et al*., 1999) e ET++ (Weinand *et al*.[5] *apud* Bosch *et al*., 1999) foram desenvolvidos e disponibilizados. Com a formação da empresa Taligent em 1992, os *frameworks* passaram a atrair o interesse de uma grande comunidade, pois a companhia desenvolveu um sistema operacional orientado a objetos baseado no conceito de *frameworks*. Ela também disponibilizou um grupo de ferramentas para o desenvolvimento rápido de aplicações, denominado CommonPoint, que consiste de mais de cem *frameworks* orientados a objetos (Taligent, 1994; 1995). A abordagem da Taligent causou uma mudança no enfoque de desenvolvimento de um grande *framework* para o desenvolvimento de vários *frameworks* menores integrados entre si.

Os padrões, principalmente de projeto, (Gamma *et al*., 1995) têm sido utilizados, preferencialmente, para documentar, desenvolver e descrever as decisões de projeto tomadas no desenvolvimento de *frameworks*, mas também as linguagens de padrões (Brugali & Menga, 1999; Roberts & Johnson, 1998) e os subsistemas de pontos variáveis (Schmid, 1997; 1999) são utilizados em menor escala. Como os *frameworks* passam por um processo de iterações antes que a primeira versão seja completada, tanto o projeto como o código do *framework* muda freqüentemente. Opdyke[6] *apud* Bosch *et al*. (1999) define uma série de transformações, denominadas refatoração, que auxiliam a remover cópias múltiplas de códigos similares sem alterar o comportamento do *framework*.

---

[2] GOLDBERG, A.; ROBSON, D. (1989). *Smalltalk-80:* The Language. Reading-MA: Addison-Wesley
[3] SCHMUCKER, K.J. (1986). *Object-Oriented Programming for the Macintosh*: Hayden Book Company.
[4] LINTON, M.A.; VLISSIDES, J.M.; CALDER, P.R. (1989). Composing User Interfaces with InterViews. *IEEE Computer*, v. 2, n. 22. February.
[5] WEINAND, A.; GAMMA, E.; MARTY, R. (1989). Design and Implementation of ET++, a Seamless Object-Oriented Application Framework. *Structured Programming*. v. 2, n. 10., July.
[6] Opdyke, W.F. (1992). *Refactoring Object-Oriented Frameworks*. Urbana-Champaign. Thesis (Doutorado) – University of Illinois,

Os *frameworks* não são limitados só aos *frameworks* de interface com o usuário, mas sua aplicação estendeu-se também para muitos outros domínios. Recentemente, Fayad e Johnson (2000) publicaram um livro com coletâneas de *frameworks* para os domínios de manufatura, sistemas distribuídos, telecomunicações e redes, além de uma seção genérica com *frameworks* de outros domínios. No escopo deste trabalho há interesse nos *frameworks* desenvolvidos para o domínio da manufatura, que são descritos em maior grau de detalhe na seção 3.6.

2.4.2 Classificação

Os aspectos variáveis de um domínio de aplicação são chamados de pontos variáveis (do inglês *hot spot*) (Buschman *et al*., 1996). As classes que compõem o ponto variável são organizadas em subsistemas de ponto variável (Schmid, 1996a). O modo como são usados um ou mais subsistemas de pontos variáveis diferencia as aplicações dentro do mesmo domínio. Os subsistemas de pontos variáveis são projetados para serem genéricos e adaptáveis às necessidades da aplicação. Normalmente, as classes, as subclasses e as relações que compõem o subsistema de ponto variável são organizadas de acordo com padrões de software (Gamma *et al*., 1995). Além dos aspectos variáveis, o domínio de aplicação contém também os aspectos que não variam. Um ponto fixo (também conhecido como *frozen spot*) define as partes básicas e invariáveis de um determinado domínio de aplicação. Em todas as aplicações derivadas de um mesmo *framework*, os pontos fixos são usados sem nenhuma alteração.

Os *frameworks* são classificados ou de acordo com o seu escopo ou de acordo com a forma de reuso para gerar novas aplicações (Fayad & Schmidt, 1997).

2.4.2.1 Classificação de Acordo com o Escopo

Considerando-se o escopo, *frameworks* de aplicação são classificados como *frameworks* de infra-estrutura do sistema, de integração *middleware* e de aplicação empresarial (Fayad & Schmidt, 1997; Fayad *et al*., 1999; Yassin & Fayad, 2000).

Os *frameworks* de infra-estrutura do sistema são uma coleção de sistemas portáteis e eficientes, que dão suporte às áreas de infra-estrutura dos sistemas como, por exemplo, os sistemas operacionais, interfaces com o usuário, sistemas de comunicação e ferramentas de processamento de linguagens. Geralmente são desenvolvidos e utilizados internamente em uma empresa ou como aplicações genéricas utilizadas por outras aplicações. Interface gráfica com o usuário, *Microsoft Foundation Classes* (MFC) e MacApp são exemplos de *frameworks*

genéricos utilizados para o desenvolvimento de outras aplicações. É difícil quantificar o número de aplicações derivadas desses *frameworks*.

Os *frameworks* de integração *middleware* são projetados para aumentar a modularidade e o reuso e para estender a infra-estrutura dos softwares desenvolvidos para ambientes distribuídos. OmniBuilder (Omni, 2000) e os *object request brokers* (ORBs), tais como VisiBroker, Orbit e DataBroker (Yassin & Fayad, 2000), são exemplos desse tipo de *framework*.

Os *frameworks* de aplicação empresarial são projetados para vários domínios de aplicação, de interesses dos desenvolvedores de softwares comerciais como, por exemplo, sistemas bancários, sistema de telecomunicações, medicina, aviação e manufatura (Yassin & Fayad, 2000). IBM *SanFrancisco Project* (Monday *et al*., 2000) e *Enterprise Resource Planning* (ERP) são exemplos de *frameworks* desta classe. Eles são mais caros para desenvolver ou comprar que os *frameworks* das outras classes, mas podem garantir um retorno significativo do investimento, pois são utilizados para o desenvolvimento de aplicações e produtos mais facilmente comercializáveis.

Os *frameworks* de aplicação empresarial englobam os *frameworks* de integração *middleware*, que por sua vez incluem os *frameworks* de infra-estrutura do sistema (Yassin & Fayad, 2000).

2.4.2.2 Classificação de Acordo com a Forma de Reuso

Considerando-se a forma de reuso, existem três tipos de *framework*: caixa branca, caixa cinza e caixa preta (Yassin & Fayad, 2000). Todos os autores aceitam essa classificação e as definições correspondentes. A adição de métodos às subclasses das classes base de um ou mais pontos variáveis define o comportamento específico do *framework*. No ponto variável caixa branca, o reuso é realizado por herança, enquanto que no ponto variável caixa preta é por composição. No primeiro caso o usuário cria e implementa subclasses da classe base, e para isso precisa conhecer detalhes de implementação do ponto variável para poder reutilizá-lo. No segundo, escolhe-se uma subclasse concreta, dentre algumas existentes, para definir o comportamento do ponto variável e para tal só é necessário o conhecimento das interfaces do ponto variável para usá-lo.

A **Figura 2.5 a**, extraída de Schmid (1997; 1999), mostra um ponto variável caixa branca, denotado por R. Para utilizar o ponto variável é necessário fornecer a implementação

referente a ele, ou seja, através da implementação de R_j. A **Figura 2.5 b**, também extraída de Schmid (1997; 1999), mostra um ponto variável caixa preta, denotado por R, para o qual existem três alternativas de implementação. O usuário deve escolher dentre elas, aquela que apresenta a implementação necessária à sua aplicação. Deve-se observar que as alternativas fazem parte do ponto variável, ou seja, já estão implementadas e portanto não precisam ser desenvolvidas pelo usuário, mas por outro lado são as únicas alternativas disponíveis para o ponto variável. No caso da Figura 2.5 a, como a implementação a ser utilizada para o ponto variável não faz parte dele, é possível efetuar-se uma implementação sob medida.

A **Figura 2.5 c** apresenta um ponto variável caixa cinza, também denotado por R. Nesse ponto variável existem disponíveis três alternativas de implementações, como no ponto variável caixa preta, mas fica aberta, para o usuário, a possibilidade de fornecer uma implementação específica sob medida. Ao observar-se as três classificações nota-se que o ponto variável caixa cinza é uma forma intermediária de reuso entre o ponto variável caixa branca e caixa preta.

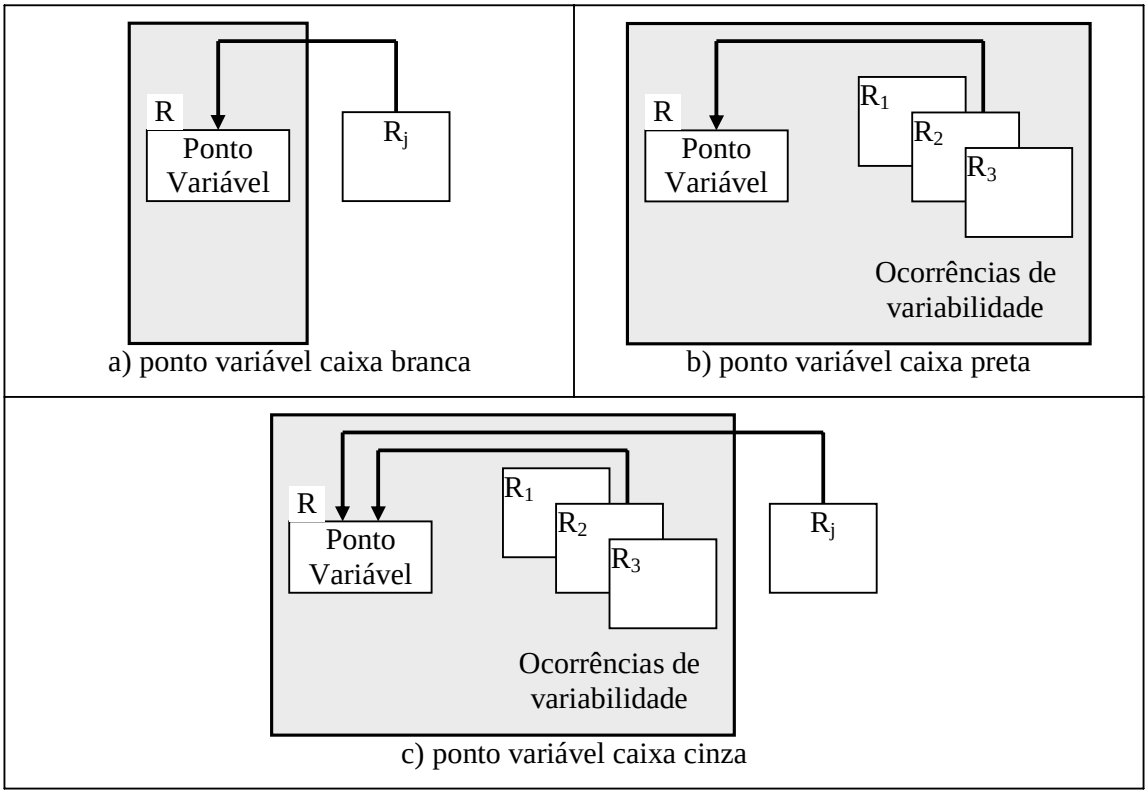


Figura 2.5 – Classificação de pontos variáveis de acordo com a forma de reuso

Tendo-se em mente os conceitos de pontos variáveis caixa branca, caixa cinza e caixa preta, torna-se mais simples a definição dos conceitos de *framework* caixa branca, caixa cinza

e caixa preta. Diz-se que um *framework* é caixa branca se todos os seus pontos variáveis são também caixa branca. Da mesma forma diz-se que um *framework* é caixa preta se todos os seus pontos variáveis são também caixa preta. A grande vantagem da forma aqui apresentada é na definição do *framework* caixa cinza. Diz-se que um *framework* é caixa cinza se todos os seus pontos variáveis forem também caixa cinza, ou se forem de dois ou dos três dos tipos de pontos variáveis definidos. Na literatura não se encontra nenhuma denominação específica para as duas possibilidades de *frameworks* caixa cinza, apesar desses *frameworks* ocorrerem com maior freqüência.

Em suma, um *framework* caixa branca é mais fácil de se projetar e desenvolver, pois não há necessidade de implementar os pontos variáveis caixa branca, o que não ocorre com os *frameworks* caixa preta, pois pelo menos uma alternativa para cada ponto variável caixa preta precisa ser implementada. Entretanto, o *framework* caixa preta é muito mais fácil de ser utilizado, pois basta ao usuário escolher uma das alternativas para cada um dos pontos variáveis caixa preta, ao invés de implementá-las. No intuito de tirar as vantagens de cada um deles, os *frameworks* caixa cinza fornecem uma grande flexibilidade, pois sempre existe a possibilidade de uma implementação específica de um ponto variável e a possibilidade de utilização de pontos variáveis previamente fornecidos. Além da flexibilidade os *frameworks* caixa cinza não precisam expor totalmente o seu código fonte.

A tendência atual é de desenvolver *frameworks* caixa cinza (Yassin & Fayad, 2000) com pontos variáveis também caixa cinza, para que o usuário sempre tenha a possibilidade tanto de utilizar pontos variáveis já implementados como também de implementá-los de forma específica. A evolução natural de um *framework* é sempre no sentido de tornar-se caixa preta, pois, a cada utilização que exija uma nova implementação para um ponto variável, mais uma possibilidade de reutilização está disponível para o próximo desenvolvimento a partir do *framework* (Roberts & Johnson, 1998).

2.4.3 Desenvolvimento de Aplicações Baseada em *Frameworks*

O desenvolvimento de uma aplicação a partir de um *framework* é diferente do mesmo desenvolvimento de forma tradicional. O desenvolvimento do *framework* propriamente dito é também diferente, pois ele deve cobrir todos os aspectos importantes do domínio da aplicação, enquanto que o desenvolvimento de uma aplicação deve ater-se aos requisitos específicos. O desenvolvimento de aplicações a partir de *frameworks* pode ser dividido em

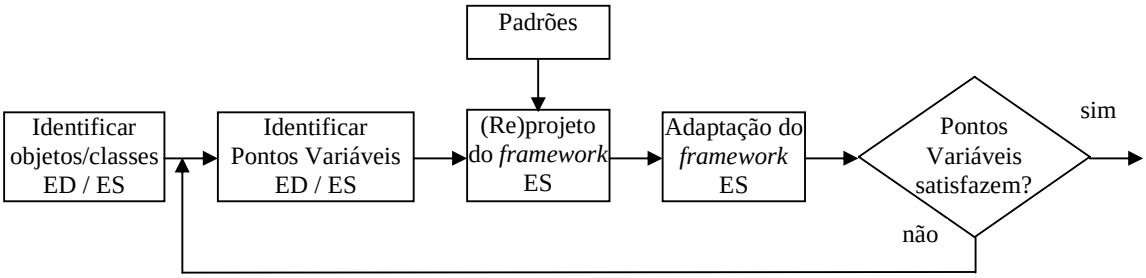
três fases (Bosch *et al.*, 1999): a fase de desenvolvimento, a fase de uso e a fase de evolução e manutenção do *framework*.

O desenvolvimento de *framework* tem sido realizado nos últimos anos, mas não existe um método ótimo para essa atividade. Alguns autores têm proposto formas para esse desenvolvimento, que em alguns aspectos diferem uma da outra. A seguir apresentam-se três propostas de desenvolvimento de *frameworks* e posteriormente descrevem-se as outras fases de desenvolvimento de aplicações baseadas em *frameworks* proposta por Bosch *et al*. (1999).

2.4.3.1 Desenvolvimento Baseado em Pontos Variáveis

Pree (1999) afirma que o desenvolvimento com sucesso de um *framework* requer a identificação dos pontos variáveis específicos do domínio. Os vários aspectos de um *framework* que não podem ser predefinidos para todas as adaptações devem ser implementados de um modo genérico. Conseqüentemente, deve-se perguntar aos especialistas do domínio, qual aspecto difere de uma aplicação para outra, qual grau de flexibilidade é desejado e se o comportamento flexível precisa ser alterado durante o funcionamento da aplicação.

A abordagem baseada em pontos variáveis pode ser resumida na **Figura 2.6**. Uma vez que os pontos variáveis desejados foram identificados, as características dos padrões auxiliam a alcançar o nível apropriado de flexibilidade. Padrões captam e classificam o projeto de *frameworks*, dando apoio à adaptação destes e ao desenvolvimento de novos *frameworks*.



ED: Especialista do Domínio
ES: Engenheiro de Software

Figura 2.6 – Desenvolvimento baseado em pontos variáveis (Pree, 1995)

2.4.3.2 Desenvolvimento Baseado em Generalização Sistemática

O desenvolvimento de *frameworks* proposto por Schmid (1997; 1999) parte de um modelo simples de aplicação no domínio específico. A análise e especificação dos pontos

variáveis são subdivididas na análise de alto nível dos pontos variáveis e na análise e especificação detalhada dos pontos variáveis. Na análise de alto nível dos pontos variáveis todos os pontos variáveis são coletados e descritos superficialmente. A atividade de análise e especificação detalhada dos pontos variáveis é realizada para cada ponto variável e nela analisam-se os requisitos de variabilidade e flexibilidade e descrevem-se as características do ponto variável. As classes e a estrutura de um subsistema de ponto variável são derivadas das características do ponto variável durante a atividade de projeto de alto nível do subsistema de ponto variável. A atividade de transformação por generalização produz um modelo de classes genérico substituindo um aspecto fixo pelo subsistema de ponto variável que resultou da atividade de projeto de alto nível. A **Figura 2.7** apresenta essas atividades em forma gráfica. Elas devem ser realizadas sempre no sentido descendente, ou seja, uma atividade só pode ser realizada, quando a atividade correspondente acima já foi completada.

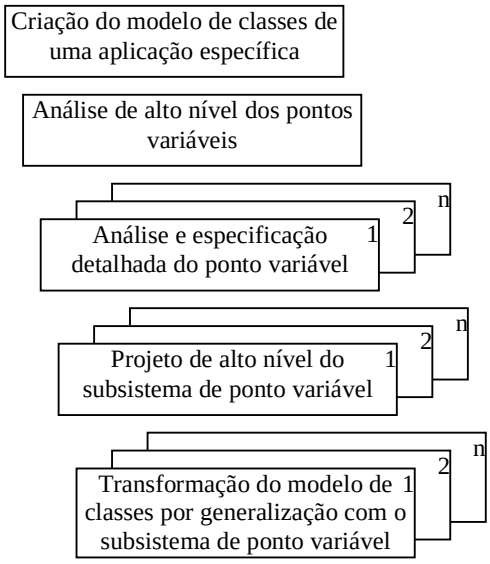


Figura 2.7 – Desenvolvimento baseado em generalização sistemática (Schmid, 1999)

2.4.3.3 Desenvolvimento Proposto por Bosch

Bosch *et al*. (1999) propõem que o desenvolvimento de *frameworks* seja realizado em seis atividades:

Na Análise do Domínio realiza-se a descrição do domínio a ser abrangido pelo *framework*. Extraem-se os requisitos e identificam-se os conceitos do domínio e a relação existente entre esses conceitos.

No Projeto Arquitetural o projetista decide o estilo arquitetural sob o qual desenvolver-se-á o *framework*. Como resultado obtém-se o projeto de alto nível do *framework* a partir da análise do domínio.

No Projeto do *Framework* refina-se o projeto de alto nível, resultando no escopo de funcionalidade, nas interfaces, nas regras de projeto baseadas nas decisões arquiteturais e no histórico do projeto do *framework*, descrevendo os problemas encontrados e as soluções adotadas.

Na Implementação do *Framework* as classes abstratas e concretas são codificadas.

Durante o Teste do *Framework* é feita uma avaliação da funcionalidade e da usabilidade do *framework*. Entretanto, não é trivial decidir se o *framework* é utilizável ou não. A única maneira é utilizá-lo para desenvolver aplicações. Dependendo do tipo de aplicação desenvolvida, podem-se testar diferentes aspectos do *framework*. Com base no teste desses aspectos pode-se decidir se o *framework* deve ser reprojetado ou se está maduro o suficiente para ser liberado.

A fase de Documentação é a atividade mais importante do desenvolvimento de *frameworks*, apesar de normalmente não ser assim reconhecida. Sem uma documentação clara, completa e correta que descreva como utilizá-lo, um manual do usuário e a documentação do projeto, que descreva como ele funciona, o *framework* será impossível de ser utilizado por desenvolvedores de aplicação que não participaram do projeto do *framework*.

2.4.4 Uso e Aprendizagem de *Frameworks*

Um *framework* é utilizado para desenvolver aplicações. Esta fase recebe, também, o nome de instanciação do *framework*. Para tal, podem-se criar subclasses das classes base dos pontos variáveis, novas classes e relações entre elas, com o intuito de desenvolver uma aplicação específica.

É fácil de aprender a utilizar um *framework* se ele tiver uma boa documentação ou se for disponível um bom treinamento. Uma das maneiras de aprender a utilizar um *framework* é por meio de exemplos. A maioria dos *frameworks* disponibiliza uma série de exemplos que podem ser estudados. São exemplos concretos, mais fáceis de serem compreendidos do que todo o *framework*. Eles mostram o desenvolvimento de uma aplicação particular e possibilitam ao usuário a entender o fluxo de controle do *framework* (Fayad *et al.*, 1999).

2.4.5 Evolução e Manutenção de *Frameworks*

O desenvolvimento de um *framework* deve ser visto como um investimento de longo prazo. Logo, ele precisa ser tratado como um produto que necessita de manutenção. Durante o desenvolvimento de um *framework* são realizadas várias iterações até atingir-se a reusabilidade requerida. O único meio de atingi-la é usar o *framework* para produzir aplicações, pois nesse processo são identificadas deficiências que precisam ser reparadas. A correção de um erro no *framework* não é tão simples como possa parecer. Para concluir o desenvolvimento da aplicação, durante a qual detectou-se o erro, a correção faz-se necessária. Entretanto, o tratamento das aplicações anteriormente desenvolvidas é problemático. Ou essas aplicações precisam também ser corrigidas, ou deixam de ter compatibilidade com a versão mais atual do *framework* (Bosch *et al*., 1999). A correção das aplicações desenvolvidas nem sempre é viável, devido ao custo envolvido e à imprecisão sobre seu conhecimento. Eventuais manutenções ou extensões dessas aplicações serão também problemáticas, pois novos erros podem ser introduzidos com o uso da nova versão do *framework*. Uma solução possível é a divisão do *framework* em dois, o que pode acarretar futuros problemas de manutenção.

Frameworks tentam captar os vários aspectos de um domínio, mas em muitos casos é difícil estabelecer as fronteiras desse domínio. Mudanças de escopo dos usuários do *framework* podem levar a uma extensão das fronteiras do domínio e, conseqüentemente, a evolução do *framework* é inevitável para a sua sobrevivência.

2.5 Comparação entre Formas de Reuso

Os padrões são utilizados para descrever a flexibilidade de um ponto variável de um *framework*. Diferentes padrões podem ser aplicados para introduzir a mesma funcionalidade em um *framework*, mas podem resultar em diferentes estratégias de implementação. A característica mais importante dos padrões é o relacionamento que ele faz entre um problema e uma solução, no contexto em que ele deve ser aplicado.

Os subsistemas de ponto variável são utilizados após a análise da flexibilidade e, principalmente, da multiplicidade requerida nessa flexibilidade. Como conseqüência tem-se a especificação do subsistema de ponto variável, não recursivo, recursivo 1:1 ou recursivo 1:n, a ser utilizado. Schmid (1999) classificou os padrões de projeto de acordo com a abordagem de subsistemas de ponto variável. Logo, ao ser determinada a multiplicidade requerida para atender a flexibilidade necessária de um subsistema de ponto variável, está-se especificando

também quais são os padrões candidatos a implementá-la. Um resumo do procedimento pode ser visto na seção 2.4.3.2 e a sua descrição completa é apresentada por Schmid (1999).

Os subsistemas de ponto variável são mais genéricos e mais flexíveis que os padrões. Em contrapartida eles não fazem referência à funcionalidade específica que um ponto variável deve possuir, como é o caso dos padrões de projeto. O subsistema de ponto variável a ser empregado é resultado de uma análise e não da experiência do desenvolvedor do *framework*. Conseqüentemente, eles serão determinados mais facilmente, mas a implementação ocorrerá por meio de padrões escolhidos pelo desenvolvedor.

Tanto os padrões como os subsistemas de ponto variável são de níveis semelhantes quando comparados com *frameworks*. Na comparação a seguir será utilizado apenas o termo padrão, como representativo de ambas as formas de reuso comparadas anteriormente.

Frameworks podem ser aplicados, desenvolvidos e documentados por meio de padrões. Como regra um *framework* possui diversos pontos variáveis implementados por meio de padrões. Logo, um *framework* contém vários padrões. Ele pode, até mesmo, ter mais de um ponto variável implementado pelo mesmo padrão. Conseqüentemente, *frameworks* são maiores que padrões. Normalmente, os padrões são compostos por poucas classes e as relações entre elas, enquanto os *frameworks* possuem um número bem maior de classes, englobando diversos padrões.

Padrões são mais abstratos que os *frameworks*, pois padrões representam uma funcionalidade que ocorre com freqüência em diversos domínios, enquanto que *frameworks* são mais concretos, incluindo até o código executável (Johnson, 1997). Padrões são mais genéricos do que os *frameworks*, pois podem ser usados em diversos tipos de aplicações, enquanto que os *frameworks* são mais especializados e, normalmente, são desenvolvidos para um domínio de aplicação específico.

2.6 Considerações Finais

Neste capítulo apresentou-se uma revisão bibliográfica para documentar o estado da arte sobre os padrões, os subsistemas de pontos variáveis e os *frameworks*. Efetuou-se a comparação entre as formas de reuso para evidenciar suas semelhanças e suas diferenças. A análise da literatura mostra que o desenvolvimento de aplicações baseadas em *frameworks* é uma tendência atual, pois eles são uma forma promissora de reuso não só de projeto e de código, mas também de todas as fases do desenvolvimento de software, desde a análise até os testes.

3 O Domínio da Manufatura

3.1 Considerações Iniciais

A manufatura caracteriza um domínio complexo, o que é evidenciado pela falta de uma padronização da classificação dos seus diversos subdomínios pelos autores da área. Dos subdomínios encontrados na literatura, são de interesse no escopo deste trabalho o planejamento do processo e a produção. Os outros subdomínios serão considerados aqui apenas como áreas afins (**Figura 3.1**). Não há também termos equivalentes para denominar o domínio e suas subdivisões. Os autores da linha americana denominam o domínio de "*manufacture*" e o subdomínio mostrado à direita na figura de "*production*". Este trabalho segue essa nomenclatura e, portanto, utiliza respectivamente os termos manufatura e produção. A linha alemã tem preferência pelos termos "*Produktion*" e "*Fertigung*", que são traduzidos por produção e fabricação respectivamente. As duas linhas de trabalho utilizam termos equivalentes para o subdomínio planejamento do processo.

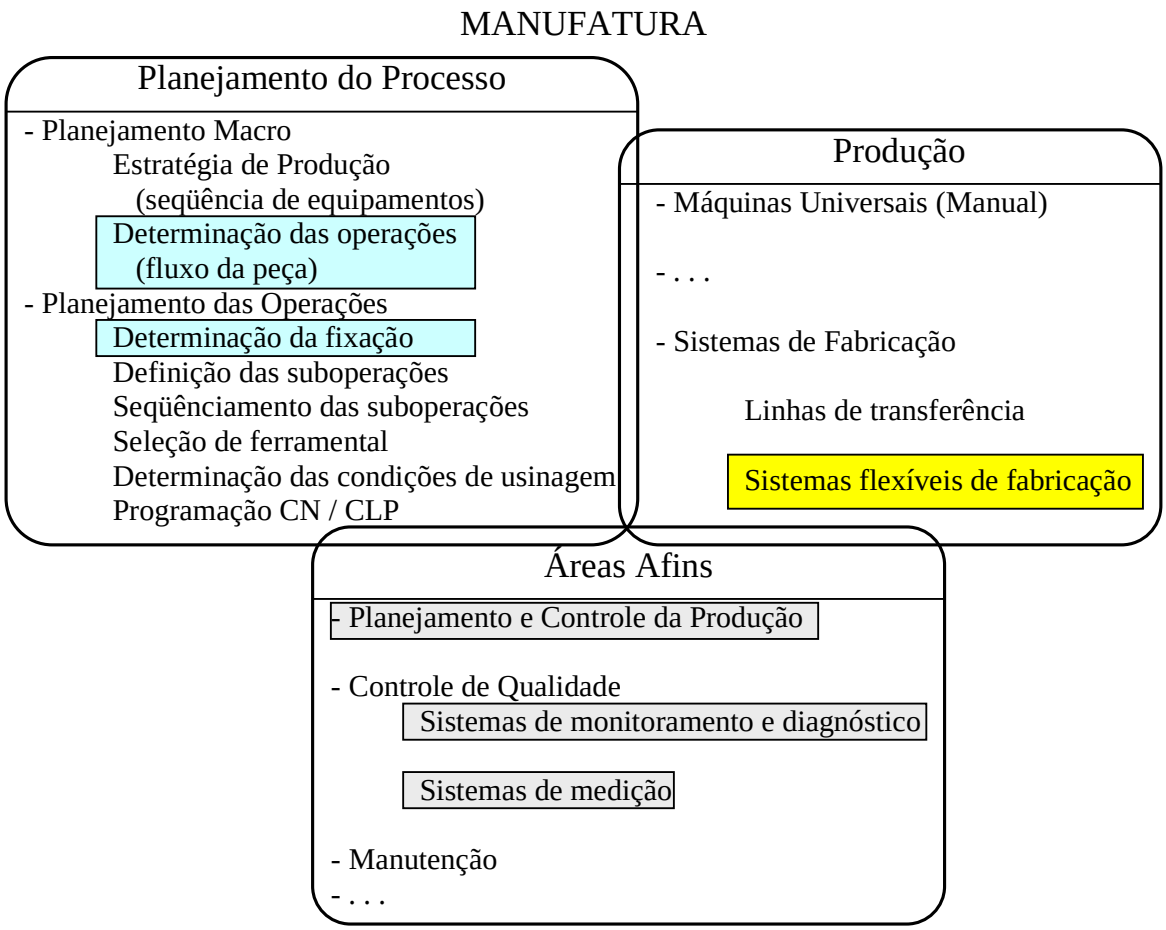


Figura 3.1 – Domínio da manufatura

A Figura 3.1 é o resultado da análise conjunta de algumas publicações (Halevi & Weill, 1995; Rozenfeld, 1992; Tonshoff *et al*., 1992; Weck, 1991). O subdomínio planejamento do processo pode ser subdividido em planejamento macro e planejamento das operações (Halevi & Weill, 1995). O planejamento macro consiste da definição dos equipamentos que serão utilizados durante a produção da peça. A seqüência em que são utilizados os equipamentos recebe o nome de estratégia. Uma operação é caracterizada pelo mesmo posicionamento geométrico relativo entre o equipamento e a peça. No caso de máquinas-ferramenta, equipamentos que realizam a usinagem da peça, o posicionamento relativo é garantido pelo sistema de fixação da máquina. Nesse caso, uma operação é caracterizada pelo uso da mesma máquina e da mesma fixação. A seqüência de operações caracteriza o fluxo da peça pelo sistema de fabricação. As operações são novamente divididas em suboperações, as quais são definidas pelo emprego da mesma ferramenta de usinagem.

O subdomínio produção é subdivido de acordo com os tipos de máquinas-ferramenta e a forma como elas estão agrupadas. Essa divisão pode variar desde uma máquina universal comandada manualmente por um operário até os sistemas de fabricação, que diferem dos demais conjuntos de equipamentos de produção por incorporarem o fluxo automático de peças e, em muitos casos, de ferramentas e dispositivos, por todos os seus equipamentos. Se o sistema é rígido, projetado para a produção de grandes lotes de uma única ou, eventualmente, de poucas peças, é denominado Linha de Transferência (do inglês *Transfer Line*). Se o sistema é flexível, desenvolvido para fabricar pequenos lotes de um espectro grande de peças, recebe o nome de Sistema Flexível de Fabricação (SFF – do inglês *Flexible Manufacturing System*) (Weck, 1991).

Na Figura 3.1 destacam-se alguns pontos do domínio. A parte do planejamento macro e a parte do planejamento das operações demarcada em azul representa a abrangência do desenvolvimento realizado no escopo deste trabalho. Os capítulos 4, 5 e 6 mostram em maior detalhe o trabalho realizado. Os SFF caracterizam a associação de equipamentos, em que os resultados obtidos pelo sistema desenvolvido neste trabalho são mais bem utilizados. Eles estão destacados em amarelo na figura. Na literatura encontram-se diversas publicações sobre *frameworks* para controle dos SFF. Em uma publicação recente são apresentados, além dos *frameworks* para controle de SFF, alguns outros que são do domínio da manufatura (Fayad & Johnson; 2000). O subdomínio em que esses *frameworks* foram desenvolvidos não são de interesse no escopo deste trabalho mas, mesmo assim, estão demarcados em cinza na figura. Um resumo dos *frameworks* encontrados na literatura é apresentado no fim deste capítulo.

Para a produção de peças em SFF são utilizadas uma ou mais máquinas-ferramenta. Durante a produção, as peças passam por uma ou, sucessivamente, por várias operações em diferentes máquinas. Uma folha de processo, o resultado do planejamento do processo, descreve, para cada peça, a seqüência de operações e os recursos necessários em cada uma delas. Uma ordem de fabricação é uma ordem para que uma dada máquina execute uma dada operação para um dado número de peças. A movimentação das peças, desde o depósito de peças em bruto até o depósito de peças prontas, passando pelas máquinas necessárias para executar a respectiva seqüência de operações, é realizada por sistemas de transporte e de manipulação. Para cada máquina existe uma programação de alocação da máquina, que especifica a seqüência e o horário em que as ordens de fabricação devem ser executadas. Além disso, a programação de alocação da máquina faz também referência à folha de processo da peça a ser processada.

Como um dos objetivos do sistema desenvolvido no escopo deste trabalho não é controlar um SFF, mas sim simular o fluxo de uma peça através dele, para encontrar a melhor alternativa para produção da peça, a folha de processo, a ordem de fabricação e a programação de alocação da máquina não são conhecidas. A informação disponível é a estratégia de produção, ou seja, os processos de fabricação necessários para produzir-se a peça e que tipo de máquina é capaz de executar o respectivo processo.

Para a produção de peças, ou ela é proveniente de uma peça em bruto disponível em um depósito apropriado, ou ela é proveniente de um material em bruto, com comprimento além do necessário. Conseqüentemente, necessita-se de uma operação de serramento para obter a peça em bruto. A peça passa por vários processos de fabricação com o objetivo de produzir mudanças em sua geometria. Existe a possibilidade de realizar um processo final de acabamento com o objetivo de melhoria da qualidade dimensional, da forma e do acabamento superficial, sem mudanças significativas da geometria.

Na seção 3.2 apresenta-se um breve histórico dos SFF, dando-se ênfase à importância do fluxo de material e aos software para sua simulação. Na seção 3.3 definem-se os conceitos básicos sobre fluxo de material e suas subdivisões. Na seção 3.4 mostram-se os equipamentos e os dispositivos que realizam as subfunções da manipulação de interesse no escopo deste trabalho. Na seção 3.5 apresentam-se vários sistemas computacionais que simulam o emprego dos equipamentos e dos dispositivos utilizados na manipulação. Na seção 3.6 apresenta-se e discute-se dois *frameworks* para controle dos sistemas flexíveis de fabricação freqüentemente presentes na literatura. Na seção 3.7 são apresentadas as considerações finais deste capítulo.

3.2 Histórico

As exigências do mercado consumidor da indústria metal-mecânica nos anos 80, caracterizadas principalmente pelo aumento da complexidade dos produtos, aumento da diversificação de seus tipos, bem como uma diminuição de sua vida útil, associada à necessidade da diminuição do tempo entre o pedido e a entrega, levaram à instalação de sistemas de fabricação capazes de produzir lotes cada vez menores, muitas vezes unitários, com alta produtividade, qualidade e, principalmente, flexibilidade. A fim de se poder produzir um maior espectro de peças, sem a necessidade de reconfiguração do sistema de fabricação, as indústrias viram-se obrigadas a uma automação flexível de sua produção. Desde então, os chamados Sistemas Flexíveis de Fabricação têm aumentado em importância e número de instalações (Weck *et al*., 1987).

Nos anos seguintes, o desenvolvimento dos SFF foi caracterizado pelo acréscimo de unidades produtivas a eles agregados. As possibilidades de comunicação entre computadores e os controladores das máquinas (comando numérico – CN ou controlador lógico programável - CLP) permitem que as informações sejam disponibilizadas em seu destino em poucos ou até mesmo frações de segundos. As interfaces utilizadas são descritas em uma série de normas. Portanto, a realização do fluxo de informação em um SFF depende apenas de sua estruturação. Entretanto, a realização do fluxo de material pelos SFF permaneceu desde o início como sendo um problema mecânico, que abrange principalmente a configuração das interfaces mecânicas. O encadeamento de diferentes unidades produtivas, com o auxílio de equipamentos para manipulação, transporte, teste e/ou armazenamento gerou requisitos, até mesmo normas, para a configuração das interfaces mecânicas, tanto do lado dos equipamentos quanto do lado dos dispositivos (dispositivos de fixação e sistemas de apoio de peças, entre outros). Apesar das deficiências de padronização, a interligação de cada um dos equipamentos dos SFF tornou-se possível graças a trabalhos abrangentes de adaptação dessas interfaces (Weck *et al*., 1990).

Ao contrário dos equipamentos dos SFF e dos dispositivos por eles utilizados, as peças a serem produzidas não podem ser padronizadas. Além disso, deve-se considerar a variação de sua forma durante o processo produtivo. Como conseqüência, a verificação da possibilidade de realização do fluxo da peça continua sendo um grande problema (Gertosio *et al*., 2000; Kogan *et al*., 1997). Devido a essa dificuldade, as interfaces mecânicas entre os equipamentos dos SFF ou entre seus dispositivos e a peça, normalmente, só podem ser

verificadas durante intervalos de funcionamento dos SFF. Muitas vezes a paralisação se faz necessária somente por causa dessa verificação.

Os anos 90 caracterizaram-se pela globalização do comércio e os mercados, que antes eram locais ou regionais, tornaram-se mundiais. Com isso aumentou a quantidade de clientes e também a quantidade de fornecedores. A concorrência ficou ainda mais acirrada, exigindo que as reações da indústria metal-mecânica às mudanças do mercado fossem cada vez mais rápidas.

Os softwares comerciais hoje existentes simulam cada vez com melhor qualidade o funcionamento de diversos equipamentos dos SFF, gerando até mesmo os comandos necessários para controle desses equipamentos (parte não marcada no planejamento das operações na Figura 3.1). Os softwares utilizados são subdivididos de acordo com o tipo de equipamento simulado, apesar de, em muitos casos, considerarem de forma simplificada os demais equipamentos (por exemplo, a necessidade de espaço). Assim, pode-se evitar a colisão entre dois equipamentos do SFF e/ou seus dispositivos, atuando diretamente sobre a peça, no instante em que ela é transferida entre eles.

3.3 Conceitos Básicos

3.3.1 Fluxo de Material

Apesar da existência de inúmeras normas do Instituto Alemão de Normalização (DIN – do alemão *Deutsches Institut für Normung e.V.*) e diretrizes da Associação dos Engenheiros Alemães (VDI – do alemão *Verein Deutscher Ingenieure*) sobre fluxo de material, esse conceito não é unicamente definido. Enquanto, por um lado, é definido como sendo "a interligação de todos os processos no ganho, na fabricação, no trabalho e na divisão de bens materiais dentro de uma região fixa"[7] (DIN30781, 1989; VDI2411, 1970; VDI3300, 1973), por outro lado é simplesmente definido como "movimentação de peças" (VDI3961, 1989).

As diversas normas e diretrizes também subdividem o conceito de fluxo de material de forma distinta. A diretriz VDI3300 (1973) o subdivide em: fabricação, manipulação, transporte, teste, permanência e estocagem[8], enquanto que a VDI2411 (1970) utiliza o termo prover (do alemão *Fördern*) no lugar de transportar. Por outro lado, a diretriz VDI2860

---

[7] *Die Verkettung aller Vorgänge beim Gewinnen, Be- und Verarbeiten, sowie bei der Verteilung von stofflichen Gütern innerhalb festgelegter Bereiche*
[8] *Bearbeiten, Handhaben, Transportieren, Prüfen, Aufenthalt und Lagerung*

(1990) subdivide o fluxo de material apenas em: prover, armazenar e manipular. Assim como essas, outras divergências são encontradas na literatura. Há cerca de duas décadas Auer (1977), Dilling (1978), Michaelis (1982) e Rittinghausen (1980) já descreveram divergências nas terminologias utilizadas na área de fluxo de material. Apesar do aparecimento de novas versões de normas e diretrizes após esta constatação (DIN30781, 1989; VDI2860, 1990; VDI3961, 1989), as divergências não foram sanadas (Thim, 1992). No escopo deste trabalho será adotada a definição de fluxo de material da diretriz VDI3300 (1973), e "transporte" será considerado como sendo a mudança genérica de posição de bens materiais dentro de uma fábrica, desde o depósito de peças em bruto até o depósito de peças prontas.

3.3.2 Manipulação como Função do Fluxo de Material

O termo manipulação é derivado da mão humana (Hesse & Mittag, 1989). Entretanto, com a tendência de automatização das técnicas de fabricação, a manipulação mostrou-se problemática, quando se tentou substituir a mão humana por dispositivos mecânicos (Warnecke & Schraft, 1991). Do ponto de vista técnico, manipulação é definida como sendo "a obtenção, a mudança definida ou a manutenção momentânea de uma localização espacial predefinida de corpos com geometria determinada em um sistema de coordenadas de referência"[9] (VDI2860, 1990). Logo, os limites da manipulação devem ser bem delimitados em relação a outras funções do fluxo de material pela necessidade de existência de corpos com geometria determinada e localização espacial definida. A localização espacial de um corpo em um sistema de coordenadas é dada por sua posição e orientação. A posição de um corpo é o local que um ponto determinado do próprio corpo assume no sistema de coordenadas de referência. A orientação é dada pelo ângulo formado entre os eixos do sistema de coordenadas do corpo e do sistema de coordenadas de referência. Para a descrição da localização espacial de corpos é utilizada a definição de mais três grandezas (Graf, 1984):

- grau de orientação (GO), um número inteiro entre 0 e 3 que quantifica em quantos graus de liberdade rotatórios a orientação do corpo está definida,
- grau de posicionamento (GP), um número inteiro entre 0 e 3 que quantifica em quantos graus de liberdade translatórios a posição do corpo está definida, e

---

[9] *Das Schaffen, definierte Verändern oder vorübergehende Aufrechthalten einer vorgegeben räumlichen Anordnung von geometrisch bestimmten Körpern in einem Bezugskoordinatensystem*

- condição de localização (CL = GO/GP), que quantifica em quantos graus de liberdade a localização espacial (orientação e posição) do corpo está definida (**Figura 3.2**).

Conseqüentemente, um corpo pode estar localizado da seguinte forma (VDI2860, 1990):

- totalmente não localizado CL = 0/0,
- parcialmente localizado 0/0 < CL < 3/3 e
- totalmente localizado CL = 3/3.

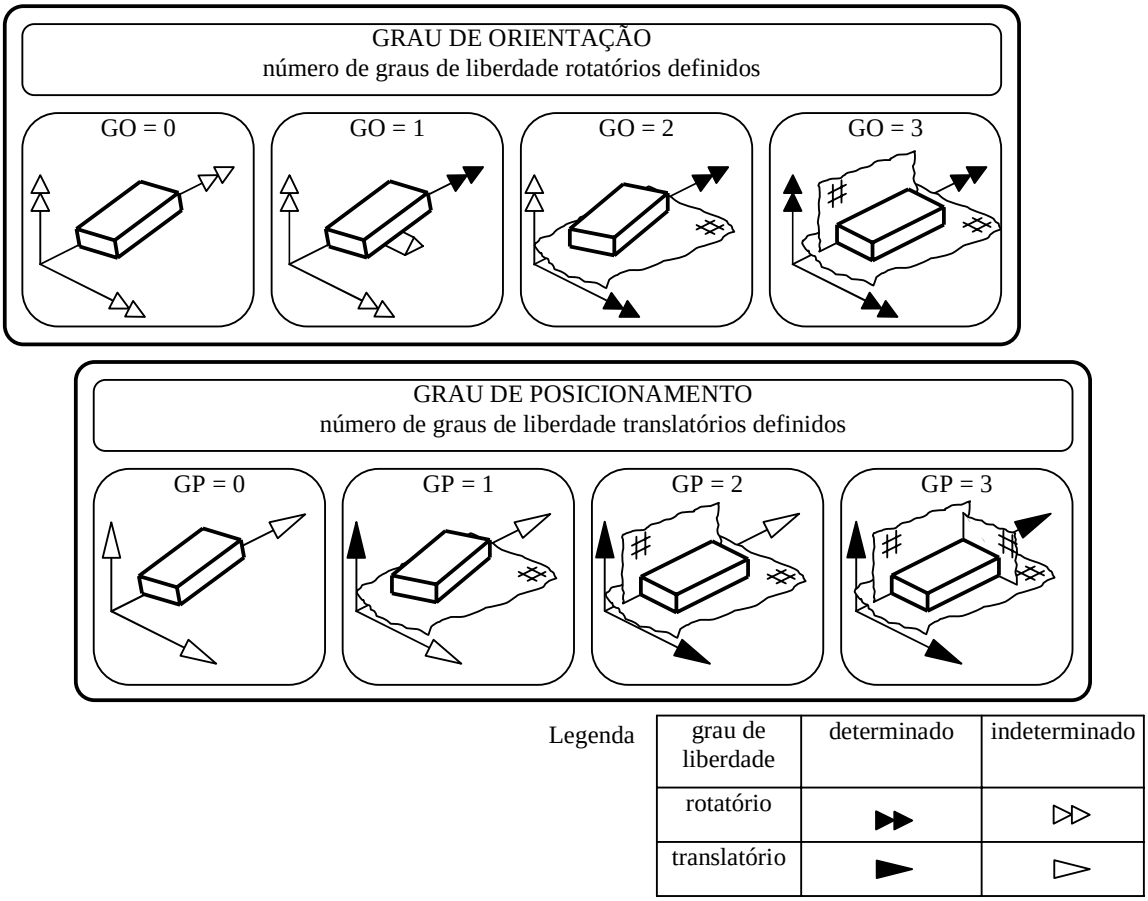


Figura 3.2 – Grau de orientação e posicionamento de um corpo

3.3.3 Subfunções da Manipulação

A função manipulação pode abranger diversas subfunções. A diretriz VDI2860 (1990) subdivide a função manipulação em cinco subfunções, que podem ser classificadas em elementares e combinadas. Subfunções elementares são aquelas que não são mais subdivisíveis de forma significativa. Já, as subfunções combinadas podem ser ainda

subdivididas em elementares (VDI2860, 1990). Das cinco subfunções da manipulação são de importância no escopo deste trabalho a subfunção armazenar (do alemão *Speichern*), movimentar e assegurar (do alemão *Sichern*), que serão tratadas detalhadamente a seguir.

3.3.3.1 Armazenar

Em geral, a subfunção armazenar tem o objetivo de guardar provisões (material, energia ou informação). Em fluxo de material somente a guarda de provisão material é considerada (VDI2860, 1990). A subfunção armazenar, normalmente, é classificada como subfunção da função estocagem do fluxo de material. Para classificá-la como subfunção da manipulação deve-se primeiramente atentar para o fato de que deve se tratar de um corpo com geometria definida. Além disso, a armazenagem do corpo deve ocorrer sob condições de localização pelo menos parcialmente definida. Caso o corpo esteja armazenado em condição totalmente não localizada, então deve ser classificada como subfunção da função estocagem. A subfunção armazenar pode ser dividida em três subfunções combinadas, como mostrado a seguir:

- armazenagem ordenada (CL = 3/3),
- armazenagem parcialmente ordenada (0/0 < CL < 3/3) e
- armazenagem desordenada (CL = 0/0),

sendo que esta última não preenche os pré-requisitos da função manipulação e, portanto, deve ser classificada como estocagem desordenada, uma subfunção da função estocagem. No escopo deste trabalho será tratada apenas a armazenagem ordenada de peças.

3.3.3.2 Movimentar

Movimentar é a subfunção da função manipulação que se destina de forma genérica à mudança da localização espacial dos corpos. Ela é dividida em duas subfunções elementares, que alteram, respectivamente, a orientação e a posição:

Rotacionar — é o movimento de um corpo de uma orientação determinada a outra em torno de um eixo que passa pelo seu ponto de referência. Conseqüentemente, a posição do ponto de referência não muda.

Transladar — é o movimento de um corpo de uma posição determinada a outra por meio do deslocamento ao longo de uma reta. A orientação do corpo não muda.

Além destas são definidas várias subfunções combinadas, que são compostas de diferentes combinações das subfunções elementares. As subfunções combinadas podem ocorrer sob condições de localização inicial e final definidas ou não e o caminho de deslocamento que pode ser predeterminado ou não.

3.3.3.3 Assegurar

Em geral entende-se por assegurar a manutenção de uma condição definida. O assegurar da localização espacial de corpos pode ser duradouro ou momentâneo. Os princípios de atuação possíveis da subfunção assegurar para manter a localização espacial de corpos são: por meio de força, por meio de material ou por meio de forma, que normalmente ocorrem em suas combinações. A subfunção assegurar é dividida em duas subfunções elementares:

Segurar — é o assegurar momentâneo de um corpo em uma determinada orientação e posição,

Soltar — é o inverso do segurar.

Outras duas subfunções combinadas são definidas: fixar é o segurar com a utilização de força mecânica e o afrouxar o seu inverso.

A **Figura 3.3** mostra a classificação dos conceitos apresentados nesta seção. Além da nomenclatura, são apresentados os símbolos utilizados para representar as subfunções em diagramas de fluxo de material. O subdomínio “fluxo de material” engloba muitos outros conceitos que não estão apresentados na figura, pois não são de interesse no escopo deste trabalho.

3.4 Equipamentos e Dispositivos de Manipulação

Os equipamentos e dispositivos de manipulação são aparatos técnicos utilizados para a realização da função manipulação. Ainda que normalmente um equipamento possa realizar várias subfunções da manipulação, uma delas assume a característica de subfunção principal e esta determina a classificação do equipamento (VDI2860, 1990). Os equipamentos de manipulação, assim como os equipamentos de fabricação, transporte, teste e estocagem, devem ser integrados entre si, a fim de alcançar a utilização ótima de sistemas flexíveis de fabricação. Como normalmente os equipamentos utilizados para realizar as demais funções são precedidos e/ou seguidos por equipamentos que realizam funções classificáveis como manipulação, os dispositivos desses equipamentos, ou seja, as interfaces entre esses equipamentos e a peça, também serão tratadas no escopo deste trabalho.

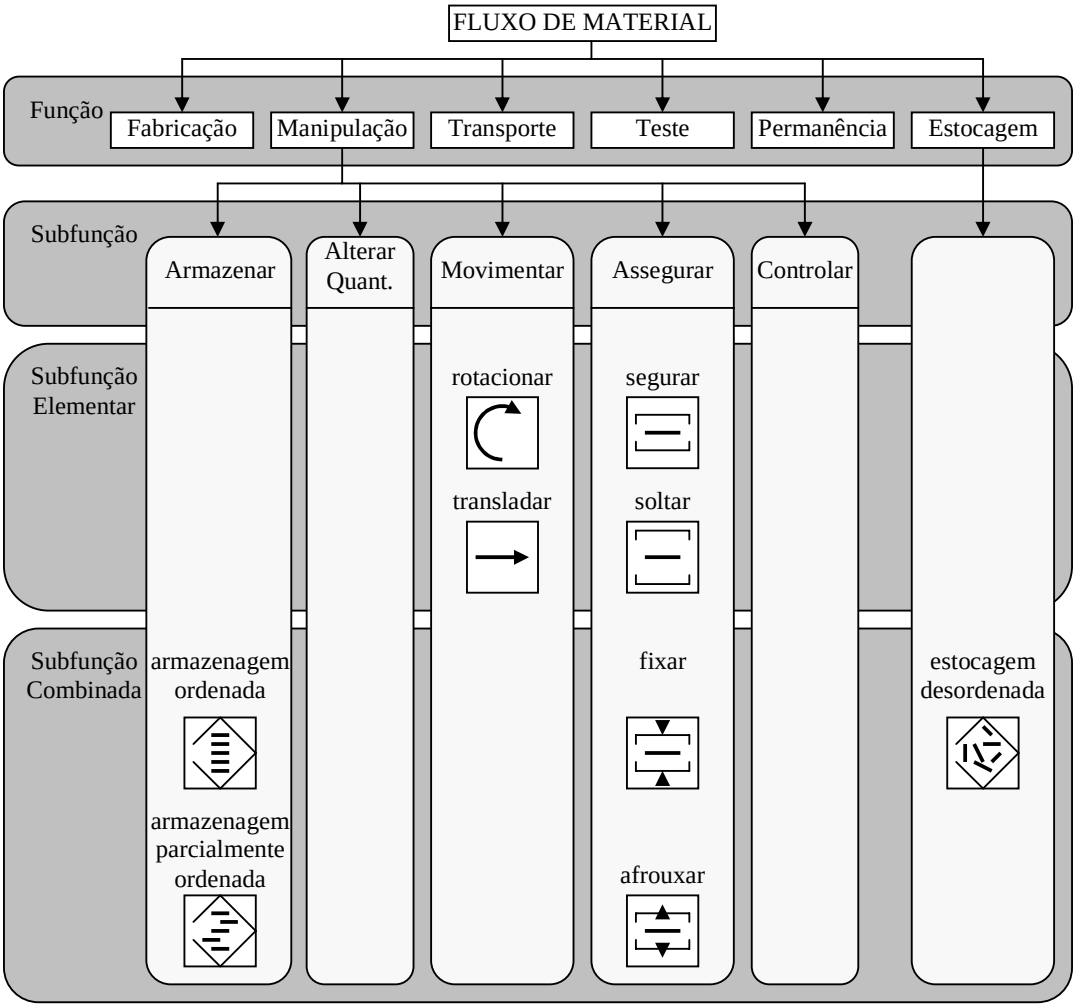


Figura 3.3 – Divisões do fluxo de material

3.4.1 Dispositivos para Armazenar

Os dispositivos para armazenar peças são denominados de diferentes formas pelos diversos usuários e fabricantes. Para descrição de dispositivos do mesmo tipo são utilizados como sinônimos os termos transportador de peça, palete (de transporte), magazine (plano) e palete de máquina, entre outros (Die Sache, 1989; Enzler, 1989). No escopo deste trabalho será adotada a nomenclatura de palete, que é definido como sendo uma plataforma transportável, com ou sem laterais, para agrupamento de bens em uma unidade de carregamento (VDI2411, 1970). Essa definição será aqui utilizada para os diversos termos apresentados anteriormente, desde que os bens agrupados possam ser entendidos como sendo peças. Esses dispositivos podem também ser utilizados para transporte, estocagem ou mesmo para disponibilizar peças, ferramentas, dispositivos de usinagem ou medição.

A história do desenvolvimento dos paletes levou inicialmente à normalização das suas dimensões externas. Da utilização na indústria automobilística surgiu a dimensão de 1000 mm x 1200 mm. Atualmente, o formato conhecido como europeu (800 mm x 1200 mm), assim como as suas subdivisões inteiras, são normalizadas (DIN15141, 1986; VDI2496, 1969).

Na produção seriada e produção em massa utilizaram-se paletes rígidos, adaptados ao produto específico (Bünning, 1990). Entretanto, para a fabricação unitária ou de pequena série, fez-se necessário paletes flexíveis. Na literatura encontram-se vários exemplos que mostram o desenvolvimento deste dispositivo. Há duas décadas construíram-se paletes apropriados às funções de manipulação, compostos de uma placa base, em cima da qual eram fixados, normalmente parafusados, elementos para apoio de peças em formato de discos, principalmente deitadas, mas também em pé (Rittinghausen & Sinning, 1979; Viehweger, 1980; 1981). Com o passar do tempo surgiram também elementos para apoio de peças em forma de eixos, prismáticas e de formas mais complexas (Ganiyusufoglu, 1984; Zipse, 1987).

Um grande passo no desenvolvimento deste dispositivo foi obtido pelo projeto de pesquisa “Armazenamento na Fabricação de Peças Rotacionais” (Uetz & Schuler, 1984). Os paletes desenvolvidos constituem-se de um quadro base e elementos de apoio específicos para cada tipo de peça, que são fixados sob a base. Devido à uniformidade da interface entre o quadro e os diversos elementos de apoio, e também à modularidade, eles são facilmente adaptáveis a peças de diversas formas e dimensões (Schuler & Hardock, 1986; Uetz & Schuler, 1984). Além dos paletes unitários, pode-se agrupá-los para obter unidades de carregamento maiores. Com isso economiza-se tanto o espaço necessário como a quantidade de operações de transporte.

Os paletes construídos de chapas de alumínio ou aço são capazes de disponibilizar peças com precisão de posicionamento na faixa de décimo de milímetro (Bloksma, 1988; Werkstückträger, 1988). Para não ser necessária a utilização da mão de obra economizada com a padronização dos paletes, na preparação dos mesmos, desenvolveram-se também equipamentos para ajustar os elementos de apoio das peças sobre a base (Keller, 1989). Trata-se de um equipamento operado por um único trabalhador, que com o auxílio de um eixo controlado numericamente (CN) é capaz de posicionar os elementos de apoio com precisão de décimo de milímetro sobre a base (Hardock, 1988; Keller, 1989). Pode-se também economizar mão de obra, por meio da montagem automática dos elementos de apoio. Um sistema de paletes direcionado para a montagem automática com a utilização de robôs foi construído de forma que o ponto onde a garra segura o elemento de apoio é também o

acionador do sistema de fixação deles na base (Rüsten, 1990; Weck & Beer, 1991). O sistema de garras necessário para esta operação é composto de duas garras pneumáticas convencionais, que tem a distância entre si controlada por um eixo CN (Weck *et al*., 1990).

3.4.2 Equipamentos para Movimentar

Os equipamentos para movimentação são classificados de acordo com a sua flexibilidade e grau de automação, ou seja, quão necessário se faz uma intervenção manual para alteração do programa de movimentação, ou de outro ponto de vista, quão automática a sua mudança pode ocorrer, em função da utilização de sensores (Schuler, 1987). Os equipamentos são classificados como manipuladores, teleoperadores e robôs industriais, entre outros. Estes são subclassificados de acordo com a flexibilidade para mudança do programa (**Figura 3.4**) (VDI2860, 1990). No tocante à fabricação, os equipamentos de movimentação são empregados principalmente para a movimentação de peças e ferramentas. Na movimentação de peças são empregados para as seguintes tarefas (Fricke, 1985):

- alimentar o sistema de fixação da máquina-ferramenta com uma peça ainda não trabalhada proveniente de um local onde elas estavam disponibilizadas,
- alterar a fixação de peças semi trabalhadas na máquina-ferramenta e
- descarregar a máquina-ferramenta após o término do processo de fabricação.

3.4.3 Dispositivos para Assegurar

3.4.3.1 Placa de Fixação

As peças são asseguradas em posição e orientação definidas durante o processo de fabricação. Como a localização da peça é assegurada sob a ação de forças, deve-se utilizar o termo fixar (VDI2860, 1990). Na área de fabricação com remoção de cavaco, um dispositivo utilizado para a fixação de peças, que permanece parado é chamado de dispositivo de fixação e um dispositivo rotativo é denominado placa de fixação. As placas de fixação são utilizadas principalmente em tornos e algumas vezes em retificadoras. Os dispositivos de fixação são utilizados nas demais máquinas-ferramenta (Ganiyusufoglu, 1985).

As placas de fixação representam a interface mecânica entre a máquina-ferramenta e a peça. Elas são dispositivos rígidos, dificilmente adaptáveis a um grande espectro de peças. Esta adaptação está ligada a custos altos, impossibilitando a utilização economicamente viável

em produções unitárias ou de pequena série. A falta de flexibilidade levou ao desenvolvimento de placas de fixação com troca automática de castanhas com o auxílio de um equipamento de manipulação (Ganiyusufoglu, 1988). As soluções utilizadas distinguem-se na maneira como é feita a troca. As castanhas podem ser trocadas uma a uma ou simultaneamente (Wagner, 1987b).

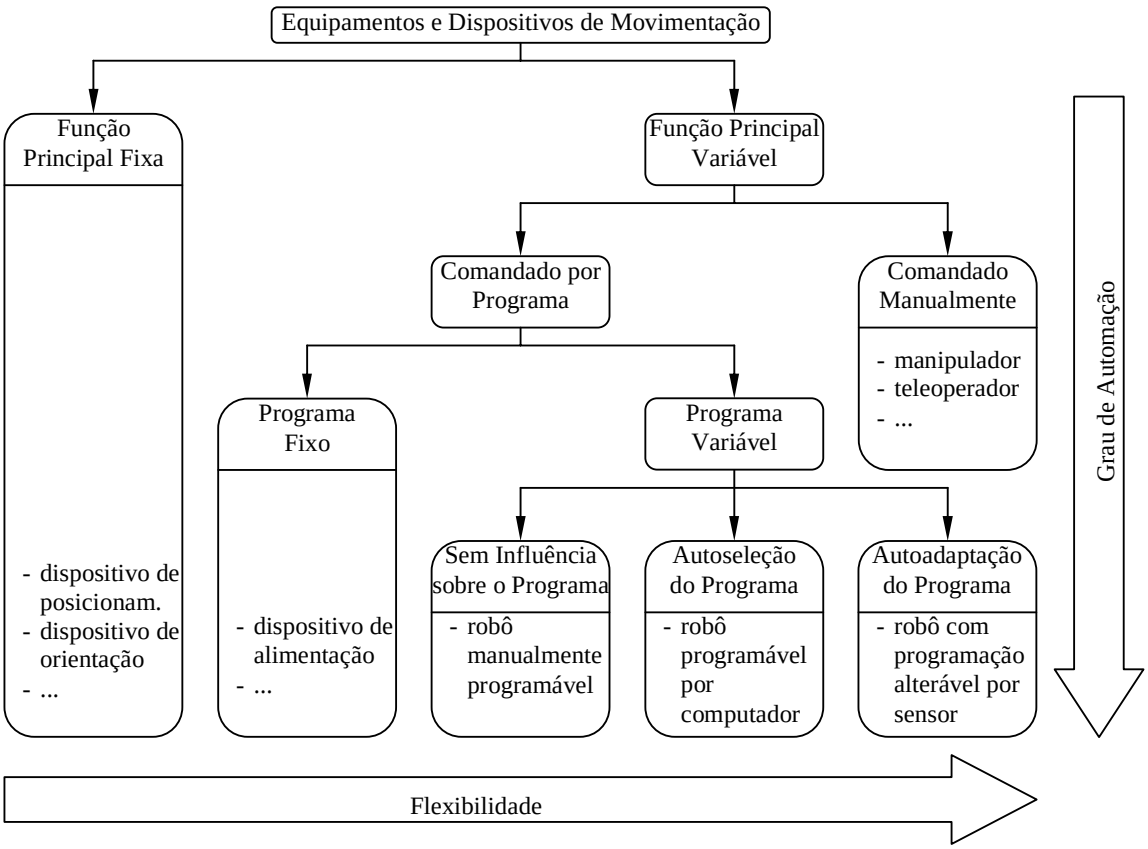


Figura 3.4 – Equipamentos para movimentar (VDI2860, 1990)

Com o objetivo de economizar trocas de castanhas, foram desenvolvidos sistemas em que a superfície de fixação e a de apoio são construídas em peças separadas. A superfície de fixação age no sentido radial e é responsável pela centragem da peça. Com isso obtém-se o posicionamento em dois eixos de translação (GP = 2) e a orientação nos três eixos de rotação (GO = 3). O posicionamento no terceiro eixo de translação é garantido pela superfície de apoio. Normalmente as duas superfícies pertencem a um único elemento. No entanto, caso as superfícies pertençam a elementos distintos, que possam ser comandados separadamente, é de se esperar uma maior flexibilidade. Michaelis (1982) mostra tanto uma construção em que a posição da superfície de apoio pode variar continuamente como uma em que a posição varia

discretamente. Ambas as construções seguem o princípio de comando separado para as superfícies de fixação e apoio.

A placa de fixação completa, com a peça a ser usinada já fixada, pode ser trocada ao invés de trocarem-se as castanhas (Hahner, 1989). Isto leva a um custo de preparação menor, caso a interface entre a placa de fixação e o torno seja prevista para uma troca rápida (Ganiyusufoglu, 1985). A vantagem desse sistema está na possibilidade de realizar a fixação e o afrouxamento de uma peça durante o tempo de usinagem de outra e na precisão de centragem, que é maior que nos sistemas de troca de castanhas. A desvantagem está no custo tanto para a aquisição das diversas placas de fixação como no custo para sua administração e armazenagem.

3.4.3.2 Garra

Atuadores finais são dispositivos para usinar, medir ou segurar, que tem uma interface com um equipamento de movimentação, cujo princípio de atuação seja pelo menos com a utilização de material. Eles são subconjuntos completos, que constituem o elemento de ação entre o equipamento de movimentação, o objeto movimentado e o processo produtivo (Severin, 1987). As garras têm como principal tarefa produzir, manter e desfazer a ligação entre o objeto e o equipamento de movimentação. Para execução dessa tarefa, as garras exercem ou recebem esforços (forças e momentos) (VDI2740, 1991).

Para transmissão de esforços entre a garra e o objeto pode-se utilizar vários princípios físicos e técnicos. Os princípios de fixação mais importantes são: por magnetismo, por engate entre as superfícies, por sucção, hidráulica ou pneumática e mecânica.

Na literatura encontram-se vários exemplos de construções de garras que atuam segundo os diversos princípios (Pham & Heginbotham, 1986; Rosenbauer, 1988; Stave, 1988; Weck & Rosenbauer, 1988). Severin (1987) e Siemens (1983) comentam a preponderância das garras mecânicas, que perfazem cerca de três quartos da totalidade das garras. Em um levantamento de mercado de 1990 constatou-se que em três anos houve um acréscimo de 500 % na quantidade de garras mecânicas, como também um aumento de 50 % no número de seus fornecedores (Dreher & Weisener, 1990).

Independentemente do princípio utilizado, a construção das garras divide-se em cinco subsistemas: de atuação, cinemático, de suporte, de processamento de informações e de acionamento (VDI2740, 1991).

O sistema de atuação abrange o local geométrico da garra que contacta o objeto e conseqüentemente transmite esforços entre suas superfícies (Spur & Severin, 1982). Elas são respectivamente denominadas superfície de fixação ou superfície ativa e superfície fixada ou superfície passiva (Cardaun, 1981). O princípio de atuação para transmissão de esforços pode ocorrer pela forma, pelo emprego de esforços ou combinações deles. Quando o princípio é pela forma ocorre a transmissão de força normal entre as superfícies ativas e passivas. Quando se utiliza o princípio do emprego de esforços tem-se não só força normal como também força de atrito entre as superfícies (Severin, 1987).

O sistema cinemático garante a transmissão de movimento e energia entre o sistema de acionamento e o de atuação da garra. Dependendo da configuração do sistema cinemático pode-se efetuar a transmissão do movimento e dos esforços segundo diferentes relações. A diretriz VDI2740 (1991) sugere um conjunto de soluções para o sistema cinemático a fim de auxiliar o projetista na construção de uma garra.

O sistema de suporte é a base da garra e constitui a interface com o membro final do equipamento de movimentação. Por intermédio desse sistema são transmitidos esforços, energia e informação entre a garra e o equipamento de movimentação. Apesar da normalização do flange para configuração dessa interface (DIN24601, 1986; ISO9409, 1988), constatou-se que de vinte e cinco robôs industriais disponíveis no mercado alemão, apenas quatro possuíam flanges segundo a norma (Buschulte, 1990). Caso haja diversos objetos a serem manipulados ou funções a serem executadas, deve-se prever um sistema de troca de garras que impõe novas exigências ao sistema de suporte. Dependendo da freqüência e da necessidade de automação, os sistemas de troca de garras são construídos em diferentes formas. A troca pode ocorrer com a utilização de ferramentas, manualmente ou totalmente automática. Quando se realiza uma troca automática da garra, deve-se também prever a separação momentânea e a reconexão automática da transmissão de energia e informações. A troca automática de garras não é prevista nas normas acima citadas (Stave, 1988), mas desde o início dos anos 80 vários fornecedores já fabricam regularmente garras com sistema automático de troca (Stolz & Boley, 1988).

O sistema de processamento de informações recebe informações de sensores, as amplifica em um sistema eletrônico e as encaminha ao próximo sistema de processamento, de acordo com uma interface apropriada. O sistema de processamento de informações também representa a interface com o comando do equipamento de movimentação (Severin, 1987).

O sistema de acionamento fornece a energia necessária para a abertura e o fechamento do sistema de atuação, bem como a força de fixação para a manipulação de um objeto. A energia do sistema de acionamento é fornecida na forma elétrica, pneumática ou hidráulica e transformada em energia mecânica dentro do sistema de acionamento.

A **Figura 3.5** apresenta de forma condensada os dispositivos e os equipamentos vistos nesta seção. O subdomínio fluxo de material incorpora muitos outros dispositivos e equipamentos que não estão mostrados na figura, pois não são de interesse no escopo deste trabalho. A figura mostra na parte superior os equipamentos e as funções ou subfunções do fluxo de material que eles podem realizar são apresentadas na parte inferior. Na parte intermediária da figura são mostrados os dispositivos. Observa-se que cada dispositivo está relacionado com um único equipamento (estão desenhados na mesma linha vertical), entretanto, não existe a mesma correspondência entre os dispositivos e as funções. Assim, por exemplo, a placa de fixação corresponde à função fabricação do fluxo de material, pois a placa pertence a um torno que a executa. Simultaneamente, a placa de fixação corresponde também à subfunção assegurar da função manipulação, pois mantém a peça sob a ação de forças em posição e orientação estabelecidas. Da mesma forma, a garra assegura a peça em posição e orientação estabelecidas, sob a ação de forças, mas é utilizada por um robô, um equipamento utilizado para movimentar peças. Analogamente, um palete é empregado para armazenar peças, mas pode estar colocado sobre uma correia transportadora ou em um depósito, que são equipamentos utilizados para desempenhar respectivamente as funções transporte e estocagem.

3.5 Simulação da Função Manipulação

A simulação é caracterizada como a representação de um processo dinâmico em um modelo, para se chegar a resultados que possam ser transferidos para a realidade (VDI3633, 1983). Nos últimos anos, o computador tornou-se o meio mais utilizado para a realização de simulações. Em função disso são apresentadas outras duas definições relacionadas. Um programa de simulação é um programa computacional composto de um grupo de instruções que representa um modelo completo para simular um processo. Um sistema de simulação é definido como um conjunto de programas computacionais, sendo pelo menos um programa de simulação e programas para auxiliar a geração de dados de entrada, a apresentação de resultados e a interface com outros programas (Kuk, 1988).

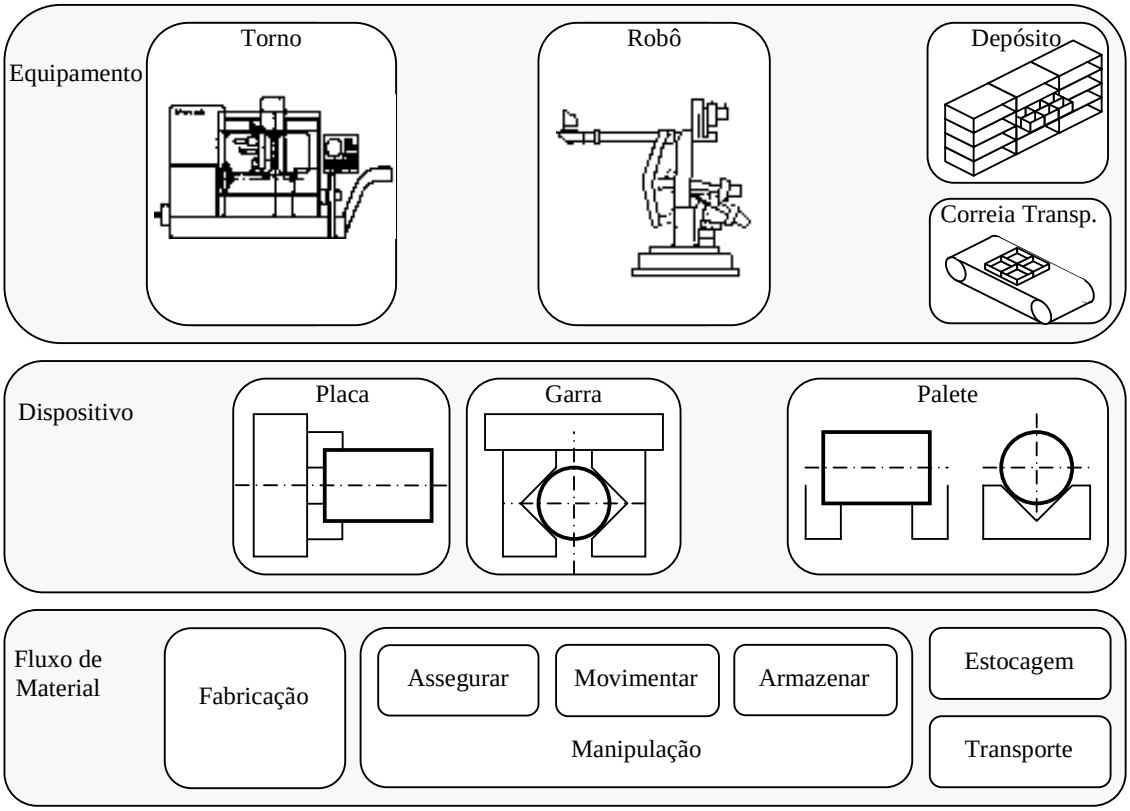


Figura 3.5 – Equipamentos e dispositivos de manipulação

A simulação é empregada em diversas áreas do conhecimento e por diversos motivos. É uma ferramenta que encontra continuamente novas aplicações. Ela pode tornar mais fácil a compreensão de sistemas complexos. Muitas vezes somente com o emprego da simulação podem-se dominar as dificuldades inerentes aos problemas (Wrba, 1990). Normalmente ela é realizada antes do processo real e raramente durante o decorrer deste. Em casos de ocorrência de perturbações ela auxilia na busca de alternativas. Com isso a simulação tornou-se parte importante de sistemas de diagnose (Weck & Kohen, 1986).

Com a utilização de computadores pode-se simular o desenvolvimento de um processo com uma duração qualquer, porém sempre em intervalos discretos de tempo. O mais importante aqui é a simulação de pontos da função manipulação propriamente dita, bem como das interfaces entre esta e as demais funções do fluxo de material. Partindo-se deste ponto de vista, a seguir discutem-se sistemas de simulação já desenvolvidos. Primeiramente são apresentados de acordo com a subfunção da manipulação que é simulada e em seguida tratados de forma conjunta.

3.5.1 Simulação da Subfunção Armazenar

Os programas computacionais para escolha e disposição de elementos de apoios para peças em paletes de transporte tiveram um desenvolvimento semelhante ao desenvolvimento do próprio palete. Inicialmente, tanto a capacidade como a flexibilidade eram limitadas. Michaelis (1982) analisou elementos de apoio para peças com simetria rotacional. Baseado em um espectro e em um princípio de apoio das peças predeterminados ele determinou a possibilidade de colocação da peça sobre o palete levando em consideração tanto parâmetros técnicos de usinagem como geométricos.

Rittinghausen (1980) e Rittinghausen e Sinning (1979) mostram um sistema de programas para a escolha e a disposição de elementos de apoio para peças em forma de disco na posição deitada. Um sistema semelhante foi também desenvolvido por Viehweger (1980; 1981). Graf (1984) considerou diversas distribuições de peças sobre um palete levando em consideração principalmente a possibilidade de atuação de uma garra, objetivando a manipulação automática. Zipse (1987) desenvolveu um sistema capaz de realizar um tratamento mais preciso dos elementos de apoio, como também da possibilidade de colocação de peças diferentes sobre o mesmo palete. As peças com simetria rotacional e prismáticas eram simplificadas, pois ele utilizava cilindros e prismas regulares parametrizados para representá-las.

Um sistema de simulação inovador para configuração de paletes de transporte é mostrado por Weck e Beer (1991). Devido à utilização de elementos de apoio manipuláveis, pode-se gerar em paralelo com a determinação da disposição dos elementos de apoio, tanto as instruções de comando para o robô que efetua a sua montagem, como para aquele que posteriormente irá colocar a peça ou irá retirá-la do palete de transporte.

3.5.2 Simulação da Subfunção Movimentar

Para a simulação dessa subfunção da manipulação deve-se imaginar a movimentação de objetos, a fim de poder determinar e também verificar o decorrer dessa movimentação. De forma primitiva o projetista esquematiza vários posicionamentos espaciais do objeto a ser movimentado, para testar a exeqüibilidade do movimento, o que, sob certo ponto de vista, pode ser considerado uma simulação da movimentação. Entretanto, quando se trata da simulação da utilização de equipamentos automáticos de manipulação, defronta-se rapidamente com os limites desse método (Wrba, 1990).

Para a simulação de movimentos mais complexos utilizam-se incondicionalmente os computadores. A maior parte dos sistemas comerciais de simulação para robôs industriais foi desenvolvida como módulos acoplados a sistemas CAD (Osterwinter, 1991). Assim, boa parte do esforço de modelagem estava solucionada. A introdução no mercado dos sistemas de simulação, como módulo de um sistema mais complexo, facilitou a sua aceitação (Wrba, 1990). O sistema CATIA da Dassault/IBM, por exemplo, permite a simulação da movimentação de robôs após a definição da tarefa a ser realizada (IBM, 1986).

Além da simulação e da programação da movimentação de robôs, a monitoração da possibilidade de colisões é uma tarefa importante dos sistemas de programação de robôs. Colisão é o encontro espacial e temporal involuntário de dois corpos, sendo que pelo menos um deles, no instante do encontro, tem uma energia cinética não nula (Stöck, 1986; Weck & Stöck, 1985). São utilizados diversos métodos para controle de colisão durante o transcorrer de movimentações, os quais podem ser classificados em métodos *online* e *offline*. A monitoração *online* ocorre durante a movimentação do robô. Caso seja determinada uma possibilidade de colisão, então, de acordo com a capacidade do sistema, ou o robô é retido, ou gera-se um novo caminho de movimentação, onde não ocorrerão colisões. A monitoração *offline* verifica a movimentação programada do robô antes de sua execução, com o auxílio de um modelo computacional do robô e de seu ambiente de trabalho (Schütze, 1988). Para minimizar o esforço computacional, a monitoração *offline* de colisões de corpos em movimento é dividida em vários passos. Enquanto é reconhecida a colisão, definem-se sucessivamente envelopes de controle (do inglês *box-test*) para o corpo, para as suas superfícies e para as suas arestas, que são utilizados nos testes de colisão. Somente então se realiza um teste preciso de colisão, utilizando a geometria exata dos corpos (Moser, 1991; Schütze, 1988).

3.5.3 Simulação da Subfunção Assegurar

3.5.3.1 Simulação da Fixação com Placas

Os sistemas utilizados para simular o comportamento das placas de fixação podem ser divididos em dois grupos. Por um lado compara-se a geometria da placa de fixação e das castanhas com a geometria da peça, a fim de determinar a possibilidade de fixação. Assim, pode-se determinar o comprimento e o diâmetro de fixação, o comprimento de apoio, etc., como também uma possível obstrução do processo de torneamento, que pode ocorrer, por

exemplo, devido à ocultação pelas castanhas da superfície a ser trabalhada. Michaelis (1982) trabalhou com esse ponto de vista e determinou, em função de um espectro de peças, a parcela de fixações realizáveis pelos elementos de fixação. Por outro lado, analisa-se a força de fixação na região de contato entre castanha e peça, bem como a sua alteração em função da rotação. Devido à tendência de utilizarem-se velocidades de corte cada vez maiores nos processos de usinagem com remoção de cavaco, a rotação máxima do eixo-árvore dos tornos aumentou consideravelmente nos últimos anos. Com isso aumentou-se a força centrífuga sobre as castanhas, que por conseqüência diminui a força de fixação sobre a peça (Spur *et al.*, 1993, Spur & Mette, 1998; Steinberger, 1989; VDI3106, 1983; Wagner, 1987a; 1987b). A diretriz VDI3106 (1983) propõe que toda força centrífuga seja vista como redução da força de fixação. Isso pressupõe uma rigidez infinita da peça, que na realidade é finita.

A diretriz VDI3106 (1983) divide as possibilidades de contato em: no meio, no canto da castanha e completo, em que o raio de curvatura da castanha é respectivamente maior, menor e igual ao raio da peça. Wagner (1987b) adota, respectivamente, uma distribuição de pressão de contato elíptica, constante e parabólica. Com isso, ele pôde determinar a rigidez de contato e da peça e comparar com valores experimentais. Por meio da utilização do sistema de medição de força de fixação FORSAVE-D da firma Forkardt (1983) pode-se determinar a curva da força de fixação. Inicialmente, a curva de força de fixação é medida com um sensor de força e armazenada. Introduzindo-se os valores de rigidez da placa de fixação, do sensor e da peça a ser usinada, determina-se a curva da força de fixação e o limite de rotação da placa de fixação referenciados à peça.

Noske (1991a) criticou esse sistema, pois a força de fixação não pode ser medida durante o processo de usinagem, além da dificuldade de medição associada à determinação da força de fixação. Em contrapartida, ele mostra um sistema baseado em extensômetros elétricos (do inglês *strain gages*) e transmissão indutiva, sem contato, dos sinais. Os componentes eletrônicos utilizados são pequenos o suficiente para serem integrados na própria castanha. Finalmente, ele mostra um sistema de monitoramento prático e barato, que determina indiretamente o atrito entre a castanha e a peça por meio da medição da diferença de pressão no processo de fixação (Noske, 1991b).

3.5.3.2 Simulação da Fixação com Garras

De forma semelhante à simulação da fixação com placas, a simulação do emprego das garras é subdividida por um lado, na adaptabilidade geométrica entre o sistema de atuação e a

peça e, por outro lado, na distribuição de pressão nas superfícies ativa e passiva de fixação. Michaelis (1982) mostrou, comparativamente, em função da largura da pinça da garra, da utilização do comprimento da garra, da força realizada pela garra e da região disponível sobre a peça para emprego da garra, a parcela de um espectro de peças de formas variadas que podiam ser manipuladas. Entretanto, ele não verificou possíveis colisões entre o corpo da garra e a peça. Um processo típico de um robô inicia com a fixação do objeto a ser manipulado, na garra. Os movimentos necessários do robô dependem dos possíveis pontos de fixação sobre a peça. Durante a escolha de uma posição para fixação, precisa-se levar em consideração que tanto essa posição como o caminho a ser percorrido precisam ser realizáveis (Weeks, 1988). Como conseqüência a simulação do processo de fixação com garras, também denominado de planejadores do emprego de garras, tornou-se parte importante dos sistemas de programação de robôs. O funcionamento de um planejador do emprego de garras é mostrado em Weck e Weeks (1991). Inicialmente, procuram-se as superfícies, para as quais é possível um contato com a pinça da garra. Em seguida examinam-se a posição e a orientação relativa dessas superfícies. Elas devem estar localizadas da forma como a garra necessita. Depois se determina a área superficial da região de contato entre a pinça da garra e a peça. Finalmente, escolhe-se o ponto para fixação com a garra entre os diversos possíveis, que entre outros fatores, considera também a distância até o centro de gravidade da peça.

3.5.4 Resumo Crítico da Simulação da Manipulação

Assim como os equipamentos de manipulação, os sistemas de simulação também são subdivididos segundo a subfunção simulada, que é considerada principal. Porém, em parte, a maioria dos sistemas pode simular outras subfunções. Enquanto que o sistema desenvolvido por Graf (1984) determina o preenchimento de paletes de transporte, no qual a possibilidade de fixação da peça com a garra é representada apenas pelo espaço necessário, outros sistemas calculam a posição exata onde a garra deve fixar a peça (Weck & Beer, 1991). A simulação da fixação com garra é freqüentemente integrada em um sistema de programação de robôs, cujo objetivo principal é a simulação da sua movimentação. Normalmente, a simulação da utilização de placas de fixação é parte integrante de um sistema de programação comando numérico.

Michaelis (1982; 1984) analisou elementos de apoio de peças, placas de fixação e garras para peças com simetria rotacional e relacionou os espectros de peças com a possibilidade de apoio e fixação com placas e garras. Entretanto, a seqüência de fabricação era predeterminada

e ele não podia exercer nenhuma influência. Para o controle de colisão entre o dispositivo utilizado na manipulação e os demais dispositivos ele utilizou apenas a superfície de atuação sobre a peça. A geometria de cada um dos dispositivos foi substituída pelo espaço da peça necessário para emprego de cada um deles. Assim, ele não pôde verificar, por exemplo, a colisão entre a garra e a placa de fixação.

Não são conhecidas soluções que considerem as interfaces mecânicas entre os dispositivos e a peça a ser produzida durante a realização das subfunções da manipulação em sistemas flexíveis de fabricação e que possam ter influência sobre a folha de processo e, conseqüentemente, sobre a ordem de fabricação e a programação de alocação da máquina.

Portanto, para poder utilizar completamente a flexibilidade dos sistemas flexíveis de fabricação, devem ser desenvolvidos métodos que, após a análise da seqüência de operações necessárias à fabricação de uma peça, sejam capazes de apontar alternativas viáveis para sua produção ou dar indicações de alterações a serem implementadas no sistema produtivo, a fim de possibilitar a produção da peça sem que haja a necessidade de paralisação desses sistemas. No escopo deste trabalho só será considerada a manipulação de peças com simetria rotacional.

3.6 *Frameworks* para Manufatura

Fayad e Johnson (2000) publicaram um livro sobre *framework* de aplicação de domínios específicos. Duas partes desse livro são dedicadas ao domínio da manufatura. Na segunda parte são descritos alguns *frameworks* que não estão diretamente ligados ao escopo deste trabalho. Entretanto, eles são aqui sucintamente comentados para uma melhor compreensão da abrangência do domínio da manufatura.

Yang e Mehta (2000) desenvolveram um *framework* para engenharia simultânea (*Concurrent Engineering Framework* – CEF). O desenvolvimento de um produto inicia-se na sua concepção, passa pelas fases de projeto e manufatura, até a fase de marketing e vendas, entre outras. A engenharia simultânea afasta-se da abordagem seqüencial normalmente utilizada na realização destas fases e busca uma maior integração entre elas. Como em cada fase, diversas informações sobre o produto já estão definidas antes que a fase esteja encerrada, a engenharia simultânea libera essas informações para a fase seguinte, a fim de reduzir o tempo total de desenvolvimento do produto.

O *framework Production Resource Manager* (PRM) (Dietrich *et al.*, 2000) foi desenvolvido para gerenciar a logística de desenvolvimento de produtos entre o fabricante e

seus fornecedores. Na Figura 3.1 apresenta-se o subdomínio Planejamento e Controle da Produção, que abrange a logística externa à fábrica.

Cham e Lammers (2000) desenvolveram um *framework* e implementaram uma aplicação para o processo de monitoramento e diagnóstico e Bosch (2000) criou um *framework* para sistemas de medições. Normalmente estas atividades são alocadas ao processo de controle de qualidade, sendo que o monitoramento preocupa-se em coletar dados dos equipamentos e os sistemas de medição atuam diretamente sobre a peça produzida. Os subdomínios em que esses *framework* estão classificados podem também ser vistos na Figura 3.1.

Além dos *frameworks* de aplicação geral na manufatura, Fayad e Johnson reúnem na primeira parte de seu livro aqueles utilizados para o controle de sistemas flexíveis de fabricação. Coincidentemente, existe uma ligação entre a produção de software e a produção de equipamentos. A produção de máquinas-ferramenta com controle numérico (CN) até a década de 80 foi liderada pelo Japão, Alemanha, União Soviética, Estados Unidos e Itália (Weck, 1991). Como regra, o software de controle da máquina era, e ainda é, fornecido pelo próprio fabricante. Entretanto, quando se integram máquinas-ferramenta e outros equipamentos em sistemas de fabricação, normalmente o software de integração precisa ser desenvolvido. A partir de 1990, em conseqüência da decadência do regime comunista na União Soviética, este país, que era um grande produtor, mas um exportador restrito ao bloco comunista, teve sua produção altamente reduzida. O Japão, por sua vez, é um país de pouca tradição na produção de software. Em outra publicação reunindo artigos sobre *frameworks* para o domínio da manufatura, na *Communications of the ACM* de outubro de 1997, observa-se que as três linhas de trabalho mostradas por Fayad e Johnson também estão presentes. Tratam-se de um trabalho da escola italiana, um da escola americana e um da escola alemã, que também são grandes produtores de máquinas-ferramenta.

O *framework* desenvolvido por Doscher e Hodges (1997; 2000), denominado SEMATECH (tecnologia de manufatura de semicondutores, do inglês ***Se**miconductor **Ma**nufacturing **Tech**nology*) é aplicável à Manufatura Integrada por Computador (CIM). O *framework* visa ao desenvolvimento de aplicações para controle da manufatura de semicondutores. Boyle (2000) realiza um desenvolvimento baseado no *framework* SEMATECH para dar suporte às mudanças freqüentes e complexas que ocorrem na manufatura de semicondutores. Entretanto, a manufatura específica de semicondutores afasta-se do escopo deste trabalho e não será tratada de forma detalhada. A seguir são apresentados

os dois outros *frameworks* da escola alemã e italiana para o subdomínio dos sistemas flexíveis de fabricação na indústria metal-mecânica. O primeiro recebe o nome de OSEFA. O segundo é baseado em uma linguagem de padrões denominada G++.

3.6.1 *Framework* OSEFA

Schmid (1997; 1999) elaborou uma técnica para o desenvolvimento de *frameworks* baseado em generalização sistemática (seção 2.4.3.2) dos subsistemas de pontos variáveis (seção 2.3). Utilizando essa técnica ele construiu um *framework* para o domínio Sistema Flexível de Fabricação, denominado OSEFA (componentes de software orientados a objeto para instalações produtivas, do alemão ***O****bjektorientierter* ***S****oftwarebaukast****E****n für* ***F****ertigungs****A****nlagen*), documentado em diversas publicações (Schmid, 1995; 1996a; 1996b; 2000a; 2000b; Schmid & Mueller, 1998).

O primeiro passo da generalização sistemática consiste em criar um diagrama de classes para uma aplicação específica do domínio no qual pretende-se desenvolver o *framework*, utilizando as técnicas usuais de análise e projeto orientados a objetos. A **Figura 3.6** mostra o diagrama de classes da aplicação convencional criada para controle da célula de manufatura localizada na Faculdade de Konstanz, Alemanha, composta por um torno, um robô de pórtico, um depósito (do inglês *store*) e um armazém (do inglês *buffer*) próximo ao torno. A lógica da aplicação desenvolvida para controle da célula de manufatura é decomposta em passos e pode ser vista na **Figura 3.7**. Os passos mostrados na figura correspondem à seqüência de processamento das ordens de fabricação. O sistema, na realidade, é mais complexo, pois cada ação só é realizada após a ocorrência de um evento notificando o fim da ação anterior. As seqüências de processamento são otimizadas, permitindo a simultaneidade entre transporte de paletes e o processamento da peça no torno.

Ao contrário de uma aplicação específica, um *framework* deve proporcionar uma série de variabilidades para poder ser utilizado no desenvolvimento do maior número possível de aplicações do domínio. Os requisitos de variabilidade para diferentes configurações de sistemas flexíveis de fabricação são apresentados a seguir:

- O número, a variedade e o tipo de máquina-ferramenta;
- O número, a variedade e o tipo de equipamento de manipulação;
- A associação entre o equipamento de manipulação e a máquina-ferramenta;
- A geometria, a topologia e o tipo de sistema que move os paletes;
- A combinação dos sistemas de transporte e de manipulação;

- A geometria, a topologia e o tipo de armazém;
- A geometria e o tipo de palete e longarina.

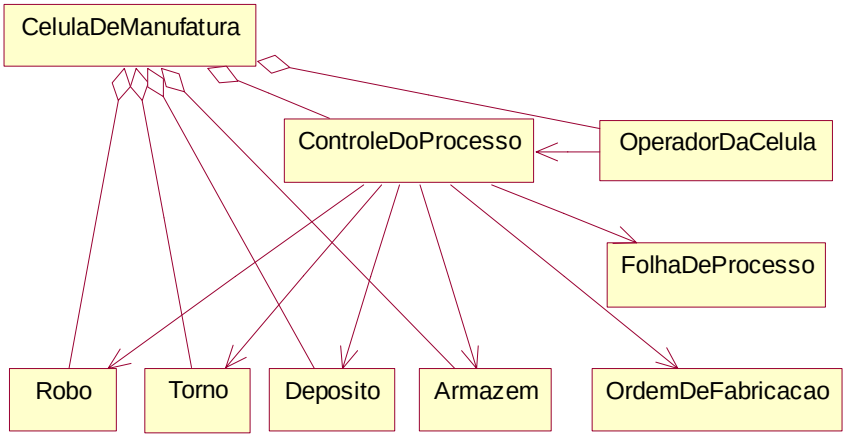


Figura 3.6 – Diagrama de classes de uma célula de manufatura (Schmid, 1996b)

0- Ler as Ordens de Fabricação e as Folhas de Processo correspondentes, preparar as máquinas e equipamentos com dispositivos, ferramentas e programas CN.

Repetir os passos 1 a 5 até que todos os paletes de uma Ordem de Fabricação sejam processados.

1- Mover o palete com peças em bruto do depósito para o armazém.

Repetir os passos 2 a 4 até que todas as peças do palete sejam processadas.

2- Carregar uma peça em bruto do palete no armazém para o torno.

3- Processar a peça-obra no torno.

4- Descarregar a peça pronta do torno para o palete.

5- Mover o palete com peças prontas do armazém para o depósito.

Figura 3.7 – Lógica de uma aplicação específica (Schmid, 1996b)

Quando é desenvolvida a aplicação para uma célula de manufatura em particular, a variabilidade global é fixada. Em contraste, a configuração local pode ser alterada em função da ordem de fabricação a ser processada. Por exemplo, uma ordem de fabricação pode requerer o uso de um tipo de palete para transportar tanto peças em bruto como peças prontas ou dois tipos de palete, cada um com diferentes tipos de longarina para cada estado da peça. Conseqüentemente, a seqüência de processamento de uma ordem de fabricação pode ser ligeiramente diferente daquela mostrada na Figura 3.7.

Deve-se analisar e estruturar as variabilidades descritas na lista de variáveis estruturais do domínio e identificar os aspectos variáveis elementares que formam os pontos variáveis

(Schmid, 1999). Um diagrama de pontos variáveis reúne-os de acordo com os seus níveis de abstração. Alguns pontos variáveis correspondem diretamente à variabilidade do domínio, enquanto que outros cobrem apenas parte ou algumas variabilidades. Os pontos variáveis estão representados na **Figura 3.8** por caixas de cor branca, enquanto que as caixas de cor mais escuras representam os pontos fixos do domínio.

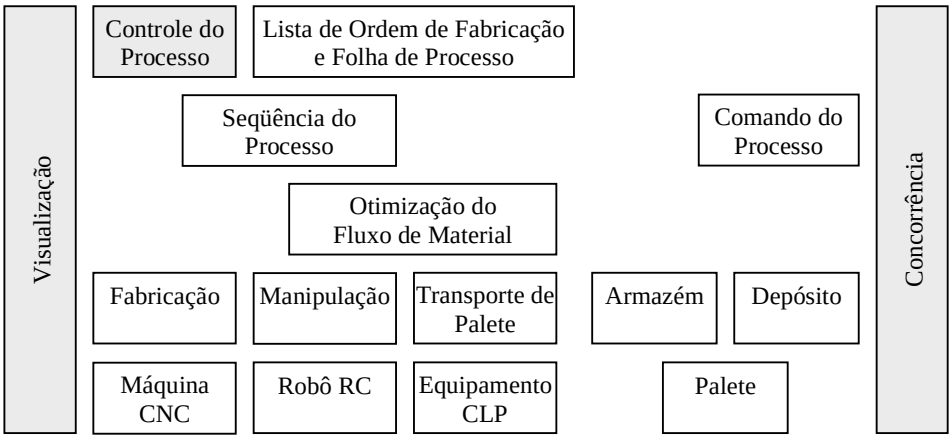


Figura 3.8 – Diagrama de pontos variáveis estruturado de acordo com o nível de abstração (Schmid, 2000a)

Os pontos variáveis Máquina CNC, Robô RC, Equipamento CLP e Palete representam a variabilidade tanto do tipo de máquina como do equipamento de manipulação ou de transporte. Os pontos variáveis Fabricação, Manipulação e Transporte de Palete representam responsabilidades específicas do domínio, correspondente às ações realizáveis pelas máquinas ou equipamentos do nível inferior. O ponto variável Otimização do Fluxo de Material permite que diferentes estratégias de otimização possam ser utilizadas, quando a célula de manufatura é composta por diferentes tipos de máquinas e equipamentos. Os pontos variáveis Armazém e Depósito admitem o uso de diferentes equipamentos para realizar, respectivamente, a armazenagem ou a estocagem. O ponto variável Seqüência do Processo permite o uso de diferentes fluxos de material e seqüências de processamento de peças de acordo com a configuração específica. O ponto variável Comando do Processo admite comandos internos, que permitem, por um lado, que uma requisição a um serviço possa variar com a evolução do *framework* e, por outro, que a estruturação do software seja melhorada. O ponto fixo Controle do Processo implementa o controle genérico de funcionamento da célula de manufatura (Schmid, 1996a; 2000a).

Um *framework* é estruturado em mais camadas do que uma aplicação específica, pois os pontos variáveis introduzem novos conceitos generalizados e, conseqüentemente, novas

camadas na estrutura de classes. A aplicação específica representada pelo diagrama de classes da Figura 3.6 é estruturada em duas camadas: uma camada superior constituída pelo Controle do Processo e uma camada inferior constituída pelas máquinas e equipamentos. A camada superior implementa de forma concreta tanto o fluxo de material como a seqüência de processamento das peças e a camada inferior recebe requisições de serviços da camada superior. A camada inferior é dividida em duas subcamadas. A subcamada superior contém as classes que modelam as máquinas e os equipamentos e disponibilizam a interface para a camada superior. A subcamada inferior é responsável pelo acoplamento dos objetos da subcamada superior com as máquinas e os equipamentos reais via uma linha de comunicação.

A lógica da aplicação com arquitetura de duas camadas é concreta, pois consiste basicamente da seqüência de requisições realizadas diretamente às máquinas e aos equipamentos. A lógica torna-se independente das máquinas e dos equipamentos se as requisições forem mais abstratas. A requisição de serviços de forma abstrata é propagada através da estrutura de camadas do *framework* até a máquina ou o equipamento correspondente por meio dos pontos variáveis. A estrutura de camadas do *framework* resultante pode ser vista na **Figura 3.9**.

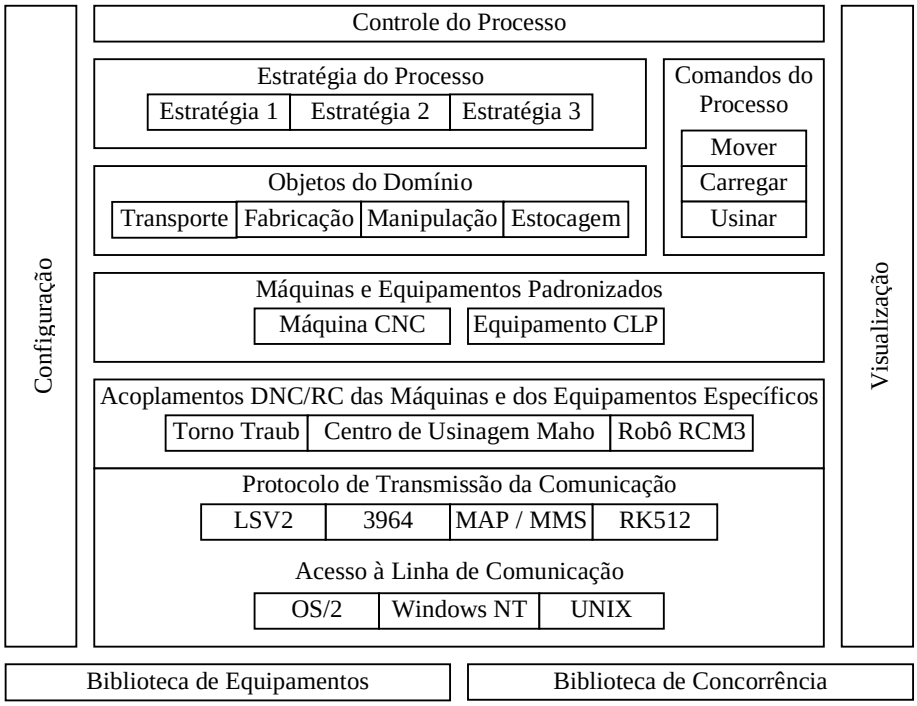


Figura 3.9 – Estrutura de camadas do *framework* OSEFA (Schmid, 2000a)

A camada Controle do Processo, que contém o ponto fixo de mesmo nome e o ponto variável para acesso à Lista de Ordem de Fabricação e Folha de Processo, é subdividida para disponibilizar a subcamada Estratégia do Processo que contém o ponto variável Seqüência do Processo. A camada Objetos do Domínio contém os pontos variáveis Otimização do Fluxo de Material, Fabricação, Manipulação e Transporte de Palete. Ela recebe requisições de serviços abstratos e propaga-as para a camada inferior. A camada Máquinas e Equipamentos Padronizados contém os pontos variáveis Máquina CNC, Robô RC, Equipamento CLP e Palete. Ela recebe requisições por serviços padronizados e propaga-as como requisições específicas às máquinas e aos equipamentos reais. A camada Máquinas e Equipamentos Específicos contém as máquinas e os equipamentos reais. Na subcamada Comunicação estão contidos os pontos variáveis Protocolo de Transmissão da Comunicação e Acesso à Linha de Comunicação, não mostrados na Figura 3.8. A camada Máquinas e Equipamentos Específicos recebe requisições por serviços específicos e propaga-as até a máquina ou o equipamento real. A propagação das requisições ocorre via a subcamada de Comunicação utilizando o Protocolo de Comunicação que tem Acesso à Linha de Comunicação.

A arquitetura de camadas do *framework* é especificada pelo diagrama de classes mostrado na **Figura 3.10**. O diagrama é o resultado da aplicação da técnica de generalização sistemática proposta por Schmid. O diagrama foi obtido por meio de várias transformações realizadas na estrutura de classes, cada uma delas para atender à variabilidade dos subsistemas de pontos variáveis. A **Tabela 3.1** resume as transformações realizadas para obter-se o diagrama de classes, os pontos variáveis ou as camadas atingidas, as classes acrescentadas e os padrões de projeto utilizados. Maiores detalhes sobre as transformações podem ser vistos nas publicações de Schmid (1995; 1996a; 1996b; 2000a) ou de forma condensada em Massaroppi (1998).

Como não existe uma correspondência biunívoca entre os pontos variáveis do domínio (Figura 3.8) e as camadas da arquitetura do *framework* (Figura 3.9), na Tabela 3.1 foram utilizadas as letras “P” (ponto variável) ou “C” (camada da arquitetura) para indicar qual deles estava sendo atingido com a transformação. As classes acrescentadas colocadas em *itálico*, representam classes abstratas. No diagrama de classes da Figura 3.10 também estão marcadas as classes que foram acrescentadas em cada uma das transformações.

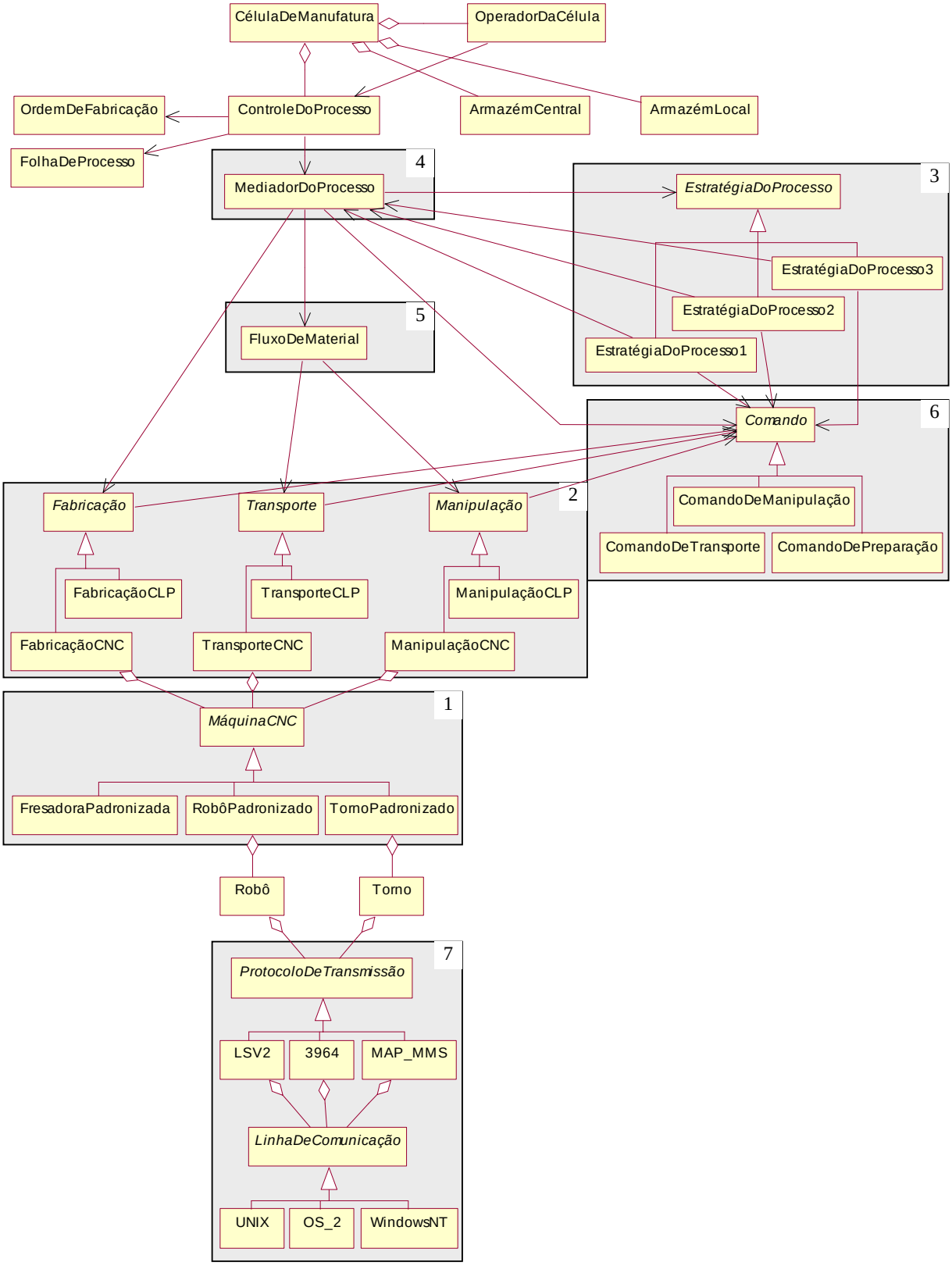


Figura 3.10 – Diagrama de classes do *framework* OSEFA

Tabela 3.1 – Transformações do diagrama de classes do *framework* OSEFA

Transformação		(Camada / Ponto Variável)	Classes Acrescentadas	Padrão Utilizado
1	Generalização dos dispositivos	(C – Máquinas e Dispositivos Padronizados)	*MáquinaCNC* TornoPadronizado RobôPadronizado FresadoraPadronizada	Adaptador
2	Independência dos dispositivos	(C – Objetos do Domínio)	*Transporte* *Fabricação* *Manipulação* TransporteCNC TransporteCLP FabricaçãoCNC FabricaçãoCLP TransporteCNC TransporteCLP	Adaptador
3	Extração da lógica da aplicação	(C – Controle do Processo)	*EstratégiaDoProcesso* EstratégiaDoProcesso1 EstratégiaDoProcesso2 EstratégiaDoProcesso3	Estratégia
4	Intermediação entre objetos	(C – Comando do Processo)	MediadorDoProcesso	Mediador
5	Otimização da requisição de serviços	(P – Otimização do Fluxo de Material)	FluxoDeMaterial	Otimizador
6	Criação de objetos para requisição de serviços de transporte e similares	(C – Controle do Processo)	*Comando* ComandoDeTransporte ComandoDeManipulação ComandoDePreparação	Comando
7	Mecanismo de comunicação	(C – Máquinas e Dispositivos Específicos)	*ProtocoloDeTransmissão* LSV2 3964 MAP_MMS *LinhaDeComunicação* UNIX OS_2 WindowsNT	Herança de Interface

3.6.2 *Framework* Baseado na Linguagem de Padrões G++

O grupo de trabalho do Departamento de Automação e Informática do Politécnico de Turim, Itália, desenvolveu uma linguagem de padrões denominada G++, documentada em diversas publicações (Aarsten *et al*., 1995; 1996a; 1996b; 2000; Brugali, 1998; Brugali & Menga, 1999), aplicável ao desenvolvimento de *frameworks* para sistemas distribuídos reativos, tal qual para um sistema flexível de fabricação (Aarsten *et al*., 1995; 2000; Brugali, 1998; Brugali *et al*., 1997; 2000; Brugali & Menga, 1999).

Uma linguagem de padrões é uma coleção estruturada de padrões. A estrutura de padrões é ordenada de tal modo que os padrões gerais ou de alto nível são apresentados antes dos padrões mais específicos. A ordem é definida de acordo com a dependência entre os padrões (Brugali, 1998).

A linguagem de padrões G++, apresentada nas publicações do grupo, evoluiu ao longo do tempo. No escopo deste trabalho será comentada apenas a versão apresentada mais recentemente (Aarsten *et al.*, 2000). A linguagem de padrões é estruturada, segundo Alexander *et al.* (1977), como a árvore mostrada na **Figura 3.11**. Cada círculo numerado representa um padrão e, conseqüentemente, um ponto de decisão de projeto, e as setas que os unem representam a seqüência temporal de decisão. Assim, o processo de desenvolvimento é atingido ao percorrer a árvore da raiz até as folhas (Brugali, 1998).

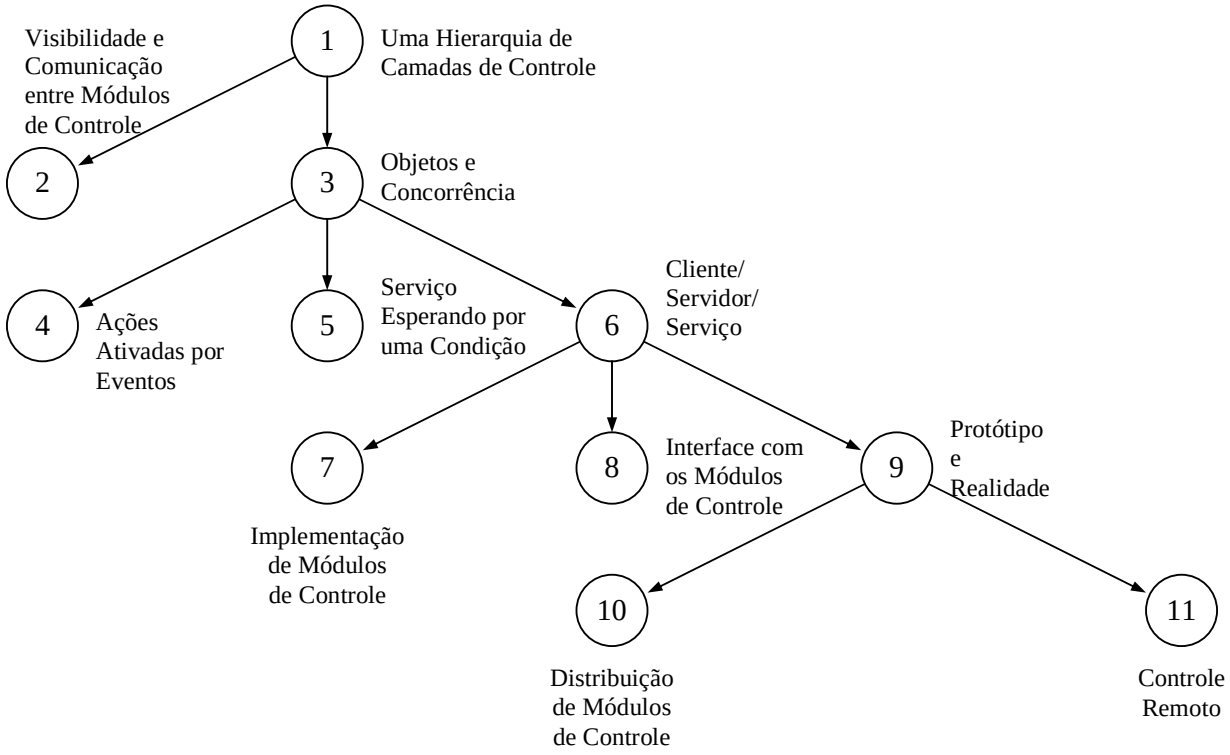


Figura 3.11 – Linguagem de padrões G++ (Aarsten *et al.*, 2000)

A **Tabela 3.2** mostra o problema tratado em cada um dos padrões da linguagem G++ e também os padrões de projeto (Gamma *et al.*, 1995) que eqüivalem àqueles propostos ou que são utilizados para implementá-los. Todos os padrões, com exceção do padrão 4, Ações Ativadas por Eventos, e do padrão 9, Protótipo e Realidade, são descritos por um diagrama de classes (Aarsten *et al.*, 2000).

A partir da linguagem de padrões G++, o grupo de Turim desenvolveu um *framework* para a operação de sistemas flexíveis de fabricação, baseado no modelo de referência da Agência Nacional de Padrões dos Estados Unidos (*National Bureau of Standards* – USA-

NBS), mostrado na **Figura 3.12** (McLean *et al.*[10] *apud* Brugali *et al.*, 2000). Uma fábrica é organizada segundo uma hierarquia de controle, com diferentes responsabilidades alocadas a cada nível. Uma planta representa um sistema completo de produção ou uma fábrica e executa o planejamento a longo prazo. Os sistemas são os módulos do chão de fábrica que programam os lotes de produção e que coordenam as células de produção. Um lote de produção é definido como um grupo de peças idênticas a ser fabricada em conjunto. Uma célula é uma unidade de produção que realiza expedições, seqüenciamentos e roteamento de peças pelas máquinas em tempo real e gerencia a preparação dos equipamentos para processar as peças. Uma estação de trabalho é uma máquina ou um equipamento de transporte de uma célula, capaz de executar diversas operações. Um equipamento representa o menor componente com controle independente.

Tabela 3.2 – Linguagem de padrões G++ (Aarsten *et al.*, 2000)

Padrão		Problema	Padrões de Projeto
1	Uma Hierarquia de Camadas de Controle	Como o sistema pode ser organizado e dividido para melhor tratar a sua complexidade? Qual é a melhor arquitetura para tais sistemas?	
2	Visibilidade e Comunicação entre Módulos de Controle	Como podem ser estabelecidas as relações de visibilidade correta entre os módulos de controle?	observador
3	Objetos e Concorrência	Como deve ser modelada a concorrência de diferentes níveis de granularidade?	
4	Ações Ativadas por Eventos	Como devem ser manipulados os eventos dentro de um dos serviços?	comando
5	Serviço Esperando por uma Condição	Como devem ser manipulados os eventos entre os serviços?	
6	Cliente/Servidor/Serviço	Como criar e gerenciar *threads* independentes, como possibilitar a concorrência e como encapsular os recursos e os serviços que os utilizam?	
7	Implementação de Módulos de Controle	Como devem ser implementados os módulos reais de controle? Como devem ser sincronizados os acessos a recursos compartilhados?	comando
8	Interface com os Módulos de Controle	Como pode ser reduzida a dependência entre o servidor e seus clientes?	adaptador, representante
9	Protótipo e Realidade	Como pode ser assegurada uma evolução sem problemas da simulação até a implementação final?	representante, adaptador, ponte, fabricação abstrata
10	Distribuição de Módulos de Controle	Como pode ser abordada a evolução para transformar um protótipo de módulo local de controle em um módulo remoto de controle de um dispositivo físico?	representante, ponte
11	Controle Remoto	Como habilitar o software de controle a acessar os estados do sistema para executar o monitoramento e o controle das atividades (também remotas)?	

[10] MCLEAN, C.; MITCHEL, M.; BARKMEYER, E. (1983). A Computer Architecture for small-batch Manufacturing. *IEEE Spectrum*. v. 20, n. 5, p. 59-64.

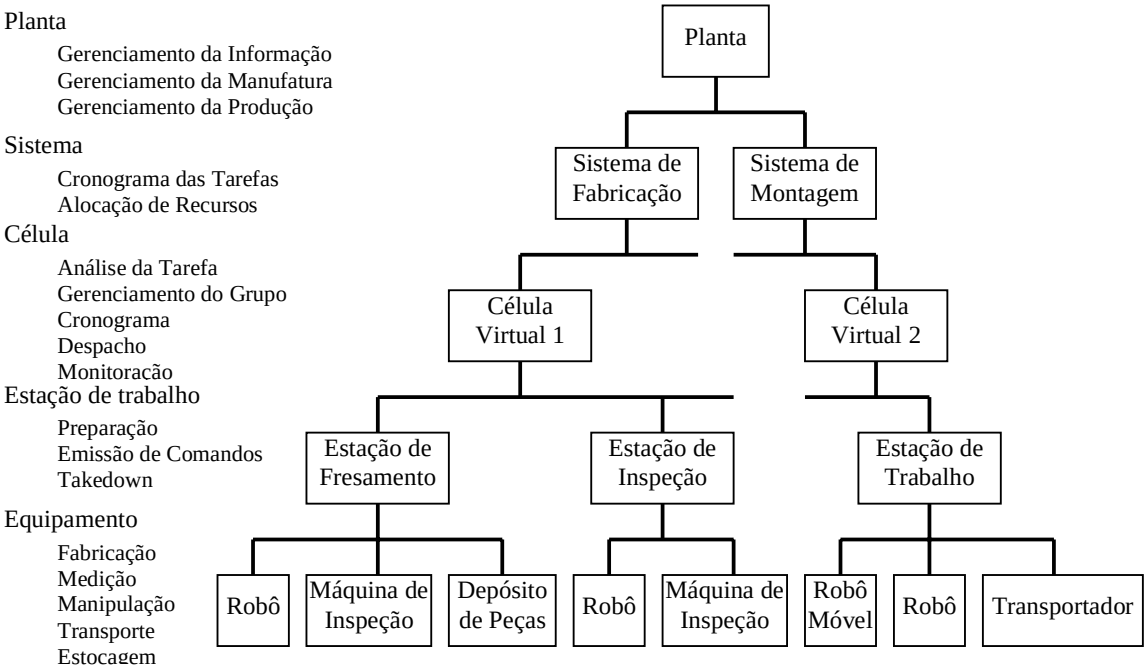


Figura 3.12 – Modelo de referência da NBS para sistemas flexíveis de fabricação (McLean *et al. apud* Brugali *et al*., 2000)

O *framework* oferece mecanismos, classes abstratas e blocos construtores adaptáveis para a maioria das soluções propostas pela linguagem de padrões G++. Os componentes do *framework* oferecem diferentes níveis de abstração de projeto, necessários no domínio da aplicação. Eles são classificados como componentes elementares, básicos e dependentes do domínio. Os componentes elementares são dados pelos mecanismos para realizar a comunicação e concorrência. Os componentes básicos são classes intermediárias, independentes do domínio.

Os níveis hierárquicos são definidos de acordo com o padrão 1, Uma Hierarquia de Camadas de Controle. O padrão 2, Comunicação e Visibilidade entre os Módulos de Controle, garante que módulos de controle de maior nível hierárquico tenham visibilidade e requisitem serviços de módulos de menor nível hierárquico, enquanto que a comunicação no sentido contrário é realizada por difusão (do inglês *broadcasting*) de eventos. Módulos do mesmo nível não devem ter visibilidade um do outro. Todos os componentes são derivados de uma classe base abstrata denominada *Object*, que oferece a possibilidade de difundir e de escutar eventos. Adicionalmente, os objetos que escutam os eventos podem adaptar o seu comportamento ao especializar a classe abstrata denominada *Action*, usando o mecanismo oferecido pelo padrão 4, Ações Ativadas por Eventos.

Cada instância da classe *Thread* representa um processo seqüencial independente. A existência da concorrência é garantida pela execução simultânea de mais de um desses processos. O *framework* prevê três tipos de concorrência:

- granulação fina: é modelada pelo padrão 4, Ações Ativadas por Eventos, ativando-se o objeto *Action* correspondente ao evento ocorrido;
- granulação média: é modelada pelo padrão 5, Serviço Esperando por uma Condição, por meio da classe abstrata *Condition*, que é especializada na implementação dos objetos de bloqueio *EventHandler*, para obter-se sincronização, e *CondCltn* (coleção de condições), para modelar os recursos compartilhados;
- granulação grossa: é modelada pelo padrão 6, Cliente/Servidor/Serviço, por meio das classes *Service* e *Server* utilizadas para implementar os módulos de controle segundo o padrão 7, Implementação de Módulos de Controle. As classes *Service* e *Server* relacionam-se tanto entre si como com outros objetos, segundo a interface disponibilizada pelo padrão 8, Interface com os Módulos de Controle.

O gerenciamento da concorrência em seus diversos níveis e tipos é garantido pelo padrão 3, Objetos e Concorrência. As classes *Object*, *Thread*, *Condiction*, *EventHandler*, *CondCltn*, *Action*, *Server* e *Service* são os componentes de baixo nível, elementares e básicos, do *framework*, as quais possibilitam ao desenvolvedor a implementação de software para sistemas concorrentes. Os componentes dependentes do domínio são especializações dos componentes de baixo nível, criados sempre tendo em mente o domínio específico para o qual pretende-se gerar aplicações.

A criação de um *framework* para o domínio Sistemas Flexíveis de Fabricação dá-se através da adição de módulos concretos de alto nível, tal qual aqueles representados pelas classes Sistema (*Shop*), Célula, Máquina e Transporte, do diagrama de classes da **Figura 3.13**. A reutilização desses módulos em uma aplicação específica requer a adaptação de seus pontos variáveis, os objetos *Action* e *Service*.

O desenvolvimento de um Sistema Flexível de Fabricação passa por uma fase de simulações do funcionamento de seu software de controle, na qual os equipamentos reais são substituídos por módulos de controle. O padrão 9, Protótipo e Realidade, assegura uma evolução sem problemas do módulo de controle para o sistema físico, devido à existência de duas implementações, uma, o Protótipo, que simula o comportamento do objeto e outra, a Realidade, que controla o objeto físico. Podem ocorrer duas situações distintas. Quando o

servidor é um módulo de software remoto implementado em outro computador, utiliza-se o padrão 10, Distribuição dos Módulos de Controle. Quando o servidor é um dispositivo físico utiliza-se o padrão 11, Controle Remoto.

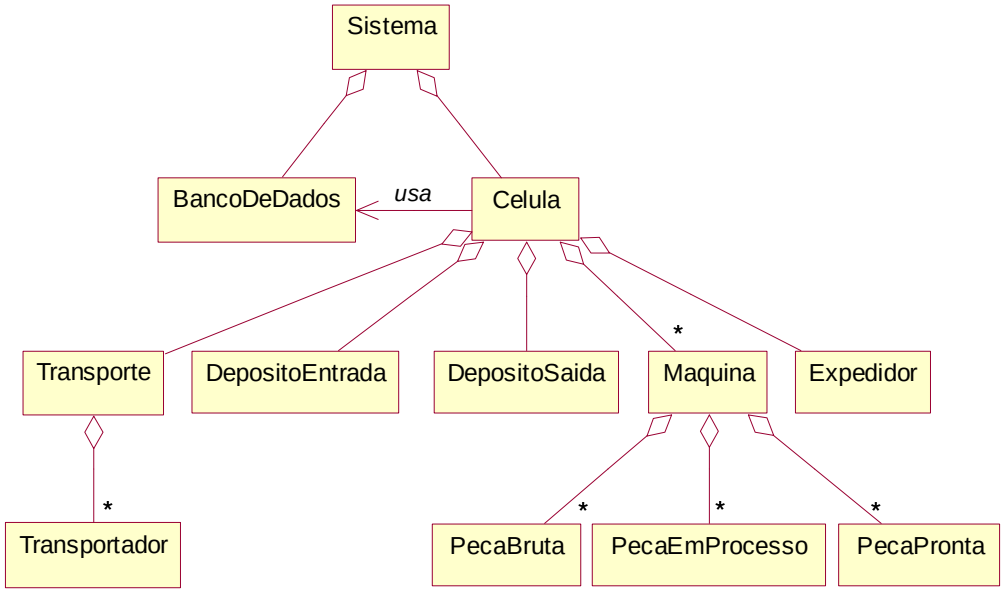


Figura 3.13 – Diagrama de classes do *framework* (Brugali *et al*., 2000)

A linguagem de padrões G++ foi desenvolvida simultaneamente com o *framework* para Sistemas Flexíveis de Fabricação utilizado para testá-la. A linguagem foi utilizada também no desenvolvimento de *frameworks* nas áreas de robótica (Aarsten *et al*., 1996a), controle de processos (VASME[11] *apud* Brugali *et al*., 2000), telecomunicações (Bosco *et al*.[12] *apud* Brugali *et al*., 2000) e integração computador-telefone (Billi *et al*.[13] *apud* Brugali *et al*., 2000).

3.6.3 Análise Comparativa dos *Frameworks*

Os dois *frameworks* apresentados nas seções anteriores possuem vários aspectos comuns, tanto que ambos os autores citam-se mutuamente na maioria de suas publicações. Ambos os *frameworks* foram desenvolvidos para o controle de um sistema flexível de fabricação e, conseqüentemente, se enquadram no mesmo subdomínio da manufatura.

---

[11] VASME (1996). VASME. Value-added services for maritime environments. *European Union Framework Program IV*. Project n. WA-95-SC.010, document n. 960628/D1.12.1996.
[12] BOSCO, P.G.; LOGIUDICE, D.; MARTINI, G.; MOISO, C. (1996). ACE: An environment for specifying, developing, and generating TINA services. *Proceedings IFIP/IEEE IM*.
[13] BILLI, R. *et al*. (1996). Field trial evaluations of two different information inquiry systems. In: *IEEE Workshop on Interactive Voice Tecnology for Telecommunications Applications, 3*, Basking Ridge-NJ, September 30-October 1.

Entretanto, a linguagem de padrões apresentada pelo grupo de Turim já foi também utilizada para o desenvolvimento de *frameworks* em outras áreas, como descrito no fim da seção anterior, enquanto que o *framework* apresentado por Schmid é específico para esse subdomínio.

Os modelos básicos de classes dos dois *frameworks* são bem semelhantes. No topo existe uma classe que representa o sistema de fabricação, que é composta pela agregação das classes que representam os equipamentos. Schmid implementa uma classe denominada controle do processo, que é responsável pelo controle efetivo das atividades que são realizadas, enquanto que o grupo de Turim delega essa responsabilidade à classe no topo da hierarquia. Schmid cria uma aplicação pela implementação de um programa principal composto por comandos que, inicialmente, definem cada um dos equipamentos e como eles se relacionam, enquanto que o grupo de Turim adiciona módulos concretos de alto nível e adapta os seus pontos variáveis em subclasses específicas. As classes que representam a hierarquia física do sistema de fabricação trocam mensagem entre si da mesma forma em ambos os *frameworks*. Uma classe do nível hierárquico superior tem visibilidade e requisita serviços de uma do nível imediatamente abaixo, enquanto que a comunicação no sentido contrário é realizada por difusão. Ambos os *frameworks* implementam o controle de sistemas reativos, ou seja, as classes do nível hierárquico superior aguardam uma resposta da classe do nível inferior como confirmação da realização do serviço requisitado a elas.

Brugali (1998) ampliou a linguagem de padrão G++ em sua tese de doutorado para incorporar a possibilidade do controle simultâneo de várias unidades fabris, inclusive fisicamente distantes uma da outra, que interagem na produção de um conjunto mecânico qualquer. O *framework* de Schmid é inicialmente previsto para o controle de um único sistema produtivo.

3.7 Considerações Finais

Neste capítulo apresentou-se, de forma sucinta, o domínio da manufatura e, de forma mais detalhada, os subdomínios fluxo de material e sistemas flexíveis de fabricação. Os conceitos básicos foram definidos, bem como os equipamentos e dispositivos utilizados e os sistemas computacionais existentes para simular o fluxo de material foram discutidos. A revisão da literatura específica mostra que existe um esforço internacional para o desenvolvimento de *frameworks* com flexibilidade e facilidade de reuso no domínio da manufatura. Constata-se também que, nesse domínio, existem várias propostas relacionadas

ao subdomínio dos sistemas flexíveis de fabricação mas no subdomínio do planejamento do processo, no qual são gerados os dados de entrada para o subdomínio dos sistemas flexíveis de fabricação, não foram encontradas propostas de soluções baseadas em *frameworks*.

No subdomínio do planejamento do processo estão disponíveis alguns pacotes comerciais para a geração de informações para equipamentos individuais (seção 3.5). Entretanto, soluções completas, utilizáveis em sistemas de fabricação que agrupam máquinas-ferramenta com vários outros equipamentos para a formação de sistemas integrados de manufatura, não foram encontradas. A determinação das operações em nível macroscópico que devem ser realizadas na peça, ou seja, o fluxo que ela deve realizar dentro do sistema produtivo desde a escolha do material em bruto até a obtenção de peças prontas no depósito correspondente ainda é normalmente realizada por um especialista do domínio, praticamente sem assistência computacional. Trata-se de um subdomínio pouco explorado que merece atenção especial em função da sua importância no domínio da manufatura.

Levando em conta os aspectos mencionados acima, propõe-se o desenvolvimento de um *framework* para a determinação macroscópica das operações que devem ser realizadas sobre uma peça com base na simulação do fluxo da peça no sistema de fabricação onde ela será produzida. O Capítulo 4 apresenta o método de simulação proposto para determinar as operações a partir da estratégia de produção da peça. No Capítulo 5 mostra-se a generalização deste método para que ele seja implementado com um *framework* e, conseqüentemente, possa ser aplicado a um espectro maior de sistemas de fabricação.

4 Um Método para Simulação do Fluxo de Peças em Sistemas Produtivos

4.1 Considerações Iniciais

Neste capítulo apresenta-se a proposta de um método exaustivo de simulação, que segundo os autores da área é classificado como: dinâmico, pois o estado do sistema modelado se altera; discreto, pois a alteração do estado do sistema ocorre em instantes bem definidos após manter-se inalterado ao longo de um intervalo de tempo; e determinístico, pois o comportamento do sistema não se altera com o tempo e os resultados da simulação podem ser reproduzidos (Banks *et al*., 1996; Freitas, 2001; Law & Kelton, 1991).

Para as partes importantes do método são mostrados o processo de modelagem do sistema de fabricação e o modelo conceitual resultante, que é a base para a implementação computacional, em um nível de detalhamento suficiente para a compreensão de seu funcionamento. Para o emprego do método proposto é suficiente o conhecimento do sistema de fabricação, da geometria da peça, dos processos de fabricação necessários a sua produção e dos equipamentos aptos a realizar os respectivos processos. A aplicação do método resulta na folha de processos para a produção de uma peça em um sistema flexível de fabricação.

O método proposto é válido para qualquer geometria da peça e para os processos de fabricação correspondentes para a sua produção. Entretanto, o método é apresentado apenas para a produção de peças com simetria rotacional. Conseqüentemente, apresentam-se somente os processos de fabricação utilizados para as peças escolhidas. Esta opção reduz a quantidade de passos que compõem a folha de processo. Os recursos envolvidos restringem-se também às máquinas e seus dispositivos utilizados para o processamento de peças rotacionais, o que simplifica a descrição, a modelagem e a implementação computacional. Apesar do método ser válido para peças prismáticas ou mais complexas, a sua extensão para esse tipo de peça não faz parte do escopo deste trabalho.

Na seção 4.2 são relacionadas as informações básicas para emprego do método de simulação proposto. A seção 4.3 mostra o procedimento de simulação proposto, que é fundamentado na simulação dos equipamentos utilizados durante o fluxo da peça e na simulação das transferências da peça entre os equipamentos. Como o método proposto pode obter mais de uma solução válida, apresenta-se na seção 4.4 uma forma estruturada de escolha

do resultado mais adequado. Na seção 4.5 apresentam-se as considerações finais deste capítulo.

4.2 Informações Básicas

Os elementos utilizados durante o processo de produção de uma peça em sistemas flexíveis de fabricação são divididos em dois grupos principais no escopo deste trabalho. Os elementos que pertencem ao primeiro grupo são denominados equipamentos. Eles representam as máquinas instaladas em caráter quase que definitivo e que praticamente não se alteram por um período de tempo relativamente longo, como robôs, máquinas operatrizes, sistemas de transporte e depósitos, entre outros. O segundo grupo é composto pelos elementos denominados dispositivos. Ao contrário dos equipamentos, os dispositivos não são instalados de uma forma fixa, mas podem ser utilizados em mais de um equipamento ou fluir pelo sistema. Os dispositivos entram em contato direto com a peça e representam a interface mecânica entre o equipamento e a peça. Além desses dois grupos, as informações relativas aos materiais em bruto e às peças em bruto devem estar disponíveis para o emprego do método. Nas seções seguintes são relacionadas as informações necessárias para descrever os equipamentos, os dispositivos e os demais itens normalmente utilizados na fabricação flexível e que interessam ao método aqui proposto.

4.2.1 Descrição dos Equipamentos

Cada um dos equipamentos pertencente a um sistema de fabricação está associado a uma função específica do fluxo de material para a qual ele foi desenvolvido, apesar de em alguns casos poder ser aplicado para exercer outras funções. A palavra “fluxo” está geralmente associada à idéia de movimento, mas somente as funções de manipulação e transporte dão importância à localização espacial, ou à sua alteração, da peça objeto do fluxo de material. O fluxo de material só ocorre quando existe a interação entre os vários equipamentos instalados em um sistema flexível. A interação não é aleatória e, além disso, em cada fase do fluxo de material ocorre de forma distinta. Os sistemas de fabricação mais complexos são subdividos em grupos responsáveis pela realização destas fases. Estes grupos recebem o nome de célula. Em sistemas de fabricação é comum a existência de:

- células de fabricação: responsáveis pela alteração da geometria da peça; algumas vezes divididas de acordo com o tipo de peça que podem fabricar (células rotacionais e células prismáticas, por exemplo);

- células de transporte: responsáveis pela movimentação de material entre as células;
- células de montagem: responsáveis pela montagem de diferentes peças, produzidas ou não no sistema de fabricação, para a formação de um conjunto;
- células de comissionamento: responsáveis pela formação de grupos de peças em bruto em uma única unidade de transporte;
- células de inspeção: responsáveis pela medição de alguma grandeza relativa à peça; etc.

Para realizar suas atividades, as células possuem não só o equipamento principal, mas também equipamentos auxiliares para estocagem temporária de peças e dispositivos e para transportes e manipulações internas (na célula), entre outros.

A **Figura 4.1** apresenta o diagrama de classes correspondente utilizando-se a notação UML. A figura mostra que um sistema flexível de fabricação (Sff) é composto de uma ou mais células e cada uma das células por um ou mais equipamentos.

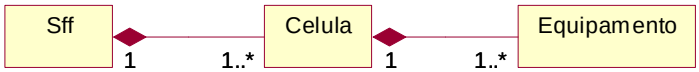


Figura 4.1 – Relacionamento entre o Sff, a célula e o equipamento

Os equipamentos são divididos de acordo com a função do fluxo de material que eles exercem preferencialmente. Como resultado tem-se cinco tipos de equipamentos. Os equipamentos que executam cada uma das cinco funções do fluxo de material são novamente subdivididos de acordo com a sua classificação técnica dentro do grupo a que pertencem. Existem diversos equipamentos, tecnicamente diferentes entre si, que executam cada uma das funções do fluxo de material, mas no escopo deste trabalho são de interesse apenas os seguintes:

- equipamentos de fabricação: serra e torno;
- equipamentos de manipulação: robô; e
- equipamentos de estocagem: depósito de material em bruto, depósito de peça em bruto e depósito de palete.

Os equipamentos acima relacionados são descritos pelos dados técnicos relativos ao emprego típico de cada um. São relacionados, por exemplo, os dados referentes ao volume de trabalho, ao peso máximo da peça e, no caso dos equipamentos de estocagem, o que está

depositado em cada um deles. A simulação da utilização dos equipamentos durante o fluxo de material, segundo essas informações, resulta em uma resposta booleana, ou seja, o equipamento pode ou não ser utilizado para executar uma determinada função. Uma simulação mais detalhada da utilização dos equipamentos de fabricação e manipulação é realizada por meio da simulação do emprego do dispositivo utilizado pelo equipamento.

Na **Figura 4.2** mostra-se o diagrama de classes que representa esta estrutura para a descrição dos equipamentos. A modelagem mais simples resultaria em relações de especialização entre os níveis de descrição dos equipamentos, mas como um equipamento pode exercer mais de uma função do fluxo de material, ou dentro de uma dada função executar atividades técnicas típicas de mais de um equipamento, a utilização de associações entre as classes torna-se necessária. Com este modelo é possível descrever, por exemplo, o robô utilizado na célula de manufatura mostrada por Schmid (1995, 1996a, 1996b, 2000a). O robô é utilizado inicialmente para o transporte de paletes desde o depósito central (*store*) até o armazém local (*buffer*) e em uma fase seguinte ele é utilizado para alimentar o torno com as peças existentes no palete. Esses dois procedimentos caracterizam, respectivamente, as funções de transporte e de manipulação. Em um modelo em que as relações entre os níveis de descrição dos equipamentos fossem de especialização, não seria possível a modelagem da execução dessas duas tarefas pelo mesmo equipamento.

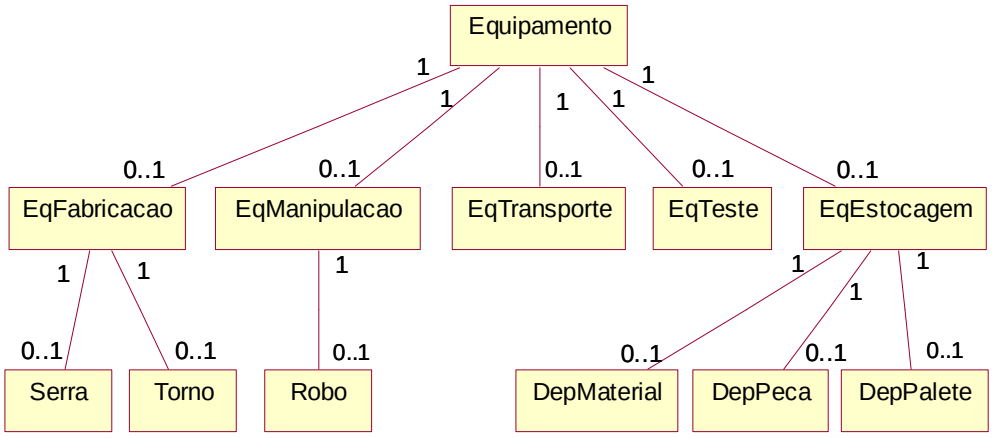


Figura 4.2 – Detalhamento dos equipamentos

4.2.2 Descrição da Interação entre os Equipamentos

Conforme mencionado na seção anterior, os equipamentos pertencentes a sistemas flexíveis de fabricação não são utilizados isoladamente, mas interagem entre si. Neste trabalho a interação entre equipamentos é denominada conexão. Uma conexão é caracterizada

pela existência de um equipamento que realiza uma operação de manipulação ou de transporte. Essa operação é realizada a partir de um equipamento de origem até um equipamento de destino. Além dos três equipamentos participantes da conexão, é necessário descrever o objeto que está sendo movimentado. O objeto manipulado em uma conexão de manipulação normalmente é a peça, tanto em bruto como semipronta ou pronta. O objeto transportado em uma conexão de transporte normalmente é um palete de transporte. O diagrama de classes correspondente é mostrado na **Figura 4.3**.

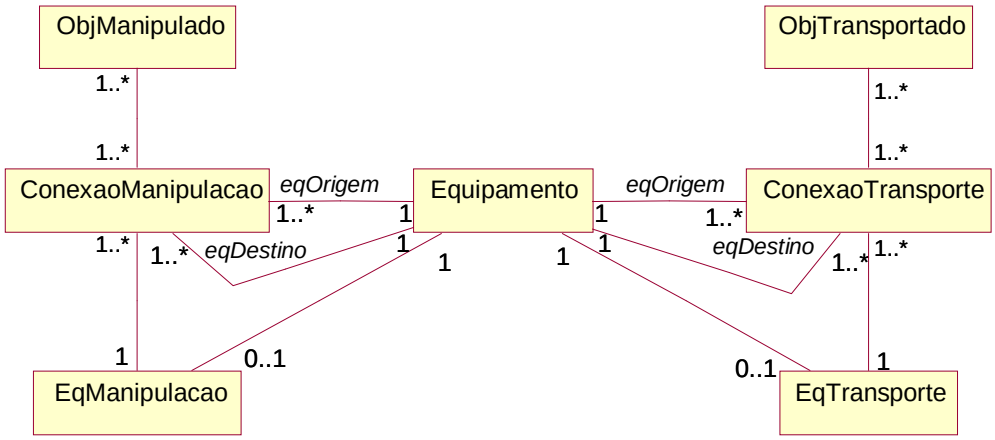


Figura 4.3 – Descrição da conexão de manipulação e da conexão de transporte

4.2.3 Descrição dos Dispositivos

A descrição dos dispositivos consiste inicialmente de seus dados geométricos. Todas as dimensões e distâncias de interesse devem ser conhecidas antes de simular a possibilidade de utilização do dispositivo. A funcionalidade do dispositivo não é descrita de forma convencional, mas sim, associa-se a cada tipo de dispositivo a respectiva funcionalidade. No escopo deste trabalho é necessária a descrição de placas de fixação, garras de robôs e paletes de transporte.

4.2.3.1 Placa de Fixação

As placas de fixação representam a interface mecânica entre o torno e a peça. A **Figura 4.4** mostra uma placa de três castanhas. Ela é composta por um corpo cilíndrico, denominado corpo da placa, e por um jogo de três castanhas (na figura estão representadas apenas duas), que podem movimentar-se, em relação ao corpo da placa, na direção radial dentro de um limite de curso preestabelecido. As castanhas possuem um número finito de degraus, cada um caracterizado por duas superfícies principais, as superfícies de fixação e de apoio. Além das

castanhas, pode existir uma tampa, que não é mostrada na figura, para obstruir o furo do corpo da placa.

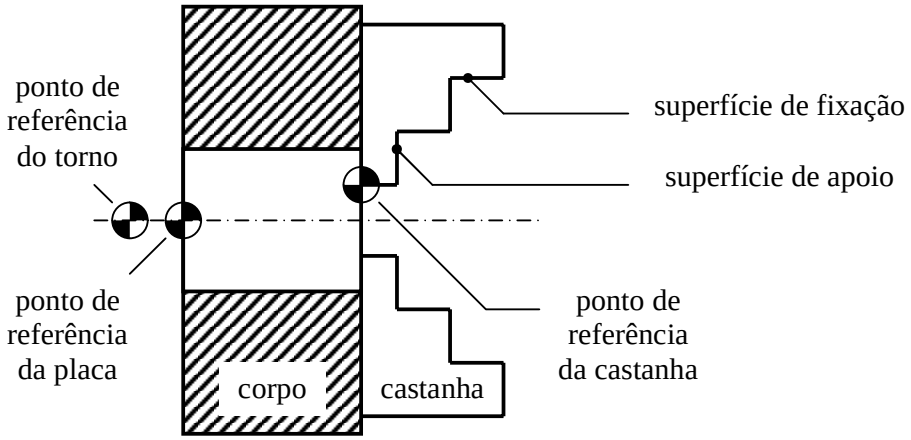


Figura 4.4 – Representação esquemática de uma placa de três castanhas

Os dados geométricos do corpo da placa são o seu diâmetro externo (D_{cp}) e interno (d_{cp}), sua largura (l_{cp}) e o curso total das castanhas (c_{cp}). Os dados geométricos da castanha consistem da sua altura (h_{ct}) e sua largura (l_{ct}), bem como da sua posição relativa ao curso do corpo da placa (p_{ct}). As informações geométricas da castanha só são completadas quando cada um de seus degraus está definido. As informações geométricas dos degraus são as suas coordenadas relativas à castanha, uma na direção da largura (p_l) e outra na direção da altura (p_h), e o raio de curvatura (R) da sua superfície de fixação. A tampa é geometricamente descrita pelo seu diâmetro externo (D_{tp}) e pela sua largura (l_{tp}). A **Figura 4.5** mostra esquematicamente os dados relacionados acima, com exceção daqueles relativos à tampa.

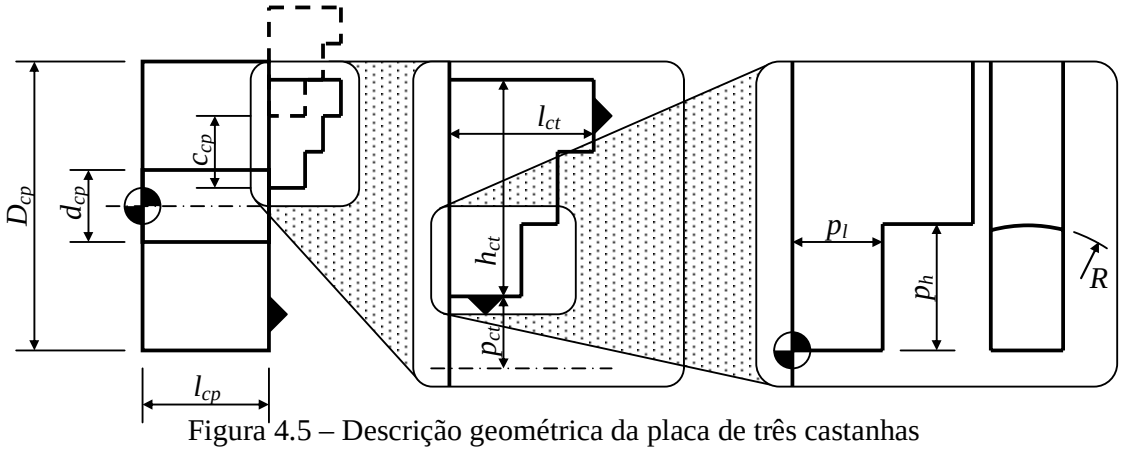


Figura 4.5 – Descrição geométrica da placa de três castanhas

Do ponto de vista funcional, as superfícies de fixação das três castanhas agem no sentido radial sobre uma das superfícies cilíndricas da peça e são responsáveis pela sua

centragem. Com isso, obtém-se o posicionamento da peça em dois eixos de translação e a orientação nos três eixos de rotação. Além das superfícies de fixação dos degraus, a superfície interna da castanha, assinalada pelo símbolo (▼), pode, ou não, ser utilizada como superfície de fixação. O posicionamento da peça no terceiro eixo de translação, aquele na direção do eixo de rotação da placa de fixação, é garantido pela superfície de apoio. Além das superfícies de apoio dos degraus, a superfície frontal da própria castanha e a superfície frontal do corpo da placa, denotadas pelo símbolo (▶), podem, ou não, também ser utilizadas como superfície de apoio. A montagem de uma tampa na placa pressupõe que sua superfície frontal possa ser utilizada como superfície de apoio.

A **Figura 4.6** mostra o diagrama de classes que representa a estrutura para descrição das placas. Os sistemas flexíveis de fabricação são compostos por pelo menos uma placa de fixação, a qual pertence a um único sistema flexível de fabricação. As placas possuem um número qualquer de tampas e pelo menos um jogo de castanhas. Tanto as tampas como as castanhas só podem ser montadas em uma única placa. As castanhas podem ter um número qualquer de degraus, que são detalhes construtivos de uma única castanha. Na figura, essas relações são representadas por agregações com as respectivas multiplicidades.

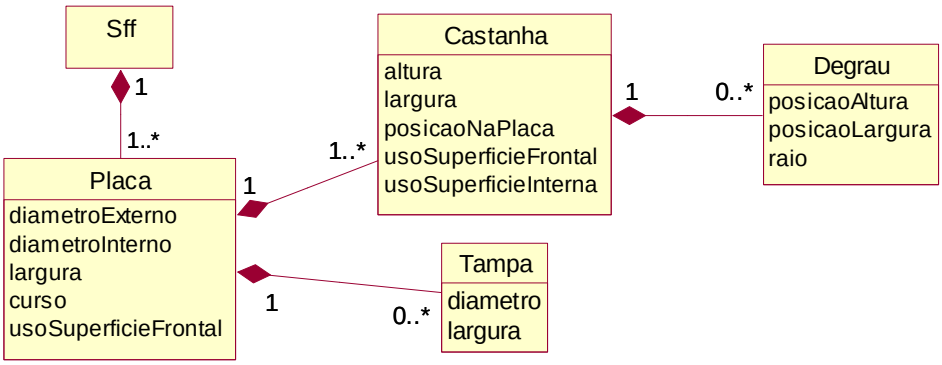


Figura 4.6 – Modelo da placa de três castanhas

4.2.3.2 Garra

As garras podem assumir diferentes formas geométricas em função principalmente da forma da peça a ser manipulada. Conseqüentemente, a sua descrição geométrica também varia. As garras mais comuns, que inclusive encontram-se disponíveis comercialmente (Schunk, 1999), são compostas de um corpo e de pinças, geralmente em número de duas ou três, que se movimentam na direção radial, dentro de um limite de curso preestabelecido.

Os dados geométricos de uma garra dependem do tipo de garra utilizado. A **Figura 4.7** mostra o esquema de uma garra de duas pinças, utilizadas para a manipulação de peças com simetria rotacional, tipicamente obtidas em processo de torneamento. As pinças possuem duas superfícies planas, defasadas de um dado ângulo, formando um assento prismático para a fixação da peça.

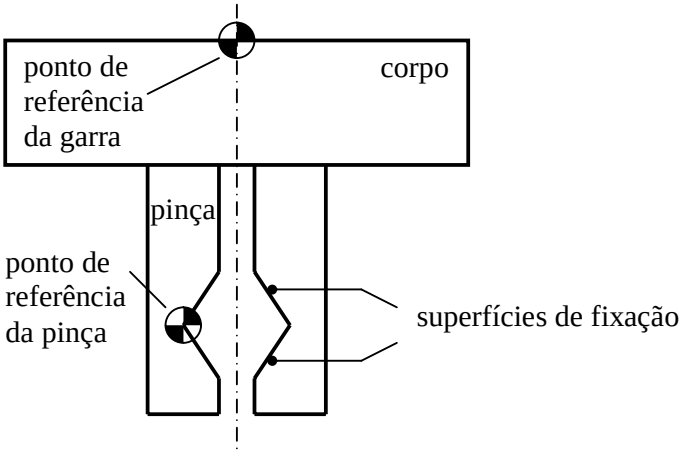


Figura 4.7 – Representação esquemática de uma garra de duas pinças

Em geral, as garras de duas pinças são construídas com corpos em forma prismática e mais raramente em forma cilíndrica. Os dados geométricos do corpo da garra em forma prismática são a sua altura (h_{cp}), o seu comprimento (l_{cp}), a sua largura (w_{cp}) e o curso total das pinças (s_{cp}). Os dados geométricos da pinça com assento prismático consistem das coordenadas do seu ponto de referência em relação ao corpo da garra, uma na direção da altura (h_{pc}) e outra na direção do comprimento, quando a pinça encontra-se na posição fechada (l_{pc}), da largura da pinça (w_{pc}), do ângulo entre as superfícies do prisma de fixação (α_{pc}) e da sua profundidade (d_{pc}). A **Figura 4.8** mostra esquematicamente os dados relacionados acima.

Do ponto de vista funcional, as quatro superfícies de fixação agem simultaneamente no sentido radial sobre uma das superfícies cilíndricas da peça e são responsáveis pela sua centragem. Conseqüentemente, obtém-se o posicionamento em dois eixos de translação e a orientação nos três eixos de rotação. A posição relativa entre a peça e a garra, no terceiro eixo de translação, está associada a um erro de posicionamento da garra em relação à peça no instante em que ocorre a fixação. Após a fixação, o posicionamento é garantido pela força de atrito entre as superfícies de fixação da pinça e a superfície fixada da peça.

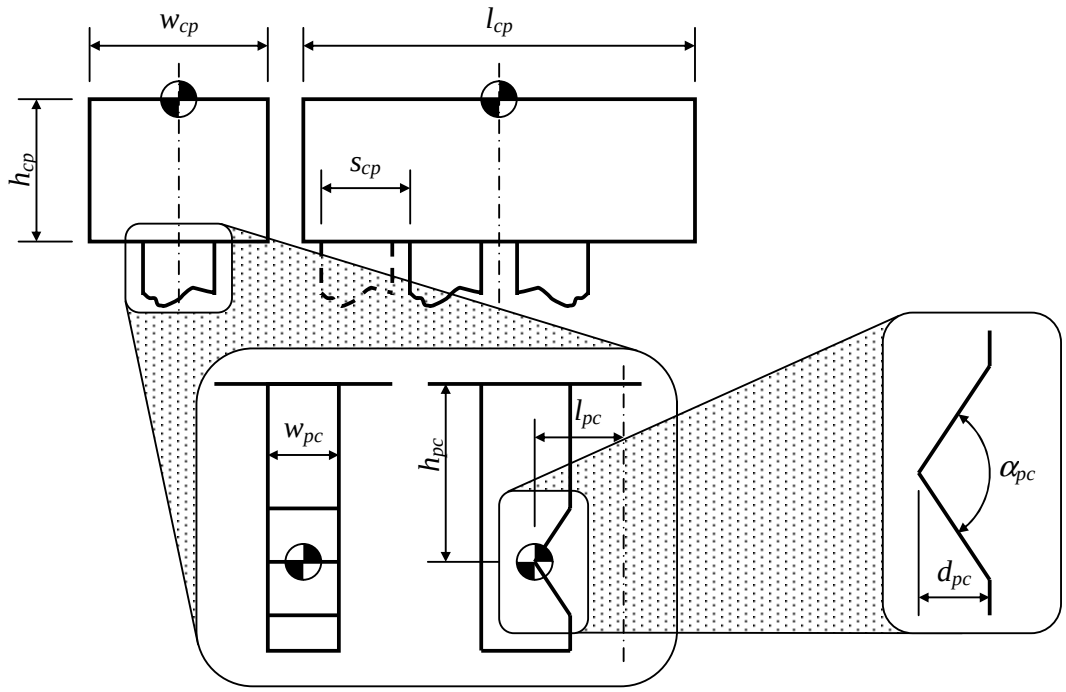


Figura 4.8 – Descrição geométrica da garra de duas pinças

A **Figura 4.9** mostra o diagrama de classes que representa a estrutura para descrição das garras de duas pinças. Os sistemas flexíveis de fabricação são compostos por pelo menos uma garra, a qual pertence a um único sistema flexível de fabricação. As garras possuem pelo menos um par de pinças, que só podem ser montadas em uma única garra. Na figura, essa relação é representada pela agregação com a respectiva multiplicidade.

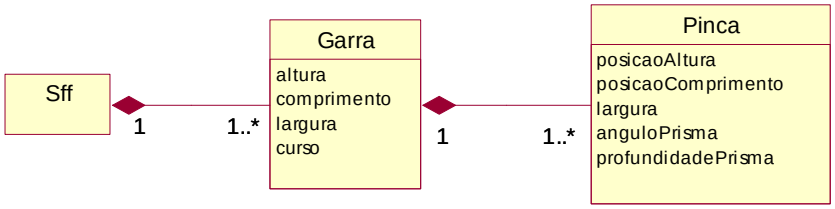


Figura 4.9 – Modelo da garra de duas pinças

4.2.3.3 Palete de Transporte

Os paletes podem transportar, ou mesmo armazenar, não só peças, mas também ferramentas e outros dispositivos. No escopo deste trabalho são analisados somente os paletes de transporte de peças. Eles são constituídos de duas partes. A parte inferior, denominada base do palete, é construída com uma geometria específica que determina se um equipamento está apto a recebê-la ou não. A base do palete pode ser utilizada para a montagem de diferentes formas de partes superiores. A parte superior, aqui denominada simplesmente de palete, é sempre montável sobre uma única base. Sobre o palete (parte superior) são montados os

elementos para apoio das peças. Os elementos de apoio são denominados longarinas. A geometria das longarinas varia de acordo com a geometria e a orientação da peça a ser transportada. Elas são normalmente construídas a partir de chapas dobradas, no comprimento necessário para a sua fixação no palete. Como, em geral, o comprimento é significativamente maior que aquele necessário para apoiar a peça, as longarinas podem transportar simultaneamente mais de uma peça. A geometria na região para acomodar a peça repete-se ciclicamente ao longo da longarina. Para o transporte de peças com simetria rotacional são utilizados principalmente dois tipos de longarina para sustentação da peça na horizontal e na vertical. A **Figura 4.10** mostra esquematicamente o apoio de peças sobre longarinas, tanto na horizontal como na vertical.

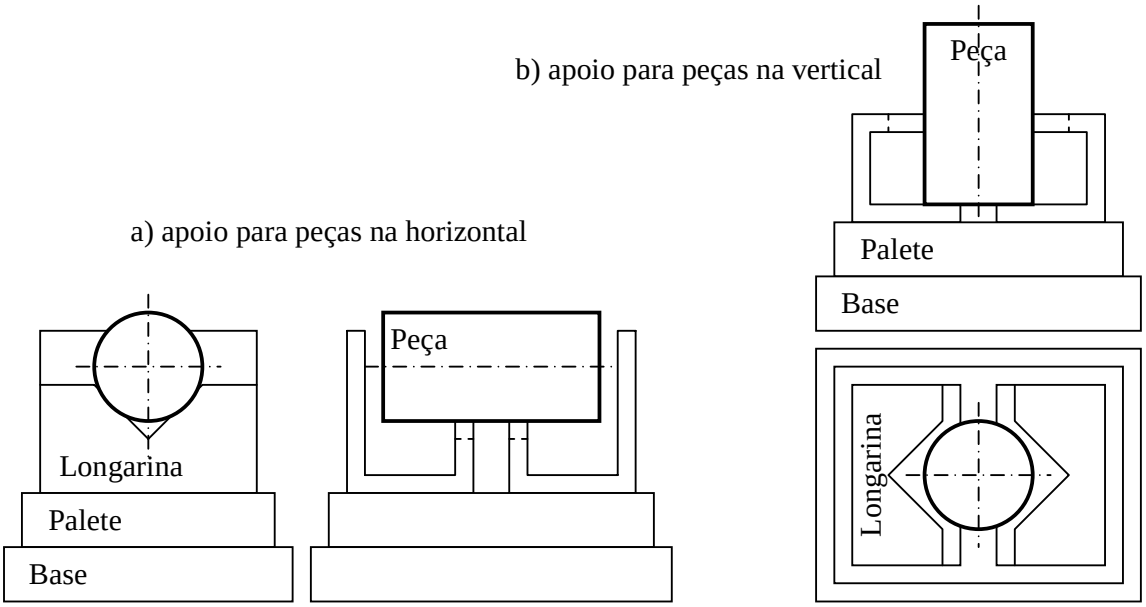


Figura 4.10 – Representação esquemática do apoio sobre longarinas

Os dados geométricos da base do palete são o seu comprimento, sua largura e sua altura. O palete possui dimensões externas compatíveis com a sua base. Os dados geométricos do palete são completados com a definição do passo de montagem das longarinas. Um passo com valor positivo significa que as longarinas podem ser montadas, tendo o seu ponto de referência localizado sempre em intervalos discretos, múltiplos do passo. Um passo nulo indica que a posição das longarinas pode variar continuamente.

Os dados geométricos de uma longarina com apoio prismático para peça na horizontal são a sua altura na superfície lisa para apoio axial da peça (h_{sm}), na superfície que contém o prisma (h_{ph}), a sua largura (w_{ls}), a posição do seu ponto de referência (r_{ls}), o passo ou distância entre dois de seus prismas (s_{ls}), a espessura da chapa (t_{sh}), a profundidade do prisma (d_{ph}) e o

ângulo entre as superfícies do prisma (α_{ph}). Os dados geométricos de uma longarina com apoio prismático para peça na vertical são a sua altura (h_{ls}), a sua largura na parte superior (w_{tp}) e na parte inferior (w_{bt}), a posição do seu ponto de referência (r_{ls}), o passo ou distância entre dois de seus prismas (s_{ls}), a espessura da chapa (t_{sh}), a profundidade do prisma (d_{pv}) e o ângulo entre as superfícies do prisma (α_{pv}). As **Figuras 4.11** e **4.12** mostram respectivamente os dados referentes às longarinas para apoio de peças na horizontal e na vertical.

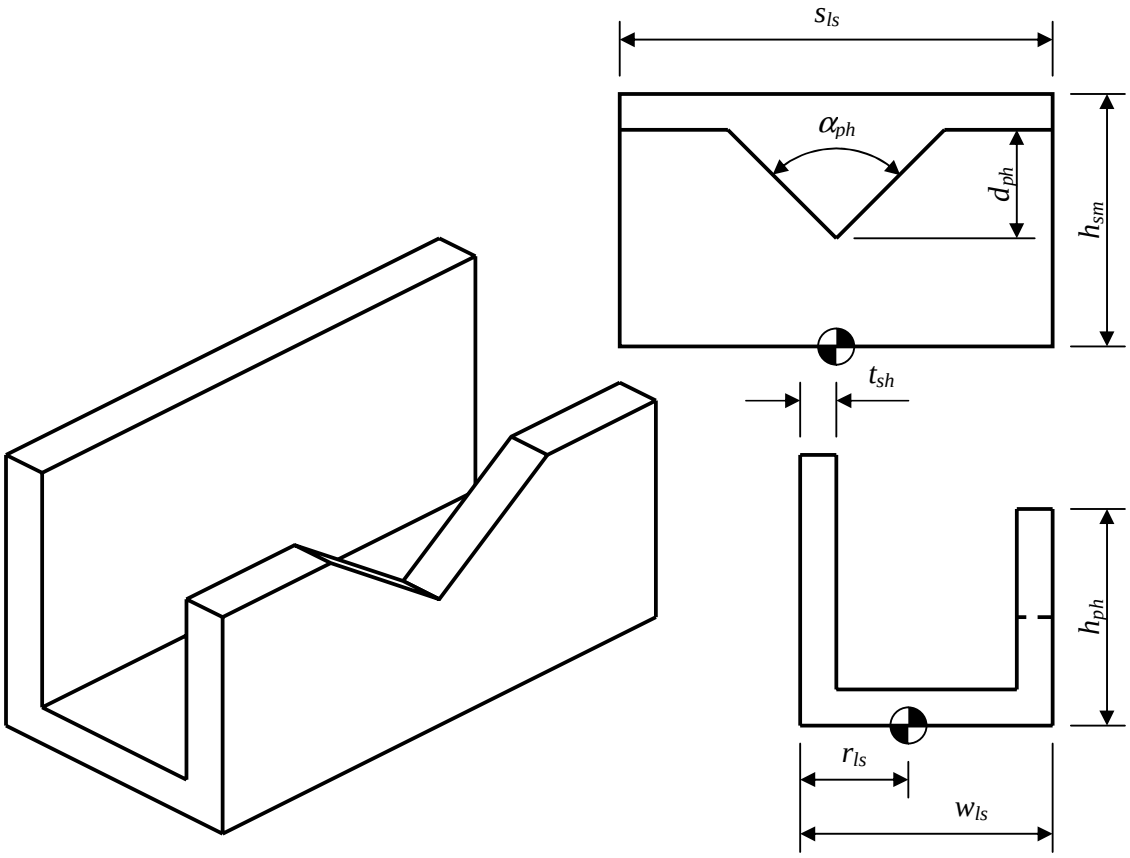


Figura 4.11 – Descrição geométrica da longarina para apoio horizontal

Os objetos transportados em paletes não são fixados nele. Os objetos ficam apoiados sobre as longarinas pela ação da força gravitacional, que garante o posicionamento na direção vertical e a orientação da peça. Nos paletes para apoio da peça na horizontal, o posicionamento em um dos eixos de translação horizontal é normalmente garantido por prismas de sustentação. O posicionamento no outro eixo não é totalmente definido, mas apenas delimitados por valores extremos, entre os quais a peça pode se deslocar. Nos paletes para apoio da peça na vertical, o posicionamento em ambas as direções horizontais é limitado por valores extremos.

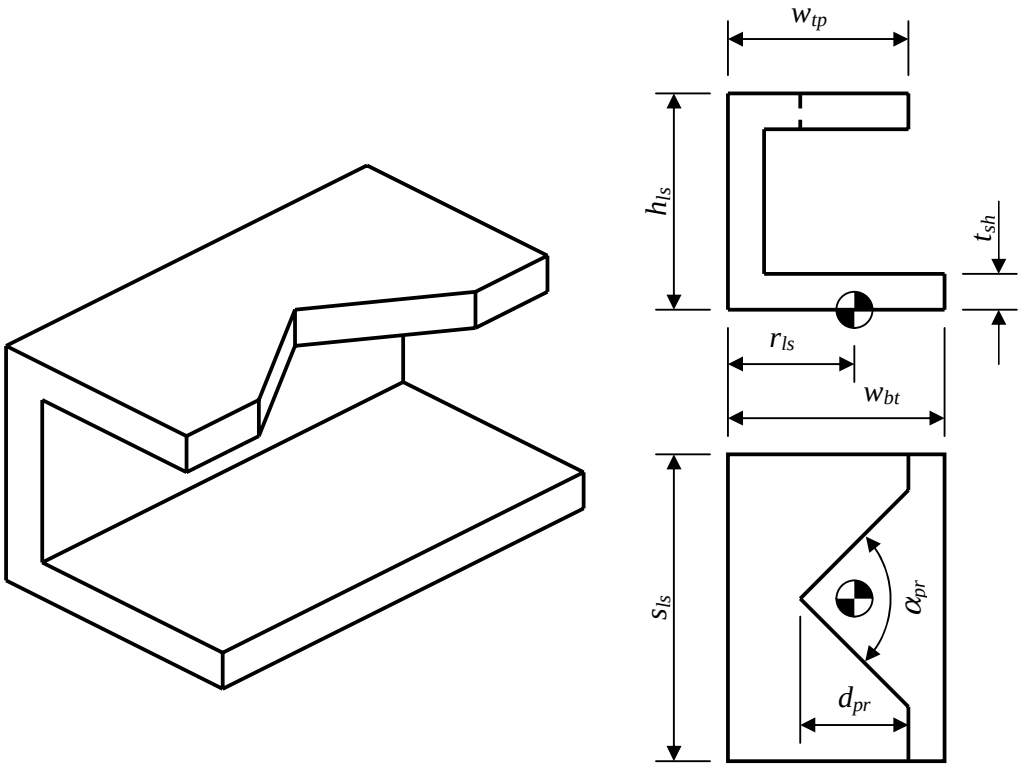


Figura 4.12 – Descrição geométrica da longarina para apoio vertical

A **Figura 4.13** mostra o diagrama de classes para descrição dos paletes de transporte. Os sistemas flexíveis de fabricação são compostos por pelo menos uma base de palete e pelo menos uma longarina, as quais pertencem a um único sistema flexível de fabricação. As bases do palete possuem pelo menos um palete (parte superior), que pode ser montado em uma única base. Os paletes são utilizados para a montagem de pelo menos um tipo de longarina, sendo que cada uma pode ser montada em pelo menos um palete. O diagrama mostra a classe abstrata longarina, que define uma interface comum para todos os seus tipos. O relacionamento entre a base e o palete é do tipo 1::N e é representado por uma agregação. O relacionamento entre o palete e a longarina é do tipo M::N.

4.2.4 Descrição dos Materiais para a Produção da Peça

O material para a produção da peça pode estar disponível em duas formas distintas. A primeira possibilidade é a existência de peças em bruto. Para a produção de peças com simetria rotacional as peças em bruto são cilíndricas cheias ou têm forma de tubos cilíndricos. A geometria das peças em bruto é, portanto, definida pelo seu diâmetro externo, pelo seu diâmetro interno (no caso de tubos) e pelo seu comprimento.

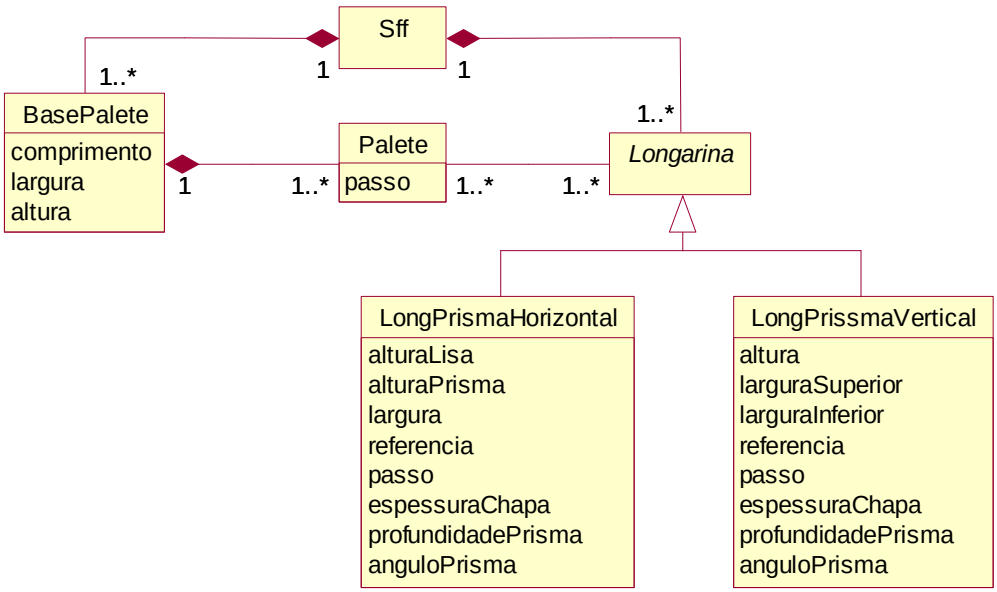


Figura 4.13 – Modelo do palete de longarinas

A segunda possibilidade é que a peça seja fabricada a partir de materiais em bruto. O material em bruto consiste de uma barra com grande comprimento, que deve ser cortada no comprimento apropriado para obter-se uma peça em bruto. A geometria da seção transversal dos materiais em bruto é a mesma que das peças em bruto (cilíndrica ou tubo cilíndrico). Tanto na fabricação convencional como na flexível, a utilização de materiais em bruto conduz, normalmente, a uma economia significativa, pois eles são serrados para a obtenção das peças em bruto em comprimentos mais adequados à fabricação da peça do que aqueles das peças em bruto preexistentes em um depósito. A economia dá-se tanto no custo de material, pois o volume da peça em bruto serrada é menor do que aquele de uma peça em bruto previamente existente, como também no custo de usinagem, pois o volume a ser removido da peça em bruto para a obtenção da peça pronta é menor.

Tanto o material em bruto como a peça em bruto devem ser constituídos do mesmo material do qual a peça deve ser produzida. A descrição do material da peça consiste, no escopo deste trabalho, de algumas de suas propriedades físicas, utilizadas em cálculos que serão descritos posteriormente.

A **Figura 4.14** mostra o diagrama de classe para descrição dos materiais para a produção da peça. Nos sistemas flexíveis de fabricação existem peças em bruto e materiais em bruto, que estão disponíveis em um único sistema flexível de fabricação. As classes peça em bruto e material em bruto são abstratas e definem interfaces comuns para todos os seus tipos.

Tanto a peça em bruto como o material em bruto são constituídos de um único material. Em contrapartida, um material específico pode ser o elemento constituinte de uma quantidade qualquer de peças em bruto ou de materiais em bruto.

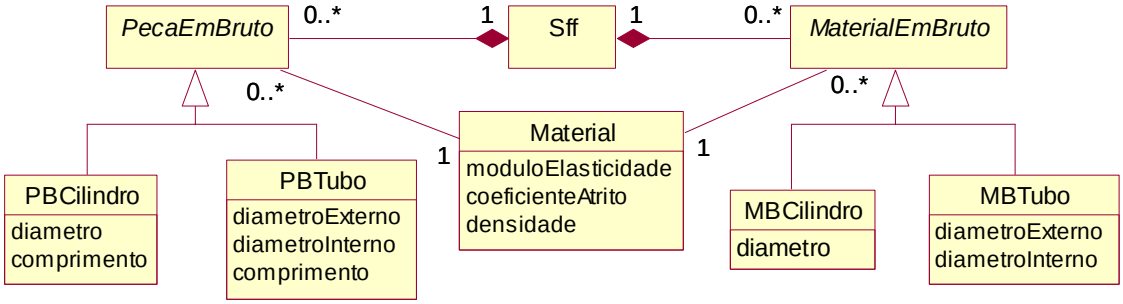
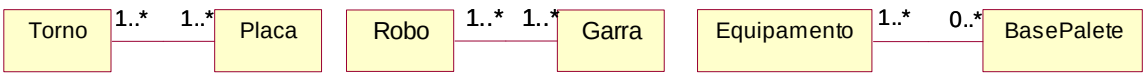


Figura 4.14 – Modelo da peça em bruto e do material em bruto

4.2.5 Relacionamento entre Equipamentos e Dispositivos

Os dispositivos mostrados na seção 4.2.3 são utilizados por equipamentos específicos que compõem o sistema de fabricação. Alguns destes relacionamentos sempre ocorrem, mas outros são características de cada sistema de fabricação. Para utilização do método aqui proposto deve-se informar todos os relacionamentos existentes. Logo, deve-se relacionar as placas de fixação com os tornos, as garras com os robôs e as bases dos paletes com os equipamentos aptos a recebê-las. Todos são relacionamentos M::N, ou seja, cada equipamento utiliza pelo menos um dispositivo e cada dispositivo pode ser utilizado em um ou mais equipamentos. O único relacionamento diferente é entre as bases do palete e os equipamentos, que em grande parte não estão aptos a receber paletes.

A **Figura 4.15** mostra os relacionamentos descritos acima. Um torno que não possua placas, uma placa que não pode ser montada em nenhum torno, um robô que não possua garras, uma garra que não é montável em nenhum robô e uma base de palete que não pode ser colocado em nenhum equipamento representam equipamentos ou dispositivos inúteis e não são utilizados para o fluxo da peça. Em contrapartida, um equipamento que não está apto a receber paletes, como ocorre, por exemplo, com tornos e robôs, pode participar de outra forma do fluxo da peça. Esta diferença está presente no diagrama de classes pela multiplicidade que pode assumir o valor nulo na associação entre equipamentos e bases de palete.

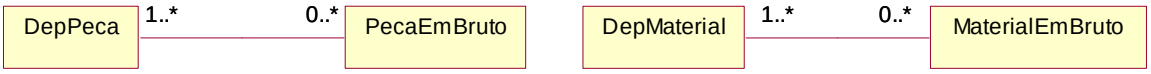


a) torno e placa b) robô e garra c) equipamento e base do palete

Figura 4.15 – Relacionamento entre equipamento e dispositivo

4.2.6 Relacionamento entre Equipamentos e Materiais

Conforme foi mencionado na seção 4.2.4, os materiais em bruto e as peças em bruto estão inicialmente disponíveis nos depósitos correspondentes. Tanto os depósitos como os materiais para produção das peças podem existir em qualquer quantidade. Conseqüentemente, torna-se necessária a descrição do relacionamento entre os materiais e os respectivos depósitos. Também são relacionamentos M::N, ou seja, tanto os materiais devem estar estocados em um ou mais depósitos, como cada depósito pode estocar um quantidade qualquer de materiais. A **Figura 4.16** mostra os diagramas de classes correspondentes.



a) peça em bruto e depósito de peça em bruto

b) material em bruto e depósito de material em bruto

Figura 4.16 – Relacionamento entre materiais e depósitos

4.3 Procedimento para Simulação do Fluxo da Peça

O procedimento de simulação proposto no escopo deste trabalho inicia pela determinação de rotas que o material em bruto pode percorrer desde o seu depósito até o depósito de peças prontas, passando pelos equipamentos que efetuam os processos de fabricação necessários à alteração geométrica da peça. No nível hierárquico inferior, as rotas são compostas por conexões de equipamentos, nas quais são realizados processos de manipulação ou transporte. A simulação do fluxo da peça pelas rotas encontradas é dada pela composição das simulações das conexões. Para simular uma conexão deve-se primeiramente efetuar a simulação dos três equipamentos que a compõem. Em seguida, simula-se a transferência da peça entre os equipamentos de origem e de movimentação e entre os de movimentação e de destino. A simulação da conexão é finalizada com a busca de pares de transferências compatíveis, caracterizadas pelo mesmo emprego do equipamento de

movimentação. Nas seções seguintes apresentam-se os procedimentos para determinação das rotas e para simulação das rotas, dos equipamentos, das transferências e das conexões.

4.3.1 Determinação das Rotas

A produção de peças por usinagem ocorre a partir de material em bruto ou de peça em bruto. No primeiro caso, o material em bruto deve ser serrado para obter-se uma peça em bruto com comprimento adequado à fabricação da peça. Em peças com simetria rotacional, a peça em bruto deve ser encaminhada a um torno para que seja efetuada a usinagem. Para peças puramente rotacionais, normalmente, a peça está pronta após o processo de torneamento. Após a usinagem a peça pronta deve ser levada ao depósito correspondente. A **Figura 4.17** mostra as duas possibilidades para realização do fluxo principal.

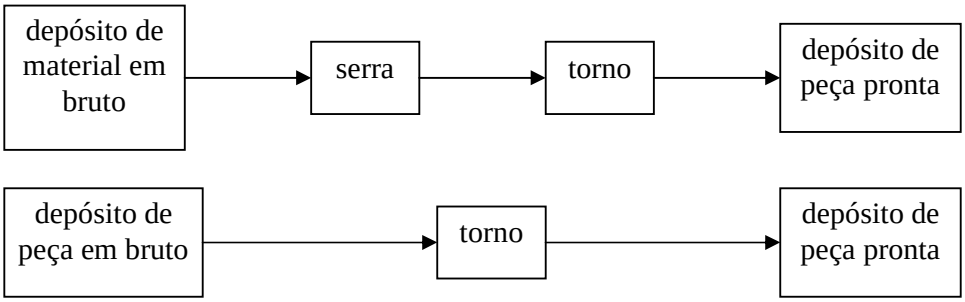


Figura 4.17 – Fluxos principais de peças com simetria rotacional

A determinação das rotas consiste, inicialmente, em encontrar caminhos parciais, pelos quais a peça possa fluir entre dois dos equipamentos acima relacionados, na seqüência em que eles aparecem. A esse caminho parcial é dado o nome de seção. O método proposto inicia pela procura de seções internas de cada uma das células, entre dois equipamentos consecutivos do fluxo principal da Figura 4.17, sempre com a restrição de não passar por outro equipamento que faça parte do fluxo principal. Em duas seções consecutivas, o equipamento de origem da seção sucessora deve coincidir com o equipamento de destino da seção antecessora para garantir a continuidade da rota.

Como os equipamentos do fluxo podem estar em células diferentes, buscam-se também as seções entre células, sem passar por uma terceira célula, e seções que representem caminhos parciais entre o equipamento do fluxo principal e um equipamento que seja considerado uma interface, entrada ou saída da respectiva célula. Como regra, as interfaces são depósitos temporários (*buffers*) de peças ou de paletes, onde os elementos depositados aguardam pela disponibilidade do sistema de transporte entre células para continuar o seu

fluxo. Assim, cada uma das quatro seções distintas pode ser subdividida em subseções conforme apresentado na **Figura 4.18**. A figura mostra ainda que podem existir oito subseções. Assim como nas seções, o equipamento de destino da subseção antecessora deve coincidir com o equipamento de origem da subseção sucessora para garantir a continuidade da seção.

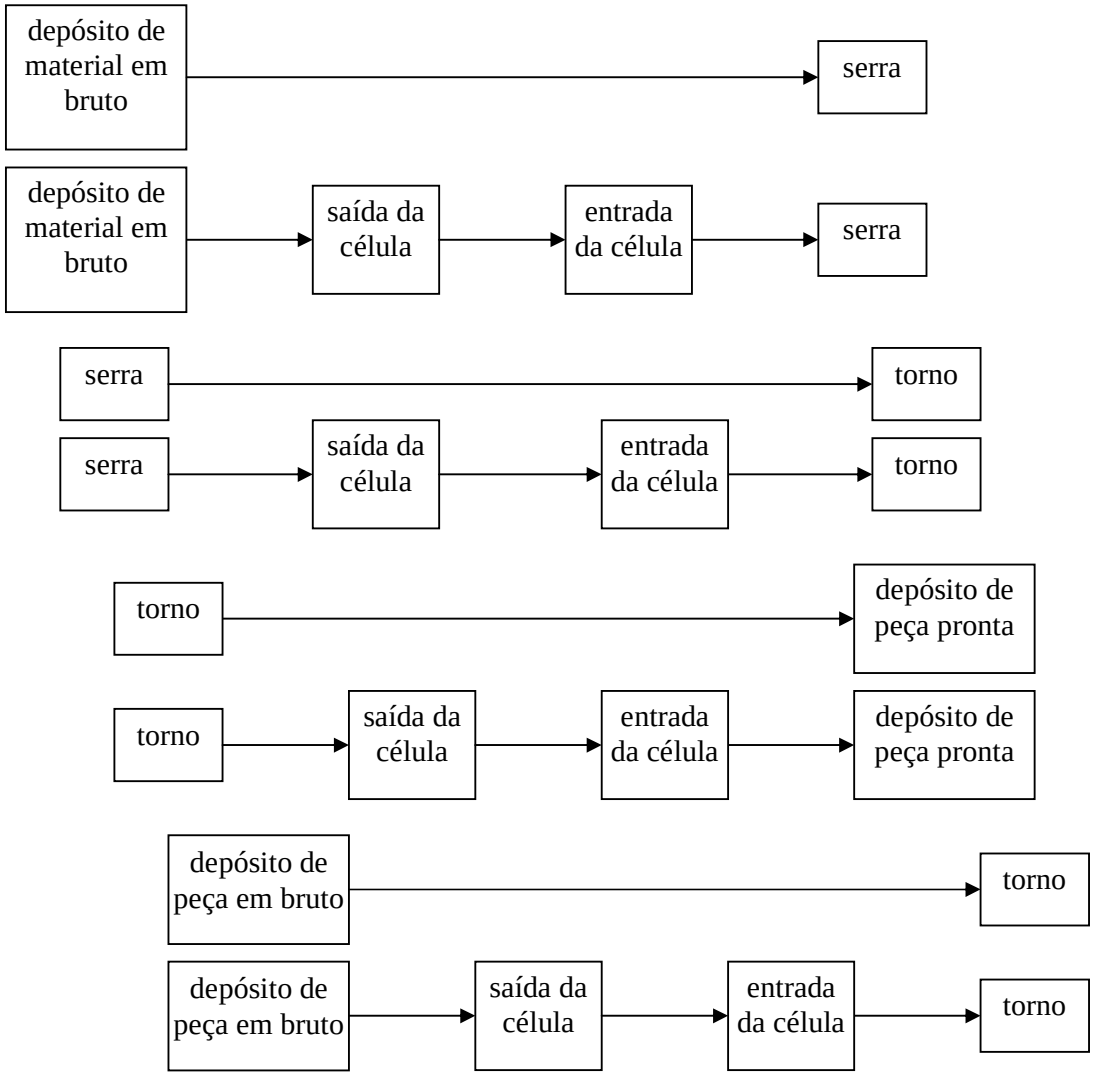


Figura 4.18 – Subdivisão das seções principais para equipamentos em células distintas

Cada uma das seções, ou uma de suas subdivisões, é definida por uma seqüência de conexões. Em duas conexões consecutivas, o equipamento de destino da conexão antecessora deve coincidir com o equipamento de origem da conexão sucessora, para garantir a continuidade das seções ou das subseções e, conseqüentemente, das rotas. O equipamento de origem da primeira conexão e o equipamento de destino da última conexão de cada uma das

seções ou das subseções são os equipamentos do fluxo principal (Figura 4.17) ou equipamentos de interface das células (Figura 4.18).

Do ponto de vista do modelo, podem existir diversas possibilidades de conexões. Entretanto, do ponto de vista funcional, somente parte das conexões possíveis são utilizadas em sistemas de fabricação. Neste trabalho não são modelados os testes ou as montagens, por exemplo, e, conseqüentemente, nenhum dos equipamentos tipicamente de testes ou montagens está presente no fluxo. Como mencionado anteriormente, está se descrevendo somente o fluxo de peças com simetria rotacional. Logo, os equipamentos de fabricação utilizados tipicamente para a alteração da geometria de peças não rotacionais não são considerados. Além destes, os depósitos de ferramentas ou dispositivos também não são considerados. A possibilidade de realização do fluxo de ferramentas e dispositivos não depende da peça que está sendo produzida. Se uma ferramenta ou um dispositivo pode ser utilizado em um equipamento, assume-se que existe a possibilidade de realizar o seu fluxo ou que ele está disponível no equipamento. Conseqüentemente, as conexões relacionadas aos equipamentos mencionados acima não são de interesse para o método aqui descrito. Portanto, somente um subconjunto reduzido das conexões possíveis está no escopo deste trabalho, as quais podem ser vistas na **Figura 4.19**. A figura apresenta em cada uma de suas partes, seções ou subseções, como na Figura 4.18, comparadas com as conexões que as compõem. As conexões são representadas por uma seta direcionada do equipamento de origem para o equipamento de destino. Abaixo da seta é mostrado o equipamento de movimentação (transporte ou manipulação) e acima dela o objeto movimentado na conexão.

A **Figura 4.20** mostra o diagrama de classes que descreve as rotas. Uma rota é uma agregação de seções, as quais podem ser simples ou compostas. As seções compostas são agregações de outras seções (subseções) e as seções simples são compostas por conexões. Na figura observa-se que as classes agregadoras são compostas por pelo menos uma classe agregada. Em caso contrário não faria sentido, pois a rota é representada por uma estrutura em árvore, em que as folhas necessariamente são conexões.

Na figura apresentam-se exemplos de seções simples, de seções compostas e de conexões. As subclasses cujos nomes terminam com reticências, indicam que outras subclasses, correspondentes aos tipos de seções, subseções e conexões, presentes nas figuras anteriores, fazem parte do diagrama de classes, mas não são mostradas em detalhe. A utilização do padrão de projeto “composição” (Gamma *et al.*, 1995) para descrição de seções compostas, permite que o cliente “rota” relacione-se com seções simples ou compostas indistintamente.

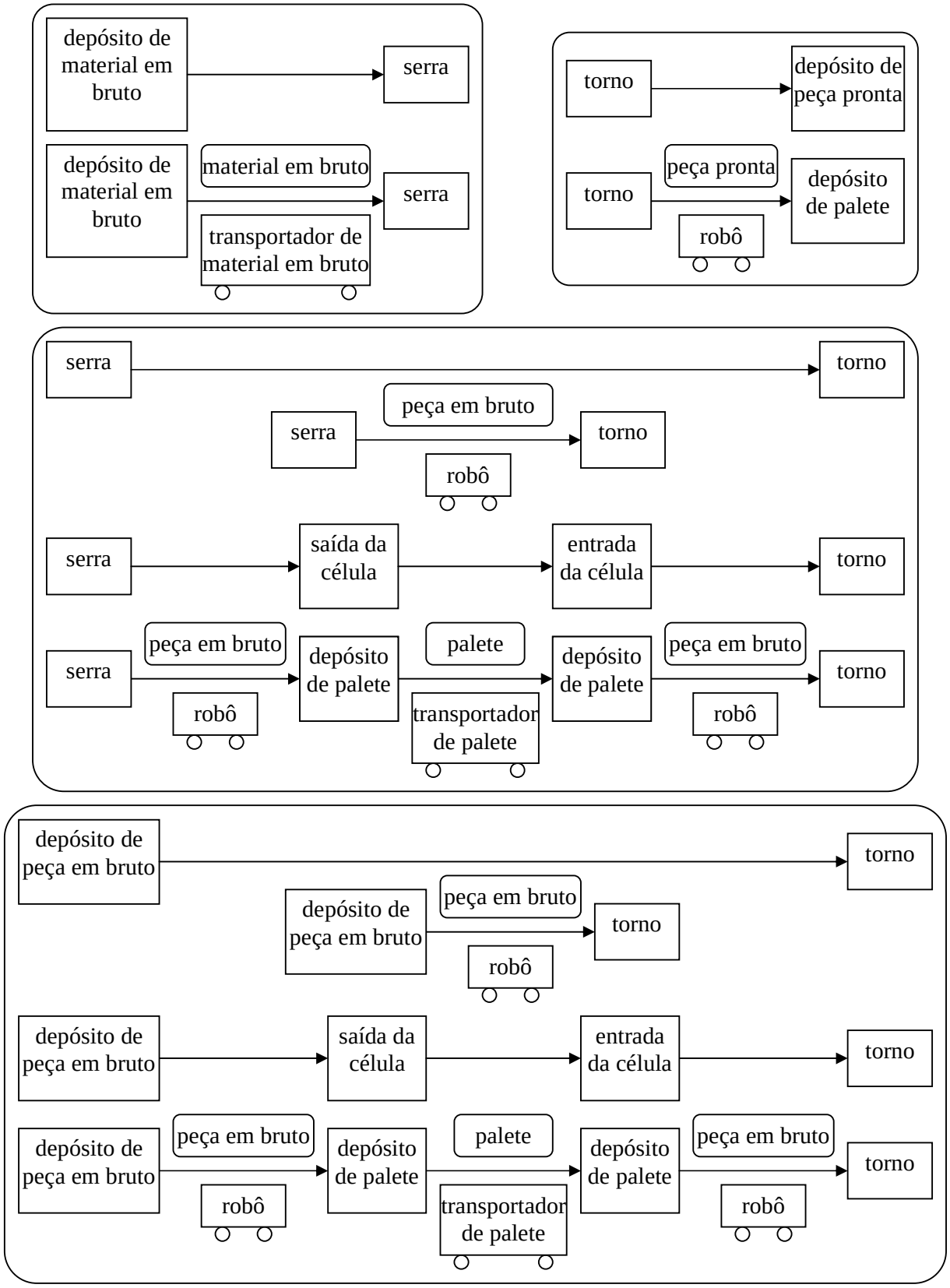


Figura 4.19 – Subdivisão das seções e subseções em conexões

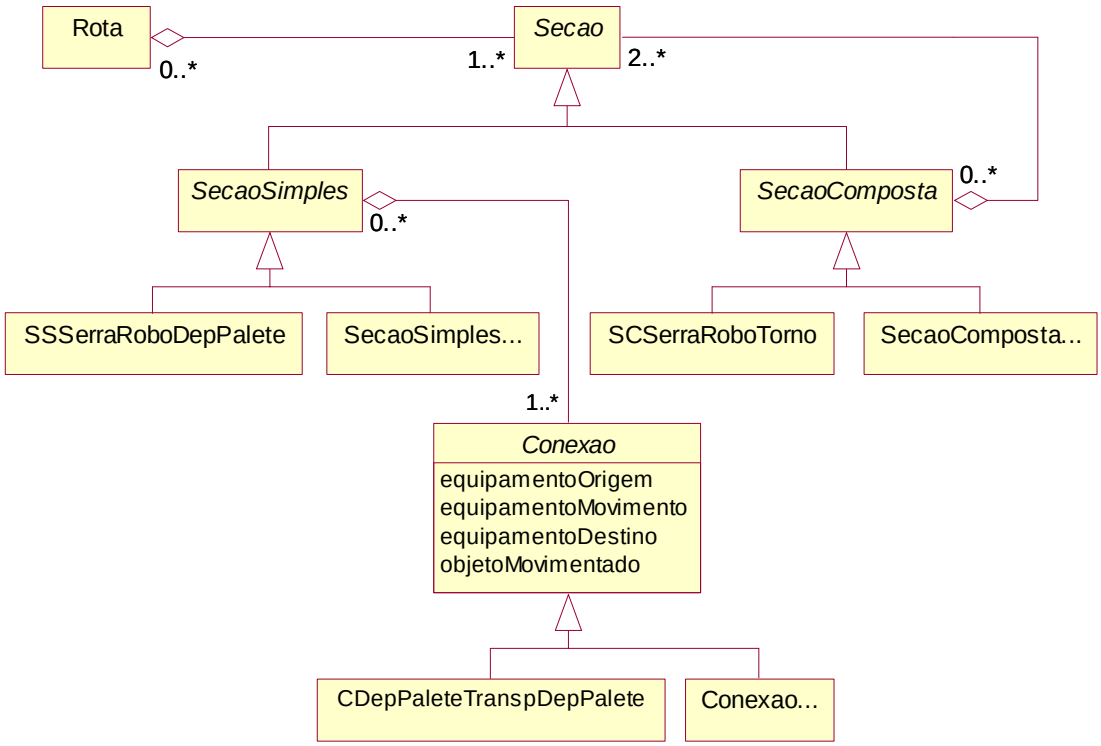


Figura 4.20 – Relacionamento entre rota, seção e conexão

4.3.2 Simulação das Rotas

A simulação do fluxo da peça durante o seu processo produtivo no sistema de fabricação é realizada pela simulação de cada uma das possíveis rotas encontradas. Para todas as rotas encontradas no sistema de fabricação, verifica-se a possibilidade de efetuar o fluxo pela análise das conexões que compõem cada uma das seções da rota. A **Figura 4.21** mostra o procedimento básico de simulação do fluxo da peça, utilizando a simulação de cada uma das possíveis rotas. A figura mostra a seção como uma classe isolada e não segundo o padrão composição, como representado anteriormente, para simplificar a descrição do procedimento.

Basicamente, em cada conexão simula-se a utilização dos equipamentos de origem, de destino e de movimentação (transporte ou manipulação). Em seguida, determina-se a possibilidade de realizar a transferência da peça entre os equipamentos de origem e de movimentação e entre os de movimentação e de destino, com base na simulação individual de cada equipamento. O procedimento de simulação de uma conexão é encerrado ao verificar se o processo completo pode ser realizado pelos equipamentos da conexão. O processo completo é caracterizado pelas duas transferências, uma no início e outra no fim. Nas duas transferências a utilização do equipamento de movimentação deve ser a mesma.

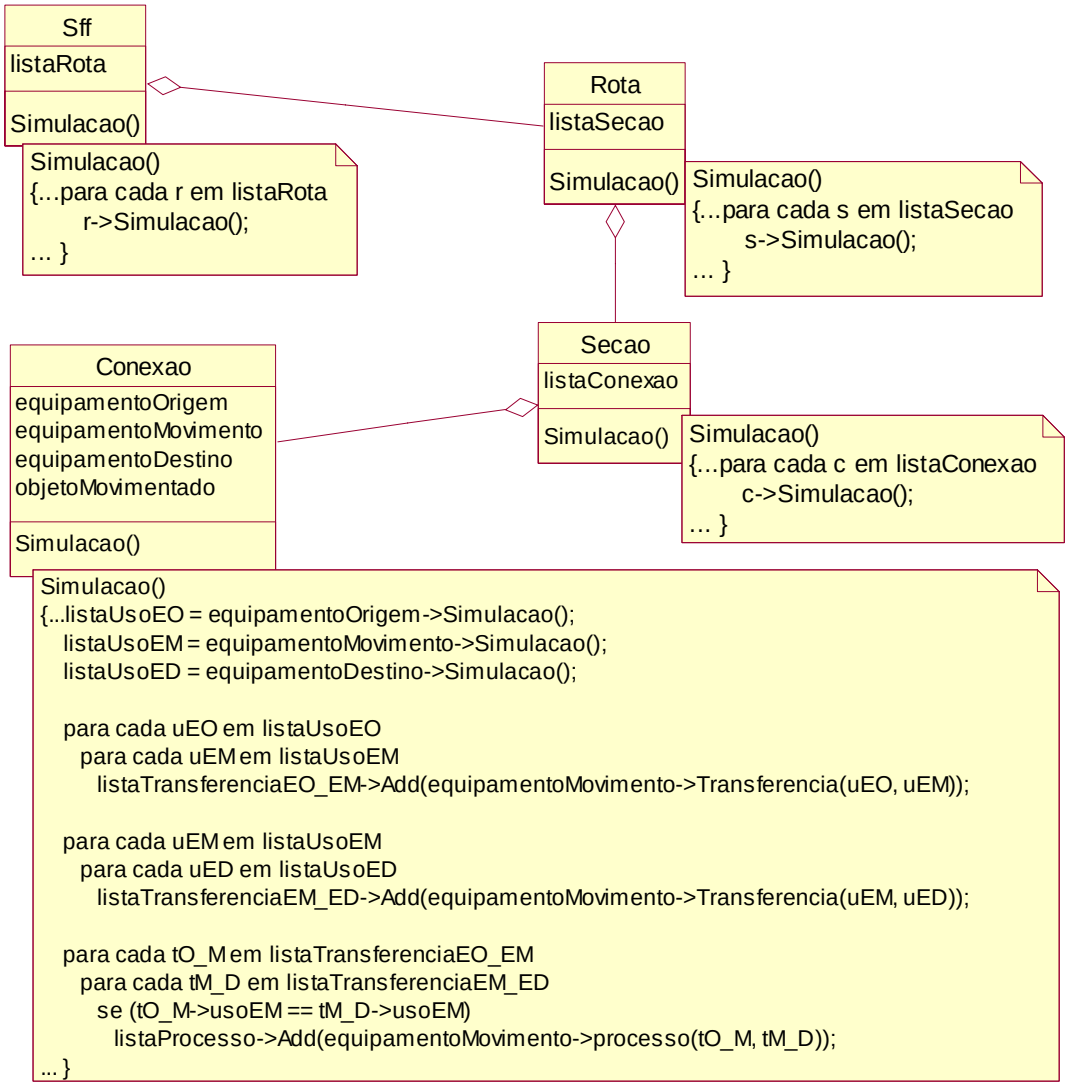


Figura 4.21 – Simulação genérica do fluxo de peças

A **Figura 4.22** ilustra de forma abstrata os resultados encontrados na simulação de uma conexão. Ao simular os equipamentos obtêm-se como resultado, por exemplo, quatro possibilidades de utilização do equipamento de origem, denotadas por O1, O2, O3 e O4, cinco possibilidades do equipamento de movimentação (M1 a M5) e três do equipamento de destino (D1 a D3). Na simulação da transferência poder-se-iam encontrar vinte transferências entre o equipamento de origem e o de movimentação e quinze entre o equipamento de movimentação e o de destino. A figura mostra que a maior parte das transferências não pode ser realizada (cruzes vermelhas). Somente oito transferências no início e três transferências no fim do processo podem ser realizadas (marcações verdes). Da mesma forma a quantidade de processos que são realizáveis é bem reduzida. Para cada utilização do equipamento de

movimentação, em princípio, poderiam ser realizados doze processos, dados pela combinação das quatro transferências no seu início com as três no seu fim. Entretanto, a figura mostra que somente quatro processos com a utilização M2 e três com a utilização M4 do equipamento de movimentação conduzem a processos que podem ser realizados entre os equipamentos presentes na conexão (setas azuis). Portanto, de um universo inicial de sessenta possibilidades de realização do processo, somente sete representam resultados válidos.

Usos do Equipamento de Origem				Usos do Equipamento de Movimentação	Usos do Equipamento de Destino		
O1	O2	O3	O4		D1	D2	D3
✗	✗	✗	✗	M1	✗	✗	✗
✗	✓	✓	✗	M2	✓	✓	✗
✓	✓	✓	✗	M3	✗	✗	✗
✓	✓	✓	✗	M4	✗	✗	✓
✗	✗	✗	✗	M5	✗	✗	✗

Figura 4.22 – Esquema de uso dos equipamentos, transferências entre eles e processos

O procedimento genérico de simulação de todas as conexões é semelhante, mas devido às particularidades de cada conexão e de cada equipamento, os detalhes da simulação são distintos entre si. Nas seções seguintes descreve-se com maior detalhe como são realizados os procedimentos de simulação dos equipamentos, das transferências entre eles e das conexões.

4.3.3 Simulação dos Equipamentos

Os equipamentos possuem características individuais e, portanto, são simulados de formas distintas. A seguir apresenta-se como é realizada a simulação dos equipamentos presentes na fabricação de peças com simetria rotacional.

4.3.3.1 Depósito de Material em Bruto

A simulação de um depósito de material em bruto é simples. Dos materiais em bruto que estão estocados no depósito, relacionam-se aqueles que são constituídos do mesmo material que a peça e cuja seção transversal seja adequada para a produção da peça. No caso de peças com simetria rotacional, buscam-se os materiais em bruto cujo diâmetro externo seja maior que o diâmetro máximo da peça, acrescido do sobremetal necessário. Se a peça possuir um furo interno passante, utilizam-se preferencialmente materiais em bruto em forma de tubo

com diâmetro interno menor que o furo da peça reduzido do sobremetal correspondente. Se mais de um material em bruto satisfizer as condições necessárias, eles são ordenados de acordo com a área da seção transversal. Quanto menor a área, menor será o volume e também a massa da peça em bruto obtida após ser cortada pela serra. Com isso, economiza-se tanto no custo de material quanto no custo de usinagem, pois o volume final da peça pronta independe da peça em bruto a partir da qual ela foi usinada. A **Figura 4.23** mostra o princípio de simulação do depósito de material em bruto, na qual somente os materiais em bruto adequados à fabricação da peça são escolhidos.

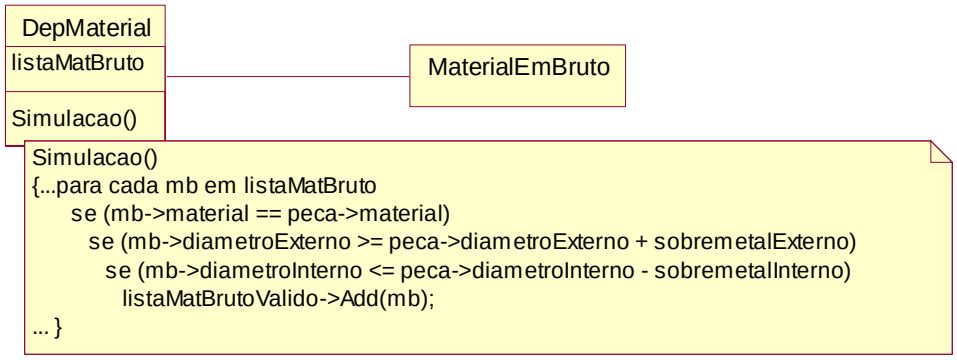


Figura 4.23 – Simulação do depósito de material em bruto

4.3.3.2 Depósito de Peça em Bruto

A simulação do depósito de peças em bruto é semelhante à do depósito de material em bruto. Além das condições impostas na escolha do material em bruto, o comprimento da peça em bruto existente no depósito deve também ser maior que o comprimento final da peça a ser produzida, acrescido do sobremetal necessário em ambas as extremidades. As peças em bruto que puderem ser utilizadas para a produção da peça são ordenadas pelo seu volume. Quanto menor o volume menor é o custo de produção da peça. A **Figura 4.24** apresenta como é realizada a simulação do depósito de peça em bruto. Somente as peças em bruto adequadas à fabricação da peça são escolhidas.

4.3.3.3 Depósito de Palete

A simulação do depósito de palete é uma das mais simples de todas. Verifica-se apenas se o depósito está apto a receber a base do palete correspondente àquele do palete que está fluindo pelo sistema de fabricação. O depósito de paletes, como qualquer outro equipamento, possui a lista de bases de palete que pode receber (Figura 4.15c).

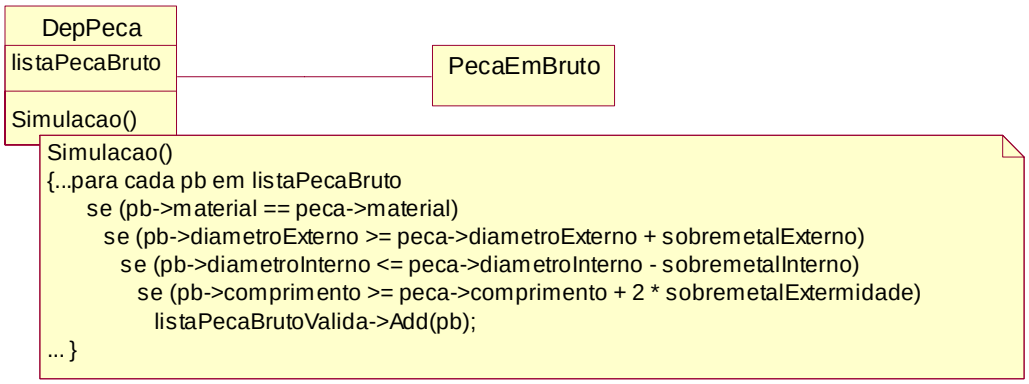


Figura 4.24 – Simulação do depósito de peça em bruto

Entretanto, se o depósito de palete recebe de um equipamento antecessor a peça para ser armazenada sobre um palete e não um palete com a peça já armazenada, deve-se efetuar a simulação da armazenagem da peça sobre cada um dos possíveis paletes que o depósito pode estocar. A simulação da armazenagem da peça é dividida em duas partes. Primeiramente, compara-se a geometria da peça e das longarinas para determinar as possíveis posições relativas entre elas, que representem armazenagens válidas. Para isso, confronta-se cada par de superfície plana e superfície cilíndrica da peça com o par de superfícies de apoio e de limitação de curso horizontal das longarinas. A determinação geométrica da armazenagem é finalizada ao encontrar-se duas armazenagens individuais nas longarinas, idênticas ou não, que sejam compatíveis entre si para apoiar a peça em seus dois lados. A **Figura 4.25** mostra os dados geométricos de uma armazenagem da peça em duas longarinas para apoio na horizontal. Na figura são apresentados a orientação da peça, o vetor posição da peça e o vetor posição da longarina da direita em relação à longarina da esquerda. O vetor posição da peça em relação à longarina da direita não é mostrado, para simplificar a figura.

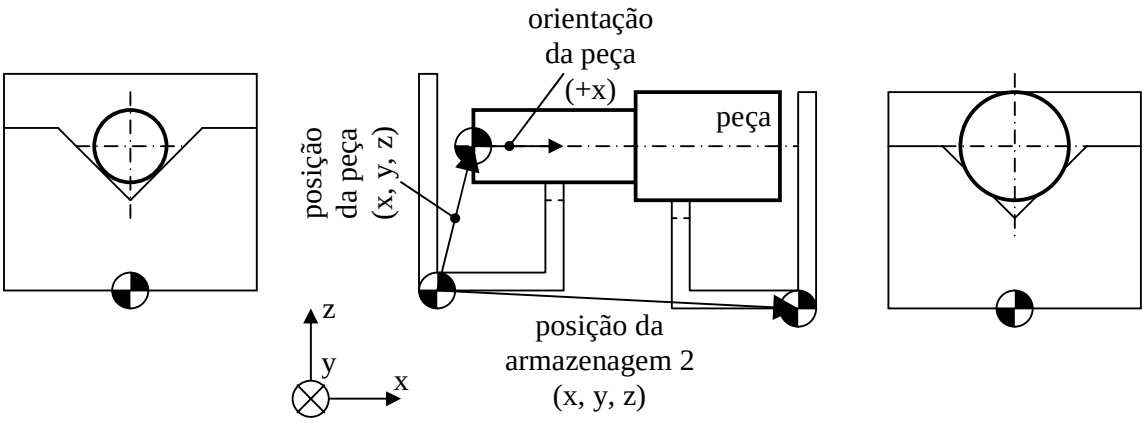


Figura 4.25 – Dados da armazenagem em longarinas

Na segunda parte comparam-se as armazenagens encontradas pela análise geométrica entre si, para ordená-las segundo um critério de qualidade. O modelo funcional, que qualifica as armazenagens, descreve a interação de esforços entre as longarinas e a peça, devida às acelerações sofridas durante o transporte. Em um primeiro passo, determina-se a aceleração necessária do palete de transporte para provocar o desequilíbrio da peça. Em seguida, após cessar o movimento acelerado, é verificado se a peça retorna à condição original de armazenagem por efeito da força gravitacional. O tratamento matemático necessário, aplicável a longarinas prismáticas para apoio na vertical (Figura 4.12), é mostrado em detalhes por Massaroppi e Masiero (2003). Para outros tipos de longarinas tem-se um equacionamento semelhante, capaz de determinar a aceleração necessária para o desequilíbrio da peça e o possível retorno à condição original após o movimento acelerado e, portanto, capaz de comparar a qualidade das diversas armazenagens entre si.

A **Figura 4.26** apresenta a modelagem da simulação do depósito de palete. Cada armazenagem parcial sobre uma longarina é descrita pela posição e orientação da peça em relação à longarina. A armazenagem sobre o palete é descrita por duas armazenagens sobre longarinas compatíveis entre si, pela posição e orientação da peça em relação à longarina principal e pela posição da longarina secundária em relação à principal.

4.3.3.4 Serra

Por ser um equipamento de fabricação, a serra provoca uma mudança da geometria do material em processo. Uma serra recebe materiais em bruto sem comprimento definido, corta-os em um comprimento apropriado, gerando uma peça em bruto. A simulação consiste em verificar se a dimensão externa do material em bruto e se o comprimento mínimo da peça em bruto a ser cortada estão dentro dos limites de utilização do sistema de fixação e de corte da serra. A **Figura 4.27** mostra como é realizada a simulação da serra. A serra é o equipamento de destino de uma conexão que movimenta materiais em bruto. Logo, a simulação da serra é realizada após outras simulações dentro da mesma conexão de forma que os materiais em bruto disponíveis já são conhecidos. O resultado da simulação é dado pelo subconjunto dos materiais em bruto que podem ser cortados pela serra.

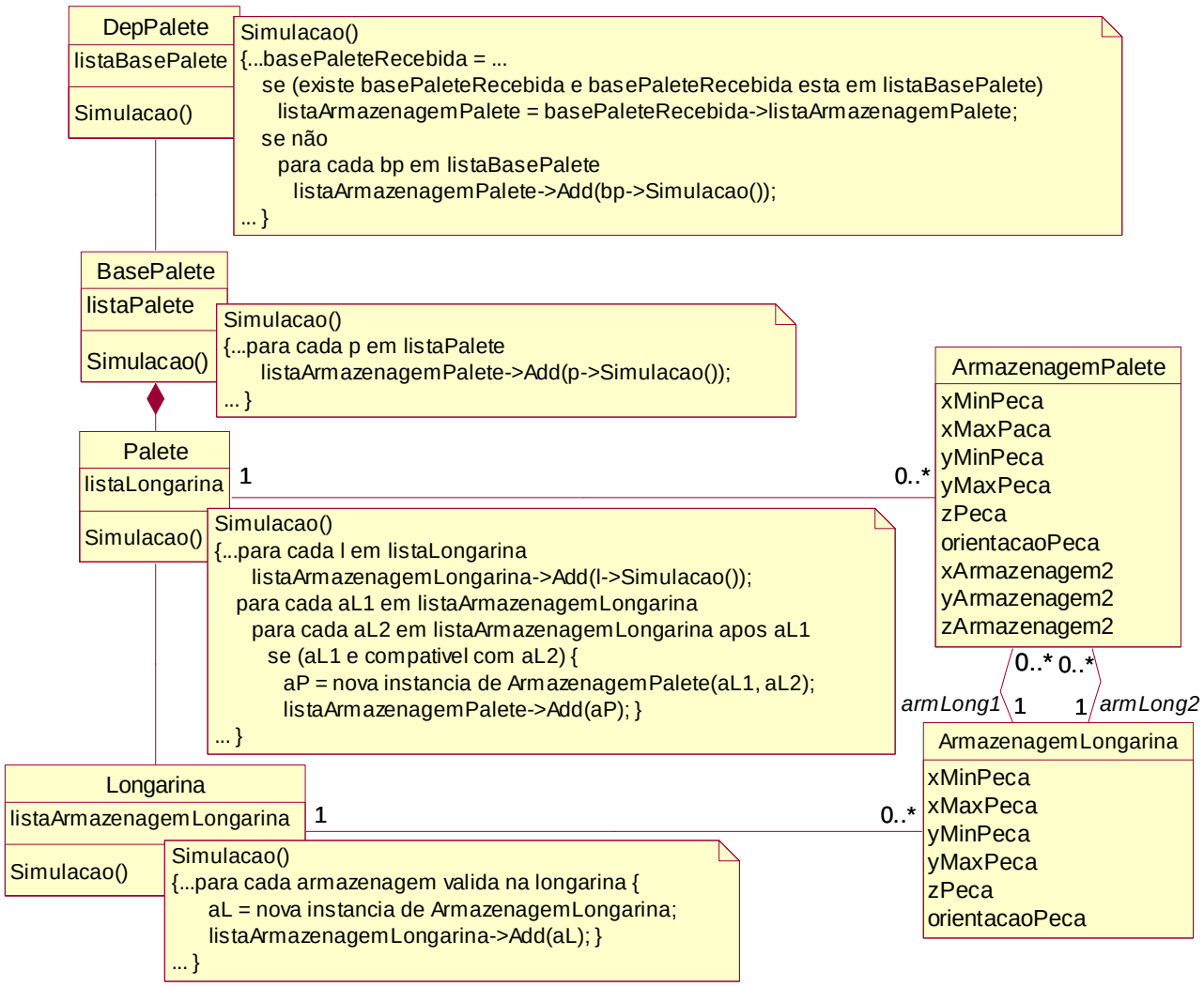


Figura 4.26 – Simulação do depósito de palete

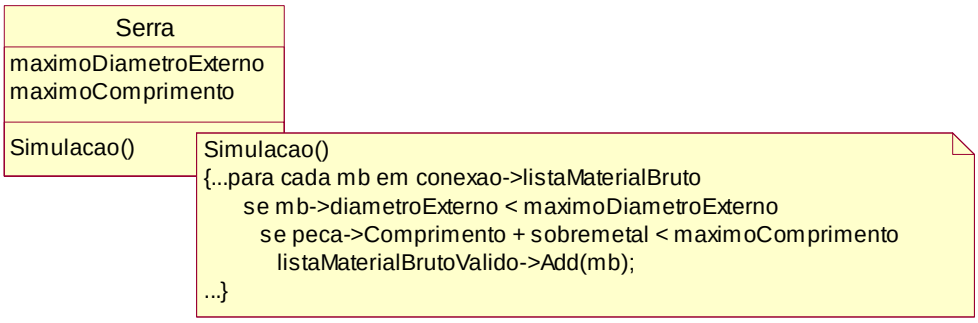


Figura 4.27 – Simulação da serra

4.3.3.5 Robô

A simulação do robô não é realizada diretamente, mas sim pela simulação da fixação com a sua garra. Inicialmente, compara-se a geometria da pinça e da peça para determinar as

possíveis posições relativas entre elas, que representem fixações válidas. Para a garra de duas pinças esquematizada na Figura 4.7, confronta-se cada superfície cilíndrica da peça com o prisma de fixação da pinça. Com exceção da posição axial da peça, determinam-se as coordenadas relativas entre a garra e a peça. Dentro do limite de posicionamento axial da peça, em que a pinça atua sobre a mesma região cilíndrica, define-se uma única fixação com um grau de liberdade em aberto. Na utilização da garra, um ponto desse limite é escolhido e a posição axial da peça é garantida pela força de atrito no contato entre as superfícies da pinça e da peça. A **Figura 4.28** apresenta os dados geométricos da fixação de uma peça com uma garra de duas pinças prismáticas. Os dados geométricos são a posição e a orientação da peça em relação à garra.

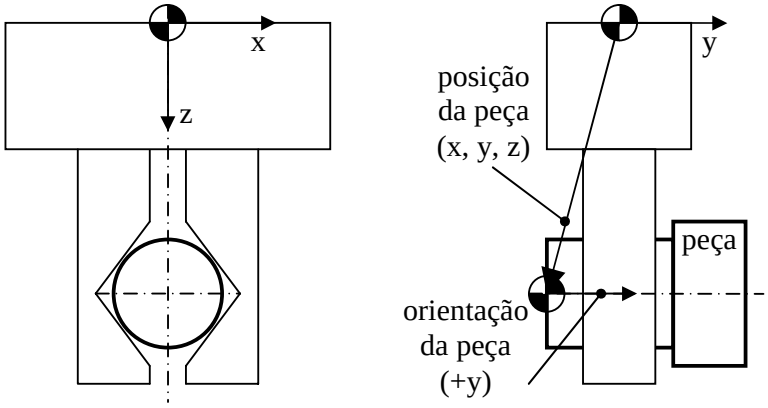


Figura 4.28 – Dados da fixação com a garra

Finalmente, define-se um índice de qualidade para comparar as fixações encontradas entre si. Quando o robô manipula a peça ao longo de uma dada trajetória, a pinça exerce esforços sobre a peça para que ela sofra as acelerações correspondentes, de acordo com as equações de movimento de Newton (Hibeller, 1999). Em função da excentricidade da peça na garra e da região de contato entre a pinça e a peça, determina-se o acréscimo do esforço de contato entre elas quando o robô descreve uma trajetória crítica. A melhor fixação é aquela que está associada ao menor aumento do esforço de contato. O tratamento matemático da interação de esforços da fixação com garras de duas pinças prismáticas é apresentado em detalhes em Massaroppi e Masiero (2002). Para outros tipos de garras e pinças existe um equacionamento semelhante, que determina o acréscimo do esforço de contato.

A **Figura 4.29** mostra o procedimento de simulação do robô. Uma garra pode não fixar a peça ou fixá-la de várias formas. Cada fixação está sempre relacionada com uma única garra. Cada fixação é descrita pelo índice que a qualifica e pela posição e orientação da peça em relação à garra.

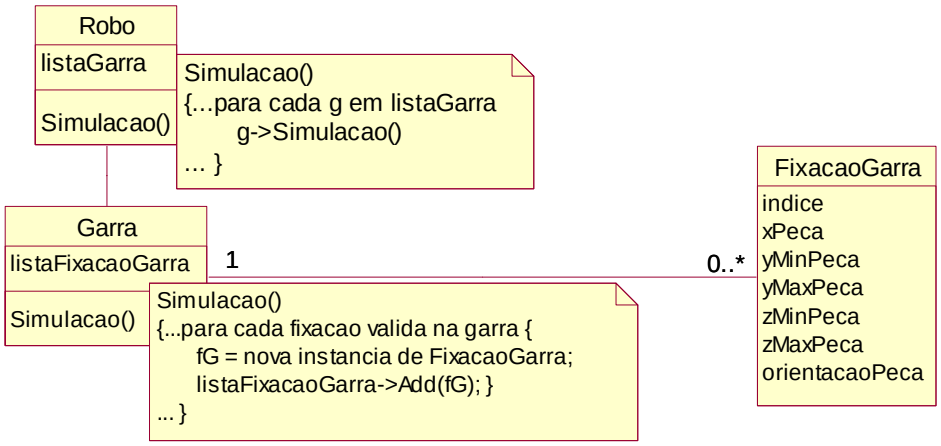


Figura 4.29 – Simulação do robô

4.3.3.6 Torno

Assim como no robô, a simulação do torno não é realizada diretamente, mas sim pela simulação da fixação com a sua placa. Em uma primeira fase, compara-se a geometria da placa de fixação e da peça para determinar as possíveis posições relativas entre elas que representem fixações válidas. Para a placa de três castanhas da Figura 4.4 confronta-se cada par de superfícies cilíndrica e plana da peça com cada par de superfícies de fixação e de apoio da placa. Com isso, determinam-se as coordenadas relativas entre a placa e a peça que representam fixações válidas. A **Figura 4.30** mostra os dados geométricos da fixação com uma placa de três castanhas.

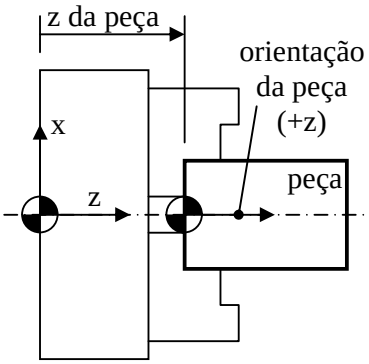


Figura 4.30 – Dados da fixação com a placa

Define-se na segunda fase um índice de qualidade para comparar as fixações encontradas entre si. Ao realizar-se a usinagem de uma peça fixada na placa do torno, a solicitação crítica ocorre quando a força de corte está aplicada na extremidade livre da peça. Em função da excentricidade da força aplicada em relação à fixação, ocorre um acréscimo do

esforço na região de contato entre a castanha e a peça, de acordo com a teoria de contato de Hertz (Niemann, 1981). A melhor fixação é aquela que está associada ao menor aumento do esforço de contato. O tratamento matemático da interação de esforços entre a castanha e a peça em uma fixação com placas de três castanhas pode ser visto em detalhes em Massaroppi e Masiero (2001a; 2001b). Para outros tipos de placas existe um equacionamento semelhante, que determina o acréscimo de carga no contato.

A **Figura 4.31** apresenta a modelagem da fixação com placas. Uma placa pode estar associada a uma quantidade qualquer de fixações, mas as fixações necessariamente estão associadas a uma única placa. Cada fixação é descrita pelo índice que a qualifica e pela posição e orientação da peça em relação à placa.

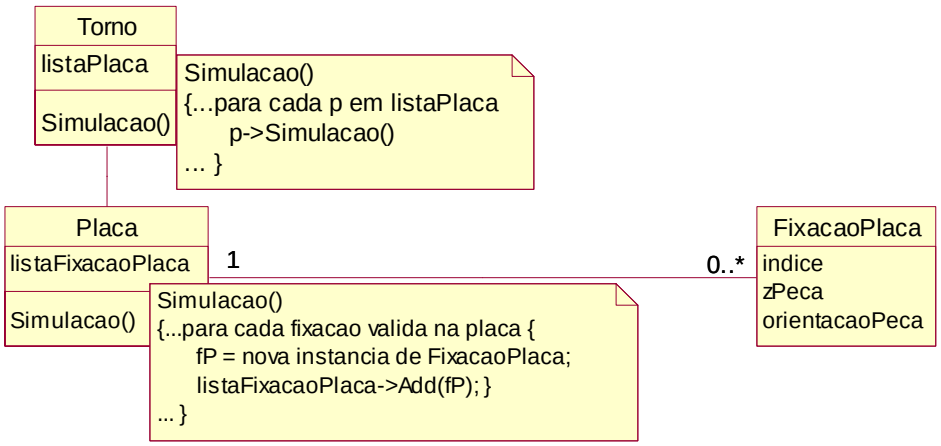


Figura 4.31 – Simulação do torno

4.3.3.7 Transportadores

No método proposto está prevista a existência de transportadores de paletes e de materiais em bruto. Para o transporte de paletes normalmente são utilizados os transportadores por correia, por rolos ou os veículos autoguiados (do inglês *AGV*). Esses transportadores, como qualquer outro equipamento, possuem a lista de bases de paletes que estão aptos a receber (Figura 4.15 c). A simulação desses equipamentos resume-se a verificar se a base do palete em fluxo pertence à sua lista. Os transportadores de material em bruto não são simulados diretamente. Eles são utilizados para retirar o material em bruto de um depósito ou para encaminhá-los a uma serra. Admite-se que o resultado da simulação do depósito de material em bruto ou da serra seja extrapolável para o transportador. Estes três equipamentos, depósito, transportador e serra, são normalmente comercializados como uma única unidade denominada centro de serramento (Kasto, 2002). A aplicação deste método de simulação é

facilitada quando o centro de serramento é descrito como três equipamentos distintos, pois a conexão correspondente é vista por um cliente como as demais. Os centros de serramento são construídos para uma determinada capacidade correspondente à capacidade individual de cada uma de suas partes. Portanto, adotar o resultado do depósito de material em bruto ou da serra como resultado da unidade de transporte corresponde à realidade destes equipamentos.

4.3.4 Simulação das Transferências entre Equipamentos

Da mesma forma que os equipamentos, as transferências entre eles possuem características individuais e, conseqüentemente, devem ser simuladas de forma distintas. As transferências entretanto, ocorrem sempre entre um equipamento de movimentação e outro tipo de equipamento. Nas seções seguintes apresenta-se como são realizadas as simulações das transferências entre equipamentos presentes na fabricação de peças com simetria rotacional. As transferências podem ocorrer no sentido como são descritas ou, na maioria dos casos, também no sentido inverso.

4.3.4.1 Depósito de Palete para Transportador de Palete

Assim como a simulação individual desses dois equipamentos, a simulação da transferência do palete entre eles é simples. Se os dois equipamentos estão aptos a receber a base do palete que está fluindo pelo sistema de fabricação, admite-se que a transferência pode ser realizada.

4.3.4.2 Transferências com o Transportador de Material em Bruto

A simulação da transferência envolvendo o transportador de materiais em bruto não é realizada diretamente. Ele é utilizado para retirar o material em bruto do depósito ou para alimentar a serra. Da mesma forma que o resultado da simulação do depósito de material em bruto ou da serra são extrapolados para o transportador, são também extrapolados para a transferência entre eles. Como mencionado na seção 4.3.3.7, a adoção desse procedimento corresponde à realidade dos sistemas de fabricação, pois os três equipamentos envolvidos formam normalmente uma única unidade produtiva.

4.3.4.3 Robô para Torno

Da mesma forma que a simulação individual dos robôs e dos tornos não é realizada diretamente, a simulação da transferência de peças entre eles é também realizada utilizando os

resultados de seus respectivos dispositivos. A simulação da transferência entre a garra e a placa, dispositivos destes equipamentos, é realizada pelo controle de colisão no instante em que a peça é transferida entre elas. A posição relativa entre a placa e a peça é definida pela fixação com a placa. Na simulação da fixação com a placa determina-se também o espaço ocupado pela placa, como é apresentado na **Figura 4.32**. Inicialmente, determinam-se os pontos que representem vértices externos. Além do vértice associado à placa, têm-se também vértices referentes à tampa (não mostrada na figura) e à castanha. Por razões construtivas, a castanha pode ter degraus voltados para o lado externo e por isso, pode ser encontrado mais de um vértice associado à castanha (parte à esquerda da figura). Para cada vértice determina-se uma região, dada pelo terceiro quadrante em relação ao vértice, na qual a garra não pode se encontrar no instante da transferência (parte central da figura). A forma geométrica obtida pela união das regiões geradas para cada vértice determina o espaço ocupado pela placa durante a fixação (parte à direita da figura). Este espaço não pode ser ocupado simultaneamente pela garra.

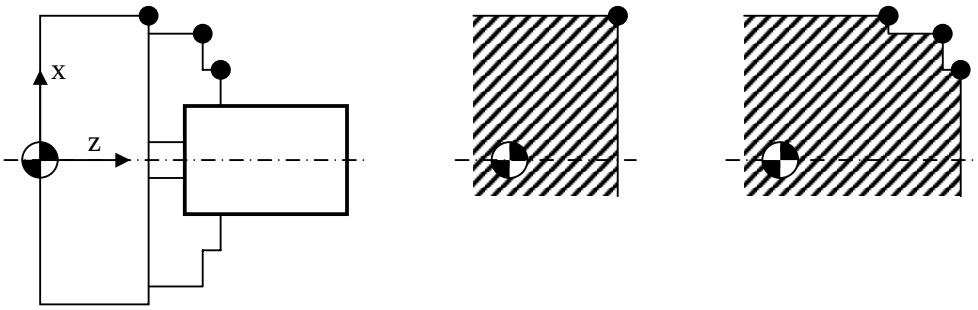


Figura 4.32 – Região ocupada pela placa durante a fixação

Para fixações com a garra sobre superfícies cilíndricas, a posição relativa entre a garra e a peça é definida a menos de um grau de liberdade na direção do eixo principal da peça. Conseqüentemente, a posição relativa entre a placa e a garra está definida, com exceção desse grau de liberdade. A garra pode se deslocar ao longo do eixo principal da peça, que tem o seu lugar geométrico definido em relação à placa. A distância mínima entre a placa e a garra é determinada pela comparação primeiro de cada ponto e depois de cada segmento de reta, que definem o contorno da garra com a região ocupada pela placa. Dentro do limite de variação da posição da garra definido pela fixação, busca-se o intervalo no qual não ocorre colisão entre a placa e a garra. A transferência da peça entre a placa e a garra depende da existência deste intervalo.

A **Figura 4.33** mostra o procedimento de simulação da transferência da peça entre a placa e a garra. A transferência não ocorre entre um robô e um torno qualquer, mas somente entre aqueles que fazem parte de uma mesma conexão. A simulação da transferência está associada ao robô, o equipamento de movimentação da conexão, que participa das duas transferências que ocorrem no início e no fim do processo associado à conexão. O robô pode ou não estar apto a realizar transferências. Entretanto, as transferências estão sempre associadas ao mesmo robô. Cada transferência está associada necessariamente a uma fixação com a placa e a uma fixação com a garra. As fixações em contrapartida podem ou não estar associadas a transferências. Na transferência é determinada a posição relativa entre a placa, a garra e a peça. A figura apresenta também a relação entre a fixação com a placa e a região por ela ocupada. A região ocupada pela fixação é definida por pelo menos uma região simples (quadrante).

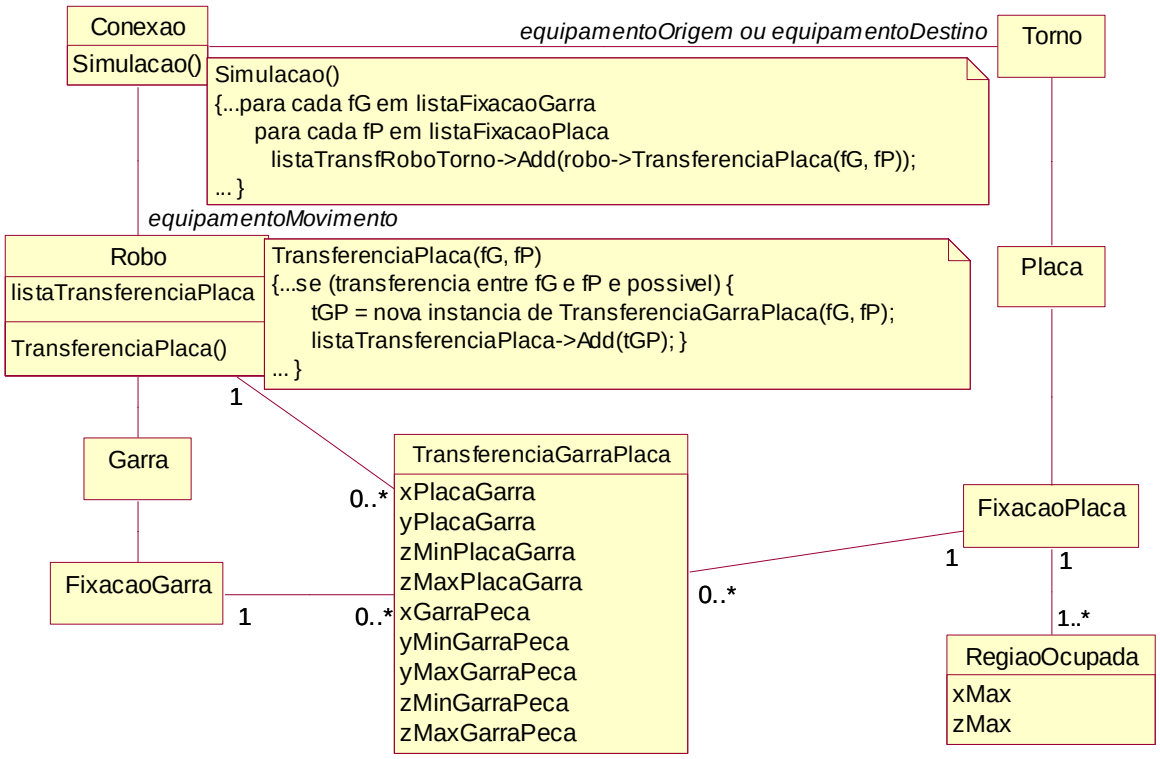


Figura 4.33 – Simulação da transferência entre o robô e o torno

4.3.4.4 Robô para Depósito de Palete

A transferência da peça entre o robô e o depósito de palete também é determinada pela simulação da transferência entre os respectivos dispositivos. A posição relativa entre as longarinas e a garra é determinada em função da fixação da peça com a garra e da

armazenagem da peça no palete. Para cada longarina determina-se a região por ela ocupada, de forma semelhante à fixação com a placa (Figura 4.32). Compara-se essa região com a linha de contorno da garra para obtenção da distância mínima entre a garra e cada uma das longarinas. Conseqüentemente, têm-se dois intervalos de posicionamento da garra, um determinado pela fixação da peça com a garra e outro pela comparação da geometria das longarinas e da garra. Se a interseção desses dois intervalos não for nula, tem-se a posição que a garra pode ocupar durante a transferência da peça entre a garra e as longarinas.

A **Figura 4.34** apresenta o procedimento de simulação da transferência da peça entre o robô e o depósito de palete. A transferência só ocorre entre um robô e um depósito de palete que pertencem a uma conexão. A simulação da transferência está associada ao robô, o equipamento de movimentação da conexão, que participa das duas transferências que ocorrem no início e no fim do processo associado à conexão. Uma quantidade qualquer de transferências pode estar associada ao robô. Cada transferência está associada a uma fixação com a garra e a uma armazenagem no palete. A fixação e a armazenagem não estão necessariamente associadas a uma transferência. Na transferência é determinada a posição relativa entre a garra, as longarinas e a peça.

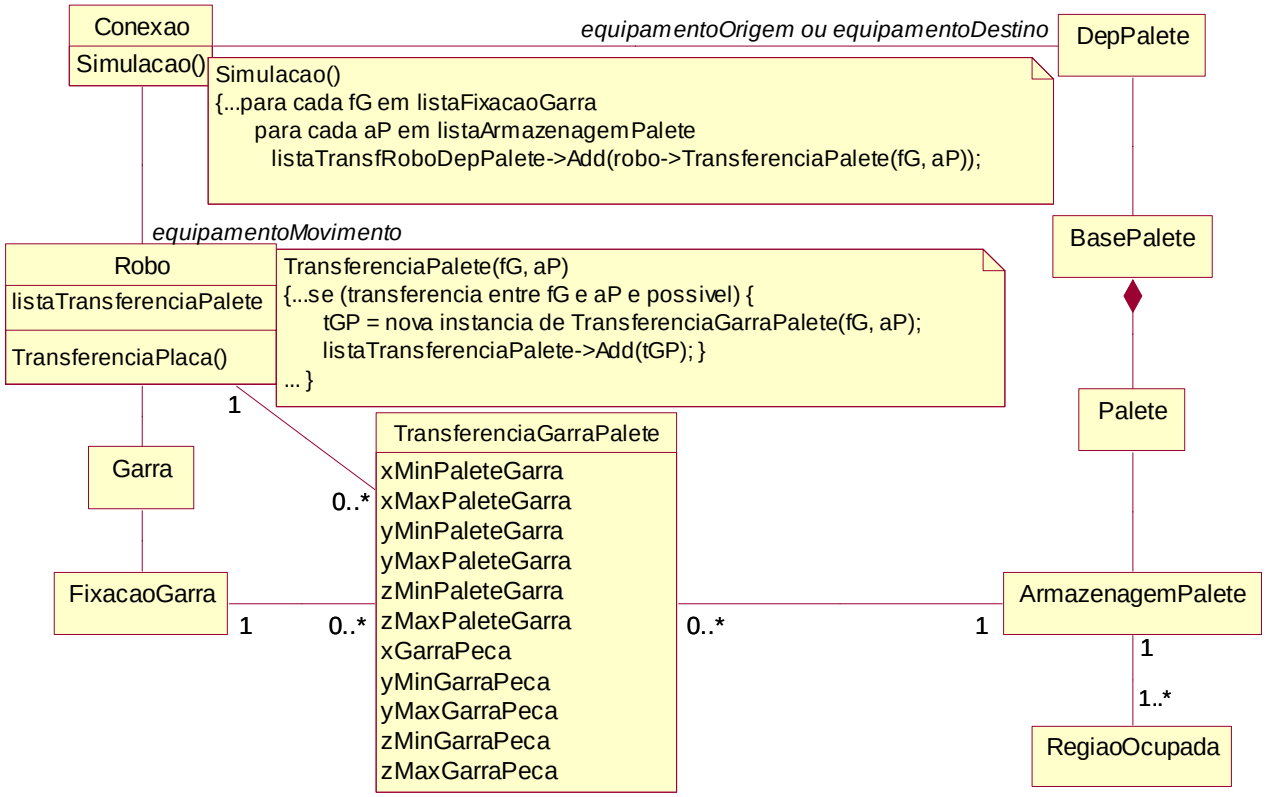


Figura 4.34 – Simulação da transferência entre o robô e o depósito de palete

4.3.4.5 Outras Transferências com o Robô

Além das transferências já descritas, os robôs são também utilizados para retirar a peça em bruto das serras e dos depósitos de peças em bruto. Tanto as mesas das serras como as plataformas dos depósitos de peças em bruto são superfícies horizontais planas com dimensões significativamente maiores que as peças em bruto. Conseqüentemente, os robôs devem aproximar-se pela direção vertical sem colidir com a base de apoio das peças em bruto. As peças em bruto podem estar na posição horizontal ou vertical e, portanto, a garra do robô deve fixar a peça perpendicularmente, como na Figura 4.28, ou alinhada com o seu eixo principal (z na Figura 4.28). Se a posição relativa entre a garra e a peça em bruto e entre a peça em bruto e a superfície de apoio forem de tal magnitude que impliquem na não ocorrência de colisão entre a garra e a superfície de apoio, a transferência é tida como válida.

4.3.5 Simulação das Conexões

Assim como os equipamentos e as transferências entre eles, as conexões também possuem características individuais que precisam ser simuladas de formas distintas. Nas seções seguintes mostra-se como as simulações das conexões entre equipamentos utilizados no fluxo de peças com simetria rotacional são realizadas.

4.3.5.1 Depósito de Material em Bruto para Serra

A conexão entre o depósito de material em bruto e a serra transporta materiais em bruto e o equipamento de transporte é denominado, de forma genérica, transportador de material em bruto. O processo associado a esta conexão recebe o nome de alimentação da serra. A alimentação da serra não é simulada diretamente. Como mencionado na seção 4.3.4.2, admite-se que os resultados da simulação do depósito de material em bruto e da serra sejam extrapoláveis para o transportador e para as respectivas transferências. Admite-se também, que esses resultados sejam utilizados como resposta da simulação do processo de alimentação desde o depósito de material em bruto até a serra, ou seja, todo material em bruto que for encontrado no depósito e puder ser cortado pela serra no comprimento adequado é solução para o processo.

Como a serra é um equipamento de fabricação e, conseqüentemente, responsável pela alteração da geometria da peça que está sendo produzida, deve-se também simular o processo de usinagem. No caso da serra, determina-se, inicialmente, o menor comprimento em que o material em bruto pode ser cortado em função da precisão e da resolução da serra e do

comprimento mínimo necessário do material em bruto para a fabricação da peça. A partir dos dados do material em bruto e do comprimento em que ele foi cortado são geradas as informações que descrevem uma peça em bruto.

4.3.5.2 Serra para Depósito de Palete

Na conexão entre a serra e o depósito de palete manipula-se uma peça em bruto com a utilização de um robô. Dependendo do ponto de vista, o processo associado a esta conexão recebe o nome de descarga da serra ou alimentação do palete de transporte. A peça em bruto é proveniente do serramento do material em bruto existente na serra. Na simulação das transferências realizadas pelo robô, para retirada da peça em bruto da mesa da serra e para sua colocação nos elementos de apoio do palete, obteve-se, entre outras informações, a faixa de coordenadas relativas entre a garra e a peça. A simulação da descarga da serra consiste em encontrar pares de transferências, no início e no fim da utilização do robô, que empreguem a mesma fixação com a sua garra e cujas faixas de coordenadas da peça em relação à garra possuam interseção não nula. Essa interseção exprime os valores que podem ser utilizados para que a garra retire a peça em bruto da mesa da serra e coloque-a no palete sem que ocorra colisão entre os dispositivos utilizados.

4.3.5.3 Depósito de Peça em Bruto para Depósito de Palete

Está conexão é semelhante à conexão anterior. Aqui também manipula-se uma peça em bruto utilizando-se um robô. Ao contrário da conexão anterior, a peça em bruto é proveniente de um depósito e não da serra, mas em ambos os casos ela é colocada em um palete de transporte. A simulação da conexão também é semelhante à anterior. Como resultado das duas transferências foram obtidas faixas de valores para as coordenadas relativas entre a garra do robô e a peça. Se a interseção destas duas faixas de coordenadas relativas às transferências não for nula, obtêm-se a faixa de coordenadas que pode ser utilizada para a realização do processo sem que ocorra colisão entre os dispositivos.

4.3.5.4 Depósito de Palete para Depósito de Palete

A conexão entre dois depósitos de palete é realizada por um transportador de paletes e o objeto transportado é o próprio palete. Da mesma forma que a simulação dos equipamentos e a simulação das transferências envolvidas, a simulação de todo o processo desta conexão também é simples. Se os três equipamentos, depósito de palete de origem, transportador de

palete e depósito de palete de destino, estão aptos a receber o palete e o objeto transportado é um palete, o processo de transporte pode ser realizado.

4.3.5.5 Depósito de Palete para Torno

O processo nesta conexão é realizado por um robô que manipula a peça em bruto desde o depósito de palete até o torno. Este processo recebe o nome de alimentação do torno. Cada uma das duas transferências, entre o depósito de palete e o robô e entre o robô e o torno, é caracterizada, entre outras informações, pela faixa de valores das coordenadas da peça em relação à garra do robô. Para cada par de transferências, no recebimento e na entrega da peça em bruto, buscam-se aqueles que são caracterizados pela mesma fixação com a garra e que a interseção das faixas de valores das coordenadas relativas entre a garra e a peça não sejam nulas. De todos os possíveis pares de transferências, o resultado da simulação da alimentação do torno é dado por aqueles que satisfazem as condições apresentadas acima.

Da mesma forma que a serra, o torno também é um equipamento de fabricação. Conseqüentemente, ele também é responsável pela alteração da geometria da peça e o processo correspondente, denominado torneamento, deve ser simulado no âmbito dessa conexão. O processo de torneamento é complexo quando comparado com a maioria das simulações realizadas no método aqui proposto. Como regra, o torneamento envolve duas usinagens distintas, uma sobre cada lado da peça em bruto. Inicialmente, interpreta-se o contorno da peça pronta como uma função, em que o raio da peça depende da posição axial considerada. A partir dessa abstração buscam-se os pontos de máximo dessa função. Esses pontos são candidatos a ponto de divisão entre as duas usinagens. Considerando-se a peça em bruto fixada na placa do torno (Figura 4.30), efetua-se a simulação da primeira usinagem, removendo-se a parcela do volume à direita do ponto de divisão das usinagens. Para cada um dos pontos de divisão e para a orientação considerada da peça, obtém-se uma das possíveis geometrias da peça semipronta.

O passo seguinte consiste em simular a refixação da peça semipronta. Inicialmente simula-se a utilização dos equipamentos envolvidos na refixação. Para isso, simula-se a fixação da peça semipronta com a garra de um dos robôs que participam de alguma conexão em conjunto com o torno, de acordo com a seção 4.3.3.5, e a fixação na própria placa do torno, mas com a restrição de que essa fixação ocorra no lado já usinado da peça, de forma semelhante àquela mostrada na seção 4.3.3.6. Em seguida simula-se a transferência da peça semipronta entre os dispositivos envolvidos. Tanto a transferência da primeira fixação na

placa para a garra, como a transferência da garra para a segunda fixação na placa seguem o procedimento apresentado na seção 4.3.4.3. Finalmente simula-se o processo de refixação. Buscam-se pares de transferências que são caracterizados pela mesma fixação com a garra e que tenham a interseção de suas faixas de coordenadas da peça em relação à garra não nula. Satisfazendo essas condições, garante-se que é possível realizar a refixação da peça semipronta. O emprego da garra para a refixação da peça semipronta deve ocorrer na faixa de interseção.

O passo final do processo de torneamento é a usinagem do lado ainda não usinado da peça semipronta. A simulação dessa usinagem é possível se o ponto de divisão entre as duas usinagens não for obstruído pela fixação com a placa. A região de obstrução da placa é a mesma que aquela utilizada para simular a transferência entre a placa e a garra (Figura 4.32).

4.3.5.6 Outras Conexões para o Torno

Além do torno poder ser alimentado com peças em bruto proveniente do depósito de paletes, existe a possibilidade delas serem originárias de uma serra ou de um depósito de peças em bruto. Nos dois casos a manipulação é realizada com um robô. As transferências da peça em bruto entre estes dois equipamentos e o robô foi descrita na seção 4.3.4.5. A partir do resultado destas transferências e do resultado da transferência da peça em bruto do robô para o torno (seção 4.3.4.3) determina-se a possibilidade de realização do processo de alimentação do torno com base no cálculo da interseção das posições relativas entre a garra e a peça em bruto. Para as fixações encontradas da peça em bruto na placa do torno deve-se efetuar a simulação do processo de usinagem da forma descrita na conexão anterior.

4.3.5.7 Torno para Depósito de Palete

Na conexão entre o torno e o depósito de palete manipula-se a peça com a utilização de um robô. O processo de retirada da peça pronta do torno e sua colocação em um palete de transporte recebe o nome de descarga do torno. A diferença entre a alimentação e a descarga do torno é que no primeiro caso manipula-se uma peça em bruto e no segundo uma peça pronta. A diferença entre as duas formas da peça é a sua geometria. Conseqüentemente, encontram-se fixações, com a placa e com a garra, e armazenagens sobre o palete distintas para as duas geometrias. Entretanto, o princípio de simulação dos equipamentos, das transferências da peça entre eles e do processo de manipulação (alimentação ou descarga) da

peça é o mesmo. Obviamente, após a simulação da descarga não é realizada a simulação da usinagem da peça.

4.3.5.8 Conexões Consecutivas

As conexões descritas nas seções anteriores não levam em consideração a posição da conexão dentro da seção e da rota que está sendo simulada. As conexões cujos equipamentos de origem são depósito de materiais em bruto ou peças em bruto são conexões iniciais e a sua simulação é realizada da maneira descrita nas respectivas seções. Entretanto, o procedimento de simulação das demais conexões é realizado de forma ligeiramente diferente. De acordo com as descrições das simulações dessas conexões, o primeiro passo consiste em simular o equipamento de origem. Como essas conexões são consecutivas a outras, a simulação de seu equipamento de origem é exatamente a mesma que a simulação do equipamento de destino da conexão antecessora. Essa simulação já foi realizada, pois a simulação das rotas ocorre no sentido de fluxo da peça. Nem todos os resultados da simulação do equipamento de destino da conexão antecessora podem ser empregados, devido a incompatibilidades na transferência ou no processo completo. Conseqüentemente, são formas de emprego do equipamento de destino que não ocorrem durante o fluxo da peça. Logo, na conexão sucessora, essas formas de emprego do equipamento (de origem da conexão sucessora) não devem ser utilizadas na simulação. Com isso, diminui-se o esforço necessário para simular o fluxo, pois por um lado o equipamento é simulado uma única vez e por outro parte dos resultados encontrados na simulação do equipamento não precisa ser utilizada para simular a transferência entre esse equipamento e o seguinte.

4.3.6 Simulação Inversa das Rotas

As conexões que compõem uma rota são simuladas no sentido do fluxo da peça. Os resultados da simulação do equipamento de destino de uma conexão só são utilizados na conexão sucessora se eles forem compatíveis com as transferências e os processos completos na conexão antecessora. Entretanto, se um resultado do equipamento de destino de uma conexão não puder ser utilizado pela conexão sucessora, ele não é invalidado da conexão antecessora. Para que todos os resultados obtidos sejam compatíveis com uma das possíveis formas de realização do fluxo completo da peça, efetua-se uma verificação da rota no sentido inverso do fluxo. Assim, a utilização O4 do equipamento de origem da conexão mostrada como exemplo na Figura 4.22, que não é compatível com nenhum dos processos que podem

ser realizados na conexão, deve ser invalidada na conexão antecessora no âmbito da simulação desta rota. Eventualmente, o resultado O4, obtido na simulação da conexão antecessora, pode ser utilizado no âmbito de outra rota, em que a peça flui para outra conexão e não para a apresentada na figura.

O procedimento de simulação inversa é realizado para todas as conexões da rota e os resultados que apresentam a incompatibilidade mostrada acima são invalidados. A invalidação do resultado de uma simulação de um equipamento em uma conexão intermediária pode levar à invalidação de outros resultados em conexões antecessoras a ela. Com isso, o universo total de resultados é reduzido, facilitando a sua análise. Porém, se o fluxo da peça no sistema de fabricação for possível, os resultados correspondentes não serão invalidados por este procedimento.

4.4 Análise e Escolha dos Resultados

O método de simulação do fluxo da peça em um sistema de fabricação apresentado na seção anterior pode levar a mais de um resultado em cada um dos passos do seu procedimento. Para comparar esses resultados definiram-se critérios aplicáveis a cada um dos passos, os quais classificam os resultados de acordo com a qualidade do emprego dos dispositivos utilizados pelos equipamentos. A classificação ocorre de acordo com o critério utilizado. O melhor resultado da simulação é aquele que engloba os melhores resultados individuais da simulação dos equipamentos. Conforme foi mencionado nas seções anteriores, a simulação da utilização dos dispositivos utilizados pelos equipamentos é realizada em duas fases. Inicialmente, compara-se a geometria da peça e do dispositivo para se obter as possíveis posições relativas que representem utilizações válidas desses dispositivos. Esta fase é comparativamente mais simples que a outra. Por isso, modelam-se os dispositivos de forma rigorosa o suficiente para encontrar todas as suas possíveis utilizações. Na segunda fase da simulação analisa-se a interação de esforços entre o dispositivo e a peça para classificar as utilizações dos equipamentos encontradas na primeira fase de acordo com um critério de qualidade. Como todo método de simulação de problemas complexos, nesta fase também se efetuam simplificações dos fenômenos simulados para viabilizar o seu emprego. Os critérios escolhidos para a classificação refletem somente os aspectos mais importantes, deixando de considerar outros que, eventualmente, seriam decisivos em empregos específicos de algum dispositivo. Conseqüentemente, a classificação dos resultados apresentada pelo método e aquela apresentada pela análise de um especialista do domínio podem diferir, principalmente,

porque não é possível englobar em um método de simulação todos os detalhes de funcionamento de um sistema de fabricação. Portanto, o método não deve apresentar uma solução única como resultado, mas sim um conjunto de soluções, ordenados de acordo com os critérios de qualidade. As soluções devem ser analisadas pelo usuário do método, levando-se em consideração todos os aspectos de seu conhecimento e todas as particularidades do sistema de fabricação real.

A escolha do resultado a ser utilizado para a fabricação da peça não pode ser aleatória. Ao escolher-se a forma de utilização de um equipamento, vários resultados associados aos demais equipamentos do sistema de fabricação são invalidados, por incompatibilidade com a escolha efetuada. Portanto, a escolha também deve seguir um critério. O objetivo principal de um sistema de fabricação é a produção da peça. Além disso, a fabricação normalmente é o procedimento mais caro dentre aqueles realizados nos sistemas de fabricação. Ao decidir-se pela produção automática, além da fabricação, o transporte, a manipulação, a estocagem e o teste da peça tornam-se também importantes, mas sempre em um nível inferior àquele do objetivo principal. Como conseqüência, é obvio que a escolha deve iniciar pelos equipamentos que realizam a função fabricação do fluxo de material. Para a fabricação de peças com simetria rotacional, a operação de torneamento é consideravelmente mais importante que a de serramento, ou seja, deve-se inicialmente escolher a forma como a peça deve ser torneada e só depois como o material em bruto deve ser cortado. As operações de manipulação são escolhidas em seguida, pois também são realizadas diretamente sobre a peça. Se os testes realizados exigem uma atuação direta sobre a peça, a escolha de como eles são realizados deve ser prioritária em relação à escolha das outras funções. Se os testes são realizados indiretamente, as escolhas da forma de utilização dos equipamentos de teste, de transporte e de estocagem podem ocorrer em qualquer ordem.

Em sistemas de fabricação de maior complexidade, a responsabilidade de operação dos equipamentos do sistema de fabricação pode estar dividida entre diversas pessoas. Aquelas que não são responsáveis pelos equipamentos prioritários recebem como alternativa de escolha apenas o subconjunto das possibilidades compatíveis com as opções já realizadas. Ocorrem casos em que a diferença de qualidade das várias opções de utilização de um equipamento é insignificante. Entretanto, a decisão tomada em um nível de prioridade superior pode invalidar as opções de melhor qualidade de um nível de prioridade inferior. A fim de evitar essas distorções, o processo de escolha prevê que as opções já realizadas possam ser revogadas quando necessário, mas na ordem inversa em que foram efetuadas. O

procedimento de escolha de utilização de todos os equipamentos e dispositivos tem como resultado a folha de processo para produção da peça que foi simulada. Após a obtenção da folha de processos, os dados correspondentes são liberados para a programação da fabricação da peça.

4.5 Considerações Finais

Neste capítulo apresentou-se um método para simular o fluxo de uma peça em sistemas de fabricação. Descreveram-se as informações básicas para emprego do método e os detalhes do procedimento de simulação. Como o método conduz a mais de uma resposta válida, também mostrou-se como é realizada a análise e a escolha dos resultados a serem utilizados para a produção da peça. Em conjunto com o método apresentaram-se também o processo de modelagem e o modelo conceitual desenvolvido para apoio à implementação computacional das partes relevantes.

5 Construção de um *Framework* para Simulação do Fluxo de Peças

5.1 Considerações Iniciais

No capítulo anterior apresentou-se um método de simulação do fluxo de peças com simetria rotacional. O processo de modelagem utilizado e o modelo conceitual obtido são típicos do desenvolvimento de uma aplicação específica para um sistema de fabricação. Observando-se o modelo obtido verifica-se a conveniência da divisão funcional em três partes. A primeira parte deve ser dedicada à coleta e armazenagem das informações básicas que descrevem o sistema de fabricação. A simulação do fluxo da peça fica sob responsabilidade da segunda parte. Como a simulação pode resultar em mais de uma alternativa para realização do fluxo, a análise e a escolha da resposta mais adequada são efetuadas na terceira parte.

O método apresentado abrange uma parte importante do domínio da manufatura. Efetuando-se uma análise mais detalhada, encontram-se diversos pontos que podem ser flexibilizados para aumentar a abrangência do método. A quantidade desses pontos é significativa, de modo a viabilizar o desenvolvimento de um *framework* que pode ser adaptado para simular um número maior de sistemas de fabricação.

Na seção 5.2 apresenta-se a arquitetura escolhida para implementação do *framework*. Como a implementação é dividida em três módulos, a maneira como é realizada a integração entre eles é mostrada na seção 5.3. A seção 5.4 apresenta a solução encontrada para realizar a persistência das informações contidas nos pontos variáveis do *framework*. A transformação do modelo conceitual de uma aplicação, mostrado no capítulo anterior, no modelo de um *framework* é apresentada na seção 5.5. Na seção 5.6 o *framework* proposto é comparado com dois outros freqüentemente presentes na literatura. A seção 5.7 mostra como instanciar o *framework* para criar uma aplicação específica. Na seção 5.8 apresentam-se a plataforma de hardware e o software utilizados para desenvolver o *framework* e criar uma aplicação, além de alguns dados que quantificam a implementação As considerações finais sobre este capítulo são apresentadas na seção 5.9.

5.2 Arquitetura do *Framework*

Mesmo que o método apresentado anteriormente fosse abstraído para outras geometrias e, portanto, para processos de fabricação diferente do torneamento, aplicável a peças com simetria rotacional, e mesmo tendo sido constatada a viabilidade do desenvolvimento de um *framework* para geração de aplicações específicas para cada sistema de fabricação, a implementação do método em três partes continua sendo conveniente.

5.2.1 Arquitetura Básica

A conveniência de dividir-se o método em três partes está presente na arquitetura do *framework*. A **Figura 5.1** apresenta uma divisão funcional das aplicações criadas a partir do *framework* em três módulos, que se comunicam com o auxílio de um sistema de persistência de informações. O primeiro deles, denominado Módulo de Configuração, auxilia o usuário a descrever de forma estruturada o sistema de fabricação em que será simulado o fluxo da peça. Este módulo é executado uma única vez, quando o usuário relata as informações do sistema de fabricação existente ou, eventualmente, quando ocorre qualquer mudança no ambiente fabril e as alterações precisam ser informadas ao sistema. Para facilitar esta tarefa, o módulo é composto por uma interface gráfica, que guia o usuário na inserção de informações consistentes. Toda a verificação da consistência entre os dados deve ser efetuada antes que ocorra a persistência das informações.

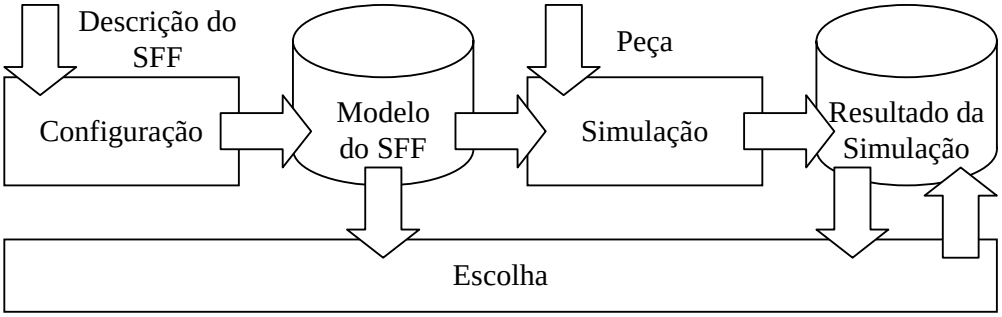


Figura 5.1 – Arquitetura básica do *framework*

O segundo módulo, chamado de Módulo de Simulação, é responsável pela simulação do fluxo de uma determinada peça no ambiente fabril descrito no primeiro módulo. Este módulo deve ser executado sempre que se desejar verificar a possibilidade de fabricação de uma peça. Todas as possibilidades encontradas são armazenadas no sistema de persistência, ordenadas segundo os critérios que qualificam a utilização dos equipamentos que compõem o sistema de fabricação.

As informações geradas nos dois módulos são conceitualmente diferentes. Essa diferença é mostrada na figura com a representação lógica de dois sistemas de persistência de informações. O primeiro grupo de informações, denominado Modelo do SFF, representa a fábrica e o segundo, Resultado da Simulação, descreve o processo de fabricação da peça, ou seja, a interação dos equipamentos e dispositivos com a peça. Os sistemas de armazenamento das informações podem ser implementados em separado, como é mostrado na Figura 5.1, ou em conjunto.

O resultado da simulação pode não ser único. Nesse caso, os resultados encontrados são ordenados segundo critérios de qualidade que abrangem apenas os aspectos mais importantes do funcionamento dos equipamentos e dispositivos do sistema de fabricação. O melhor resultado encontrado na simulação e o resultado da análise de um especialista podem não coincidir, principalmente porque não é possível embutir em uma simulação todos os detalhes de funcionamento de um sistema de fabricação. Para sanar essa divergência, existe o terceiro módulo, denominado Módulo de Escolha. O usuário indica entre as possíveis alternativas de fabricação da peça, como cada equipamento deve ser utilizado no processo de produção da peça. A opção feita pelo usuário deve ser registrada no sistema de persistência.

5.2.2 Características do Usuário

Como foi mencionado na seção anterior, os três módulos da arquitetura executam diferentes funções. Conseqüentemente, a qualificação dos usuários e a forma como eles interagem com cada um dos módulos também são distintas. A seguir descrevem-se as características desejáveis e a forma de interação dos usuários dos três módulos.

O usuário do Módulo de Configuração deve estar apto a interpretar as informações contidas em catálogos, desenhos e prospectos técnicos dos diversos equipamentos e dispositivos que compõem o sistema de fabricação. Ele deve também conhecer a forma como os dispositivos e os equipamentos foram modelados para poder associar as informações técnicas aos parâmetros do modelo. O usuário cria inicialmente um sistema de fabricação para ativar os menus que dão acesso às janelas de diálogo para definição de cada um dos elementos que o compõem. A **Figura 5.2** mostra o menu principal do Módulo de Configuração de uma aplicação desenvolvida a partir do *framework*. O item do menu denominado “Arquivo” dá acesso à definição ou à recuperação das informações de um sistema de fabricação. Após ter sido escolhido o sistema denominado “ifms”, ele passa a fazer parte do cabeçalho do menu da aplicação. Simultaneamente, o item do menu “Configuração” torna-se ativo e possibilita que a

descrição dos elementos que compõem a sistema de fabricação seja efetuada. Parte dos subitens mostrados na figura possuem um triângulo preto à sua direita. Essa marcação indica que existe uma nova subdivisão do menu. Os demais subitens dão acesso direto à respectiva janela de diálogo.

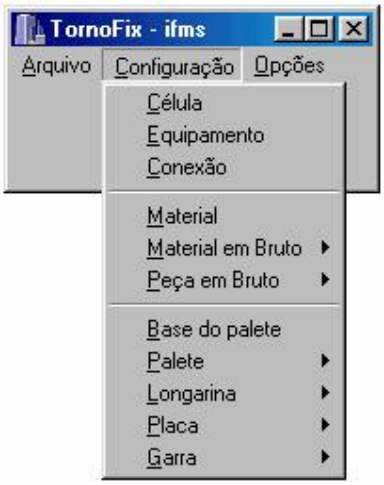


Figura 5.2 – Menu principal do Módulo de Configuração

A ordem de descrição dos elementos do sistema de fabricação deve respeitar algumas regras, mas em vários pontos ela é escolhida pelo usuário. O modelo apresentado na seção 4.2 é composto por diversos relacionamentos entre as classes de definição desses elementos. Quando o relacionamento é representado por uma agregação, a definição do elemento agregador deve ser efetuada antes do elemento agregado. Por isso, um equipamento só pode ser definido após a existência da célula correspondente (a Figura 4.1 mostra a agregação entre eles). Em contrapartida, a ordem de descrição dos equipamentos e dos dispositivos é definida pelo usuário. As janelas de diálogos relativas às classes com relacionamentos do tipo M::N permitem acesso às informações da própria classe e do relacionamento. Tanto na definição dos equipamentos como dos dispositivos, por exemplo, o usuário tem a possibilidade de relacioná-los entre si. Por um lado, a janela de diálogo de um equipamento apresenta a relação de dispositivos já definidos que ele pode utilizar e por outro a janela de diálogo dos dispositivos apresenta a relação de equipamentos onde ele pode ser utilizado.

A **Figura 5.3** apresenta a janela de diálogo para definição de uma longarina com prismas para apoio da peça na horizontal. Na parte superior da janela de diálogo observa-se que ela pode ser utilizada para inclusão, modificação, visualização ou exclusão de informações do sistema de persistência. Logo abaixo têm-se à esquerda as informações genéricas de uma longarina e à direita, a indicação dos relacionamentos com os paletes existentes. Na parte central encontram-se as informações geométricas específicas da longarina

para apoio de peças na horizontal. Na parte inferior da janela de diálogo são apresentados os desenhos da longarina para a conferência visual das informações geométricas.

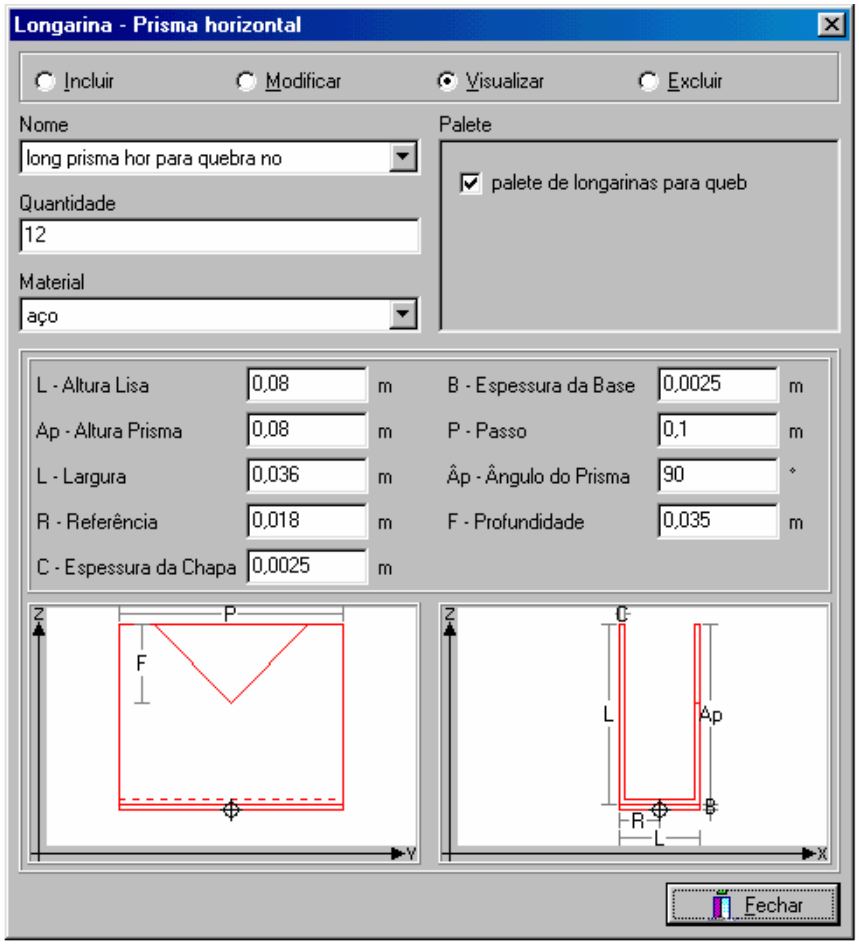


Figura 5.3 – Janela de diálogo da longarina prismática para apoio de peças na horizontal

O usuário do Módulo de Simulação não precisa ter conhecimento técnico sobre o sistema de fabricação, pois a interação necessária é bem simples. O módulo apresenta inicialmente uma janela de diálogo, em que o usuário escolhe o sistema de fabricação e a peça para que seja efetuada a simulação correspondente. A partir da escolha, o módulo é executado até o fim e as opções de fabricação encontradas são armazenadas no sistema de persistência.

Ao contrário dos anteriores, o Módulo de Escolha deve ser utilizado idealmente por mais de um usuário. Associado a cada um dos equipamentos que participa do fluxo da peça existe um processo de escolha. Em função do nome do usuário e da senha de entrada no sistema, tem-se acesso ao processo de escolha relacionado a um determinado equipamento. Cada usuário não precisa ter a mesma qualificação que o usuário do Módulo de Configuração, mas deve estar habilitado a avaliar os resultados relacionados ao equipamento sob sua

responsabilidade. Portanto, é necessário que ele possua um conhecimento detalhado da utilização do respectivo equipamento. O seu conhecimento dos demais equipamentos é desejável, porém, pode ser em um nível inferior. Se o equipamento for prioritário no processo de escolha, ela pode ser realizada, caso contrário a escolha só é possível se aquelas relativas aos equipamentos prioritários já foram efetuadas.

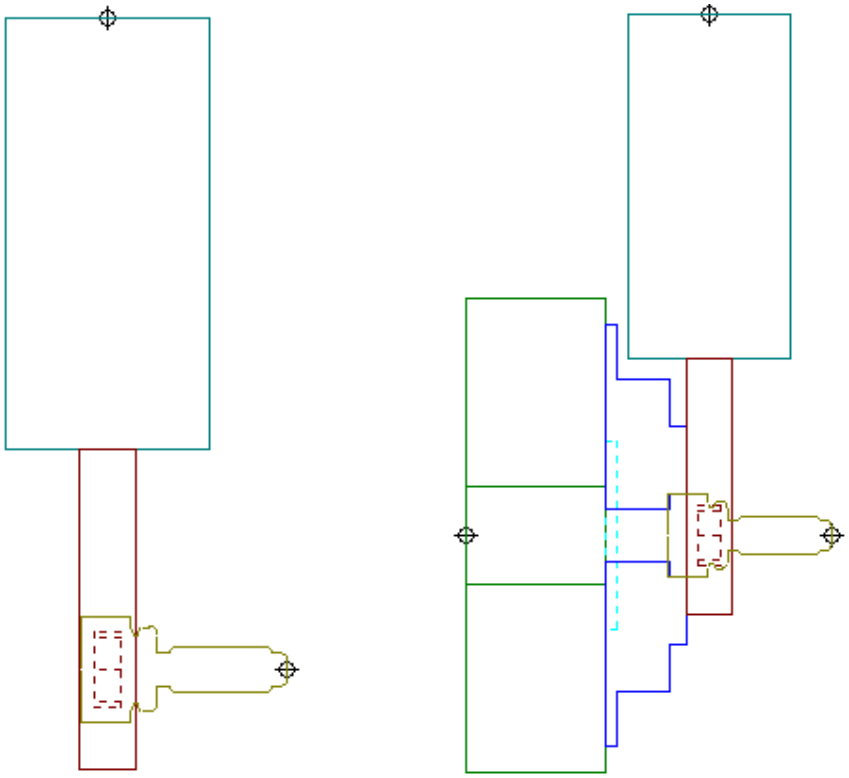
Ao acessar o sistema, o usuário recebe uma lista de peças, subdivida entre as que a escolha já foi efetuada, as que podem ser efetuadas e aquelas que ainda dependem de escolhas prioritárias. Durante o processo de escolha da forma de produção de uma peça, o usuário recebe informações gráficas e alfanuméricas que descrevem cada uma das possíveis alternativas de emprego do equipamento sob sua responsabilidade. Além de visualizar as informações relativas às opções existentes, o usuário tem acesso às alternativas já escolhidas pelos usuários prioritários, às opções disponíveis aos usuários menos privilegiados e às informações relativas às interações que ocorrem entre os equipamentos. Com isso, o usuário tem uma visão global da influência de sua escolha. A **Figura 5.4** mostra, como exemplo, algumas das informações gráficas recebidas pelo usuário responsável pela escolha da forma como um robô manipula a peça (a **Figura 5.4 a** apresenta apenas a garra do robô). A partir da escolha da fixação da peça com uma garra, ficam restringidas as possibilidades compatíveis de retirar a peça da placa do torno. A **Figura 5.4 b** mostra a transferência da peça entre a garra do robô e a placa do torno.

O sistema apresentado é utilizado para simular a fabricação de uma peça e não para controlar os equipamentos utilizados na sua produção. Portanto, após todos os usuários do Módulo de Escolha terem informado as suas opções, a peça é liberada para a programação da produção. Os responsáveis pela programação dos equipamentos utilizados na produção da peça têm acesso direto aos dados armazenados no sistema de persistência a partir de outros programas computacionais com interface apropriada.

5.2.3 Arquitetura Detalhada

A arquitetura apresentada na Figura 5.1 mostra somente alguns aspectos das aplicações desenvolvidas a partir do *framework*. As informações relativas à forma de interação dos usuários apresentadas na seção anterior possibilitam o desenvolvimento de um modelo arquitetônico mais evoluído. O resultado dessa evolução é apresentado na **Figura 5.5**. O usuário do Módulo de Configuração tem a sua disposição dados técnicos reais do sistema de fabricação. Ele estrutura as informações e as introduz no sistema de persistência que modela o

sistema de fabricação. O usuário do Módulo de Simulação só precisa escolher o sistema de fabricação e a peça a ser simulada. A diferença de dimensão dos usuários representa a qualificação que cada um deles deve possuir. Os usuários do Módulo de Escolha recebem informações gráficas e alfanuméricas que descrevem a utilização de cada equipamento. Eles realizam a escolha a ser utilizada de forma estruturada e as opções escolhidas são registradas no sistema de persistência que contém os resultados da simulação. As informações ali armazenadas ficam disponíveis para os sistemas de programação dos equipamentos, que geram programas para comandos numéricos (CN) e para controladores lógicos programáveis (CPL) para máquinas, robôs e sistemas de transporte. Os sistemas externos geram também as folhas de processo, as ordens de fabricação e as programações de alocação de máquinas para as aplicações computacionais de controle dos sistemas de fabricação, como aquelas propostas por Aarsten *et al*. (2000) e Schmid (2000a) (seção 3.6).



a) fixação com a garra b) transferência entre a placa e a garra

Figura 5.4 – Informações gráficas do Módulo de Escolha

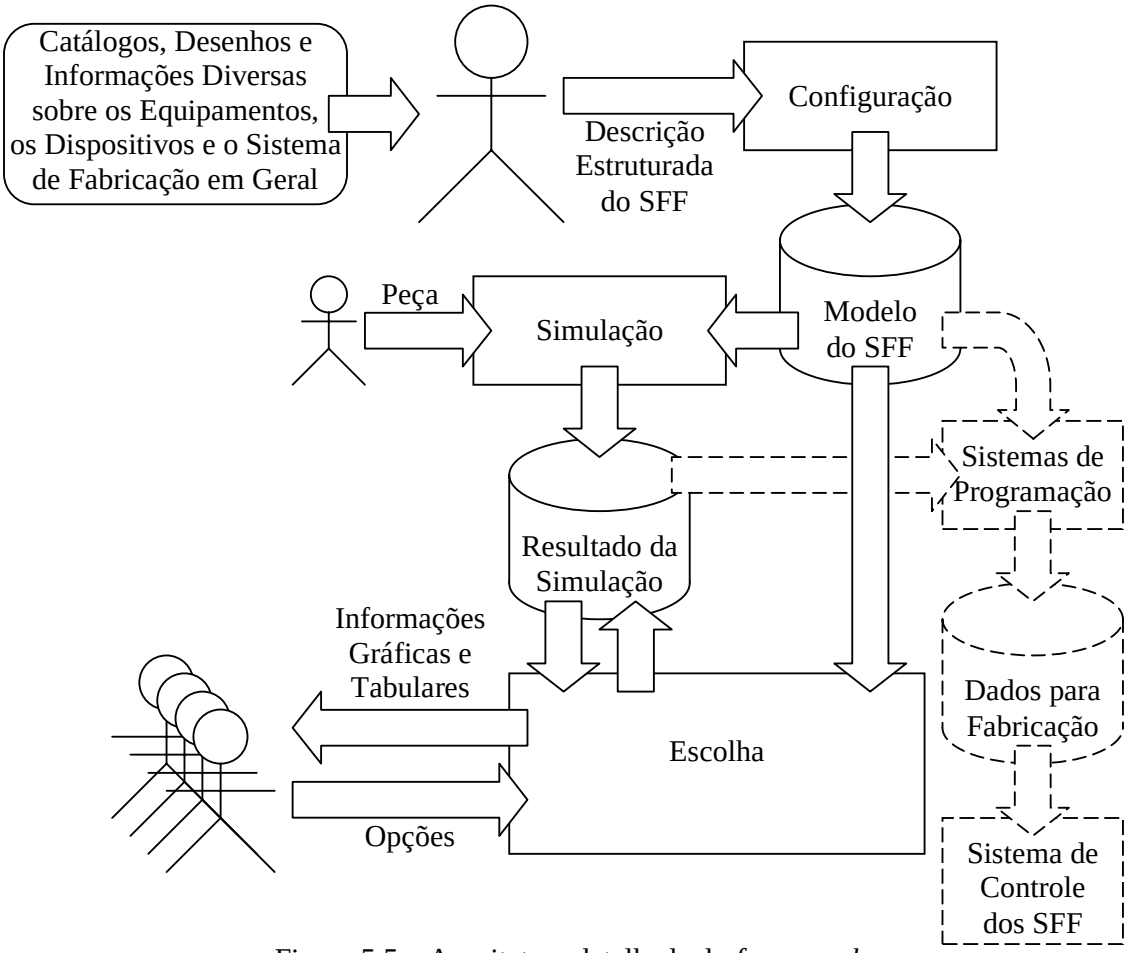


Figura 5.5 – Arquitetura detalhada do *framework*

5.3 Integração entre Módulos

Na seção anterior observa-se que os três módulos existentes não são executados simultaneamente. A quantidade de execuções de cada um dos módulos também não é coincidente. O Módulo de Configuração é executado uma única vez, o Módulo de Simulação uma vez para cada peça e o Módulo de Escolha uma vez para cada equipamento presente no fluxo de cada peça. Conseqüentemente, a troca de informações entre os módulos por envio de mensagens ou compartilhamento de memória fica inviabilizada. A troca de informações só pode ser realizada por meio de um sistema de persistência, ou seja, os módulos devem estar aptos a arquivar e recuperar informações.

A utilização de dados armazenados em sistemas de persistência é a forma mais antiga de integração entre módulos de uma aplicação. A opção mais comum é a utilização de arquivos. Entretanto, nas últimas décadas tem-se utilizado banco de dados sempre que a estrutura de dados torna-se mais complexa. Os bancos de dados oferecem uma interface padronizada de

acesso às informações, facilitando a integração de outras aplicações que compartilhem os mesmos dados. A utilização da linguagem SQL para acesso a banco de dados relacionais é a tendência mais atual em sistemas comerciais, apesar de perderem espaço para os banco de dados orientados a objetos nas aplicações acadêmicas. Para realizar a comunicação entre os módulos do *framework* desenvolvido no escopo deste trabalho optou-se pela utilização de um banco de dados relacional.

A necessidade de arquivamento e posterior recuperação de informações levou à criação de padrões de persistência que estão aptos a efetuar estas tarefas em um nível hierárquico abstrato, facilitando o seu reuso na implementação das entidades e dos relacionamentos do modelo de dados. Os *frameworks* utilizados para a criação de aplicações que realizam o arquivamento e a recuperação de informações são desenvolvidos com base nos padrões de persistência encontrados na literatura (Keller & Coldewey, 1998; Yoder *et al*., 1998). Esses padrões de persistência implementam toda a manipulação que se faz necessária em um banco de dados relacional, independentemente da aplicação em que eles serão utilizados.

Uma determinada entidade do modelo de dados é associada a uma classe em cada um dos módulos, que tem a responsabilidade da manipulação da informação correspondente no banco de dados. A classe associada a essa informação tem também responsabilidades distintas de acordo com a funcionalidade de cada um dos módulos. A fim de efetuar o reuso do projeto e do código associado à responsabilidade de manipulação do banco de dados, separa-se a classe associada a uma dada informação em duas. A primeira delas representa a interface entre a aplicação e o banco de dados e, conseqüentemente, é utilizada em todos os módulos. A segunda classe é específica de cada módulo, de acordo com a funcionalidade requerida, e tem acesso à classe da interface. Para o projeto do *framework* optou-se pela utilização do padrão de persistência proposto por Yoder *et al*. (1998). Nesse padrão existe uma classe abstrata denominada *PersistentObject* que estabelece a interface para manipulação do banco de dados. Todas as classes que manipulam tabelas são derivadas dessa classe. Por meio de outras classes pertencentes ao padrão, faz-se o mapeamento de objetos da aplicação para um sistema de gerenciamento de banco de dados relacional. Para realizar as operações de escrita, leitura, alteração e eliminação de informações, outras classes do padrão são responsáveis pela geração do código SQL, mapeamento dos atributos da classe para os atributos da tabela e vice-versa, conversão entre os tipos desses atributos, geração automática de identificação única, além de garantir a conexão com o sistema de gerenciamento do banco de dados e que as manipulações sejam realizadas de forma atômica. A **Figura 5.6** mostra o diagrama de classes resultante em

cada aplicação para a manipulação de uma tabela relacional. Uma classe da aplicação tem acesso a uma classe da interface, que através do padrão de persistência troca informações com o sistema de gerenciamento de banco de dados. A lógica da aplicação relacionada com as informações contidas na tabela relacional é implementada na classe da aplicação que difere em cada um dos módulos do *framework*. Na Figura 5.6 apresenta-se apenas a relação de uma classe da aplicação, que pode ser instanciada tantas vezes quanto necessário, com uma classe da interface, que é instanciada uma única vez, de acordo com o padrão de projeto *Singleton* (Gamma *et al.*, 1995). Em classes da aplicação mais complexas pode ser necessário que a persistência da informação a elas associada ocorra em mais de uma tabela relacional. Pode ocorrer também que distintas classes da aplicação utilizem a mesma classe da interface para a persistência da informação em uma única tabela relacional.

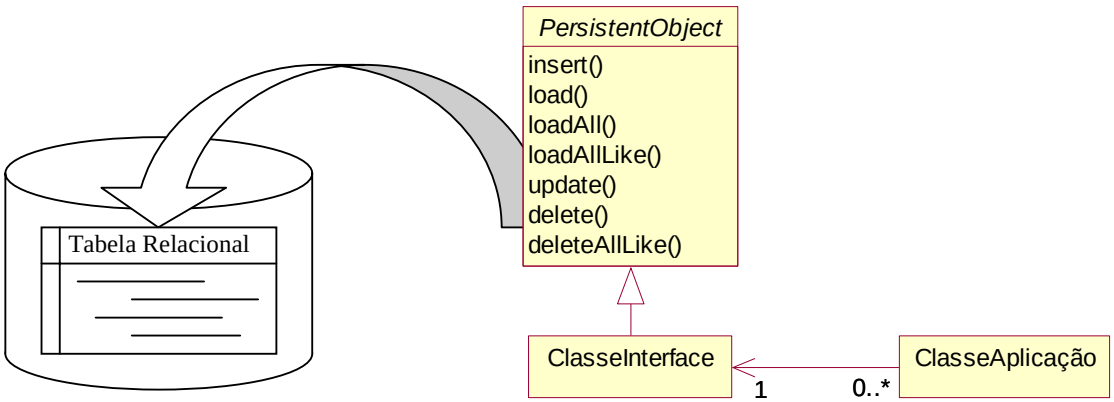


Figura 5.6 – Comunicação com o banco de dados

O padrão de projeto proposto por Yoder *et al*. (1998) delega às subclasses de *PersistentObject*, as classes desenvolvidas pelo usuário do padrão, a responsabilidade pelo tratamento dos atributos específicos de cada entidade. No escopo deste trabalho, este padrão foi estendido para que a maior parte dessa responsabilidade ficasse embutida nas classes do padrão. Para tal, o construtor de cada subclasse de *PersistentObject* informa às classes do padrão o nome da tabela relacional e o nome e o tipo de seus atributos. Com isso, o código SQL correspondente a cada manipulação pode ser totalmente gerado pelo padrão. A desvantagem desta opção é que para cada tabela relacional deve ser criada exatamente uma classe na interface.

5.4 Persistência de Pontos Variáveis

Em uma aplicação específica, o padrão de persistência original proposto por Yoder *et al*. (1988) ou a sua extensão realizada neste trabalho podem ser facilmente implementados,

pois todas as tabelas que fazem parte do modelo relacional são conhecidas. Caso as informações relativas a uma estrutura da aplicação composta por uma classe base e suas subclasses devam ser persistidas, devem ser criadas classes na interface e tabelas relacionais no banco de dados de acordo com a **Figura 5.7**.

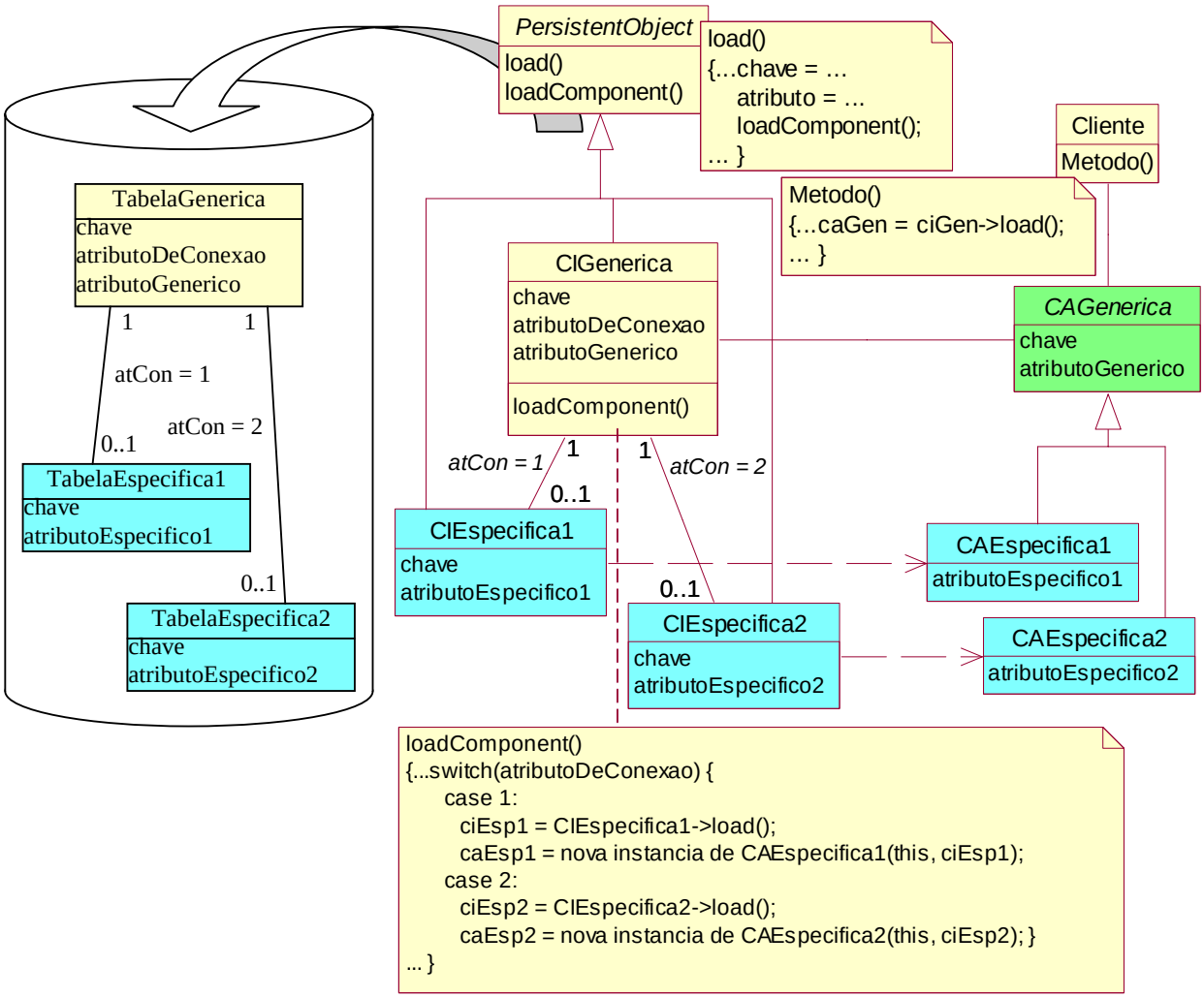


Figura 5.7 – Modelo básico de uma aplicação com generalização

A Figura 5.7 e as seguintes apresentam apenas duas subclasses da classe genérica da aplicação, assim como duas classes específicas na interface e duas tabelas relacionais específicas no banco de dados, mas o modelo deve ser abstraído para um número qualquer de subclasses, classes e tabelas específicas. A tabela genérica contém, além dos atributos genéricos, a chave, que identifica a informação, e o atributo de conexão, que relaciona a tabela genérica com uma das tabelas específicas. A tabela específica contém a chave e os atributos específicos. Esses mesmos atributos estão presentes nas classes da interface, em

função da extensão realizada no padrão, que implica em uma correspondência biunívoca entre as tabelas relacionais e as classes da interface. As classes específicas da interface não são subclasses da classe genérica, mas sim da classe *PersistentObject*. Caso o fossem, os atributos da classe genérica fariam parte das classes específicas e, pela opção de implementação realizada, as tabelas relacionais específicas também possuiriam estes atributos. A classe genérica da aplicação não possui o atributo de conexão e as classes específicas não possuem a chave, pois ela pode ser acessada a partir da classe genérica pelo mecanismo de herança.

Quando um cliente da aplicação necessita de informações do banco de dados, ele envia uma mensagem de leitura à classe genérica da interface e o resultado dessa operação é atribuído a uma referência à classe genérica da aplicação. O método de leitura da classe genérica da interface lê os atributos presentes na tabela relacional e aciona o método de leitura de seus componentes. O método de leitura dos componentes implementado na classe *PersistentObject* não realiza nenhuma operação, mas para classes que participam de uma abstração de generalização, ele é redefinido para efetuar a leitura das informações da tabela específica correta e criar uma instância da classe específica da aplicação correspondente por meio de um acesso direcionado em função do valor do atributo de conexão. A implementação do acesso direcionado, por exemplo, na linguagem C++ (Lippman e Lajoie, 1988), é realizado pelo comando "*switch* / *case*". Da mesma forma, as operações de escrita, alteração e eliminação de informações do banco de dados estão associadas à utilização do atributo de conexão.

O modelo apresentado na Figura 5.7 é suficiente para implementar uma aplicação específica que utiliza a persistência de informações. Neste caso todas as subclasses são previamente conhecidas e podem ser associadas a valores do atributo de conexão e a tabelas relacionais correspondentes. Entretanto, para implementar a flexibilidade de um ponto variável, no qual está aberta ao usuário a possibilidade de adicionar uma variabilidade não prevista, o modelo não atende aos requisitos de um *framework*. Em conjunto com a nova classe específica da aplicação que implementa a nova variabilidade, o usuário do *framework* deve criar uma classe específica da interface e uma tabela relacional específica. O acesso à nova classe específica da interface e à tabela relacional específica correspondente é realizado por meio do atributo de conexão que deve assumir um novo valor específico. O código do método de leitura dos componentes deve ser estendido para o novo valor do atributo de conexão. As classes da aplicação e da interface e a tabela relacional genéricas fazem parte do *framework* e as classes e as tabelas relacionais específicas são partes integrantes de uma

determinada aplicação. Para evidenciar essa divisão as classes e as tabelas relacionais são apresentadas na figura em cores distintas. Em azul mostram-se as tabelas relacionais e as classes, que devem ser implementadas pelo usuário do *framework* para o desenvolvimento de uma aplicação determinada. As classes e as tabelas relacionais do *framework* são apresentadas em amarelo, com exceção da classe abstrata CAGenerica que está desenhada em verde, pois ela representa a interface do *framework*, a partir da qual o usuário deve implementar subclasses. O mesmo padrão de cores é utilizado nas demais figuras deste capítulo.

A implementação de uma nova variabilidade está associada à alteração do método de leitura dos componentes que faz parte do código base do *framework*. Uma alteração semelhante é necessária para efetuar as demais manipulações nas tabelas relacionais. A fim de evitar a alteração do código base do *framework* quando se implementam novas classes na aplicação e na interface e a tabela relacional correspondente para introduzir uma nova variabilidade do ponto flexível, realizam-se três transformações no diagrama de classes, baseadas na generalização sistemática proposta por Schmid (1997; 1999).

5.4.1 Generalização de Tratamento da Chave

Todas as tabelas relacionais específicas possuem em comum a chave para identificar a informação. Outro atributo comum entre as tabelas específicas não deve existir, pois nesse caso o atributo faria parte da tabela genérica. Essa distribuição dos atributos também está presente nas classes da interface. Com o objetivo de efetuar uma generalização devido à presença comum da chave, cria-se uma classe base abstrata na interface, denominada "CIEspecificaAbstrata", também derivada da classe *PersistentObject*, na qual são implementadas a chave e os métodos relativos a ela. As classes específicas da interface passam a ser subclasses dessa nova classe base, segundo o padrão Herança de Interface (Figura 2.1) (Schmid, 1997; 1999). O diagrama de classes resultante na interface entre a aplicação e o banco de dados pode ser visto na **Figura 5.8**. Com isso, o grau de reutilização de um *framework* que implementa um ponto variável dessa forma é maior, pois, por um lado, todo o tratamento da chave não precisa ser recodificado e, por outro, as classes específicas são subclasses de uma mesma classe abstrata e, portanto, são vistas da mesma forma pela classe genérica. Entretanto, a reutilização do *framework* ainda é insatisfatória, pois a implementação de uma nova variabilidade, associada a um novo valor do atributo de conexão, continua implicando em uma grande alteração do código base do *framework* em razão do acesso

direcionado por meio desse atributo. As partes relativas à aplicação e ao banco de dados não são alteradas por essa transformação.

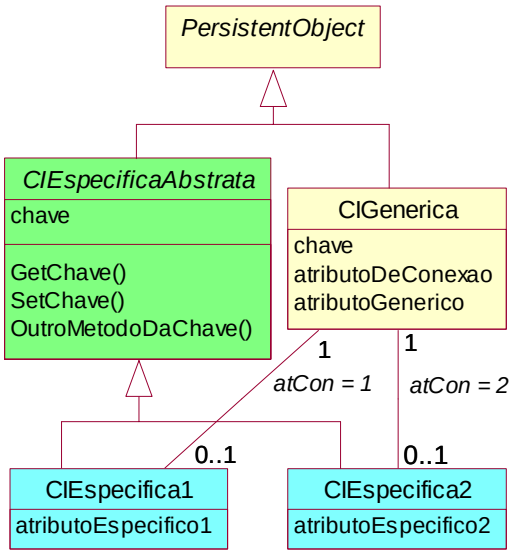


Figura 5.8 – Generalização da chave

5.4.2 Implementação Flexível de Classes Específicas

Para permitir a implementação de novas variabilidades no ponto flexível, sem a necessidade de alteração do código base do *framework*, efetua-se a segunda transformação em seu diagrama de classes e no modelo relacional do banco de dados. Utiliza-se o padrão Cadeia de Responsabilidade (Figura 2.2) (Gamma *et al.*, 1995) que implementa um atributo denominado sucessor, responsável pela formação de uma cadeia de objetos candidatos à solução de um problema qualquer. No caso, a informação sobre qual tabela específica contém os detalhes da informação presente na tabela genérica deixa de ser uma informação conhecida, associada ao atributo de conexão, para que o código base do *framework* não precise ser alterado a cada introdução de uma nova variabilidade. A informação disponível passa a ser a cadeia de tabelas relacionais candidatas a conter a informação. Portanto, o atributo de conexão deixa de ser necessário para efetuar o relacionamento entre a classe genérica e a classe específica correspondente. O método de leitura da classe genérica da interface lê seus próprios atributos e envia uma mensagem de leitura do componente ao início da cadeia de classes específicas da interface. Cada classe específica da interface procura efetuar a leitura no banco de dados. Em caso de sucesso, a classe da aplicação correspondente é instanciada, em caso contrário, envia-se uma mensagem de leitura à classe sucessora da

cadeia. A **Figura 5.9** mostra o diagrama de classes correspondente à interface entre o banco de dados e a aplicação.

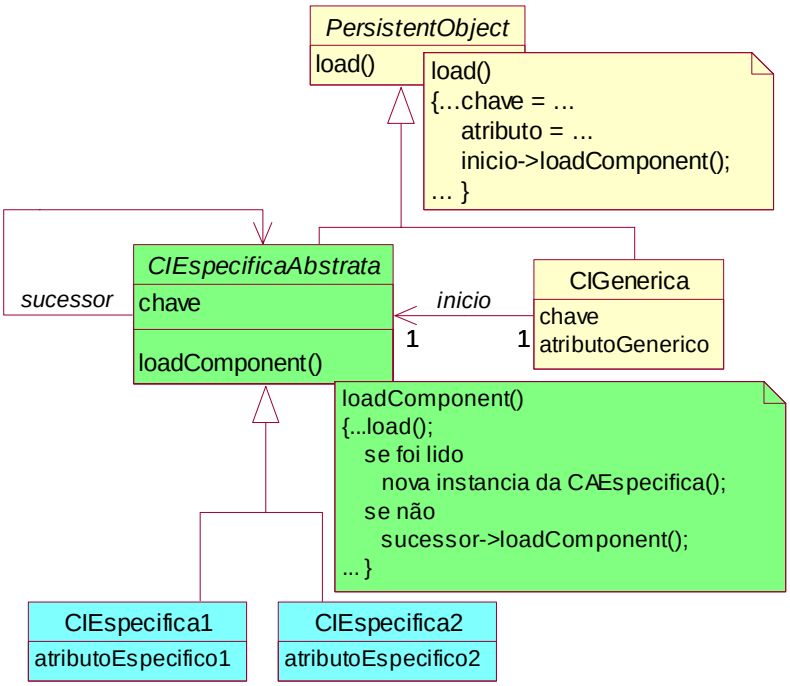


Figura 5.9 – Implementação de classes específicas

5.4.3 Criação Flexível de Objetos da Aplicação

Com a estrutura de classes da Figura 5.9 é possível definir novas variabilidades no ponto flexível, ou seja, novas tabelas relacionais no banco de dados e as subclasses correspondentes no *framework*, na aplicação e na interface, pois o relacionamento com elas será da mesma forma que com as classes previamente implementadas. Durante a inserção de informações em um banco de dados, o objeto da aplicação que deve ser persistido tem conhecimento da tabela específica apta a armazenar a informação. Entretanto, quando se efetua a recuperação de uma informação estruturada segundo a abstração de generalização, normalmente não se conhece a tabela específica que contém a informação complementar àquela da tabela genérica. Conseqüentemente, também não é conhecido o objeto correspondente da aplicação que deve ser criado. Durante a recuperação de informações do banco de dados, um objeto cliente envia uma mensagem de leitura ao objeto da interface, correspondente à tabela genérica. A leitura no banco de dados deve ocorrer tanto na tabela genérica como na tabela específica correspondente. Só no instante em que se percorrem os objetos da cadeia de responsabilidade é que se encontra a classe apta a efetuar a leitura na

tabela específica correta. Conseqüentemente, só então estará disponível a informação sobre qual classe específica da aplicação deve ser instanciada.

Para permitir a criação flexível de objetos da aplicação deve-se adaptar o *framework* para que o modo como as classes são instanciadas seja independente de qual objeto deva ser criado. A transformação do diagrama de classes da interface entre o banco de dados e a aplicação pode ser vista na **Figura 5.10**. Utiliza-se o padrão Método de Fabricação (Figura 2.3) (Gamma *et al*., 1995) que define uma interface para a instanciação de classes, mas delega às suas subclasses a responsabilidade de decidir qual objeto deve ser criado. O método de leitura dos componentes da classe abstrata da interface, usado por suas subclasses, utiliza o método abstrato de fabricação que é redefinido nas subclasses. Cada método de fabricação das subclasses da interface cria uma instância da classe correspondente da aplicação.

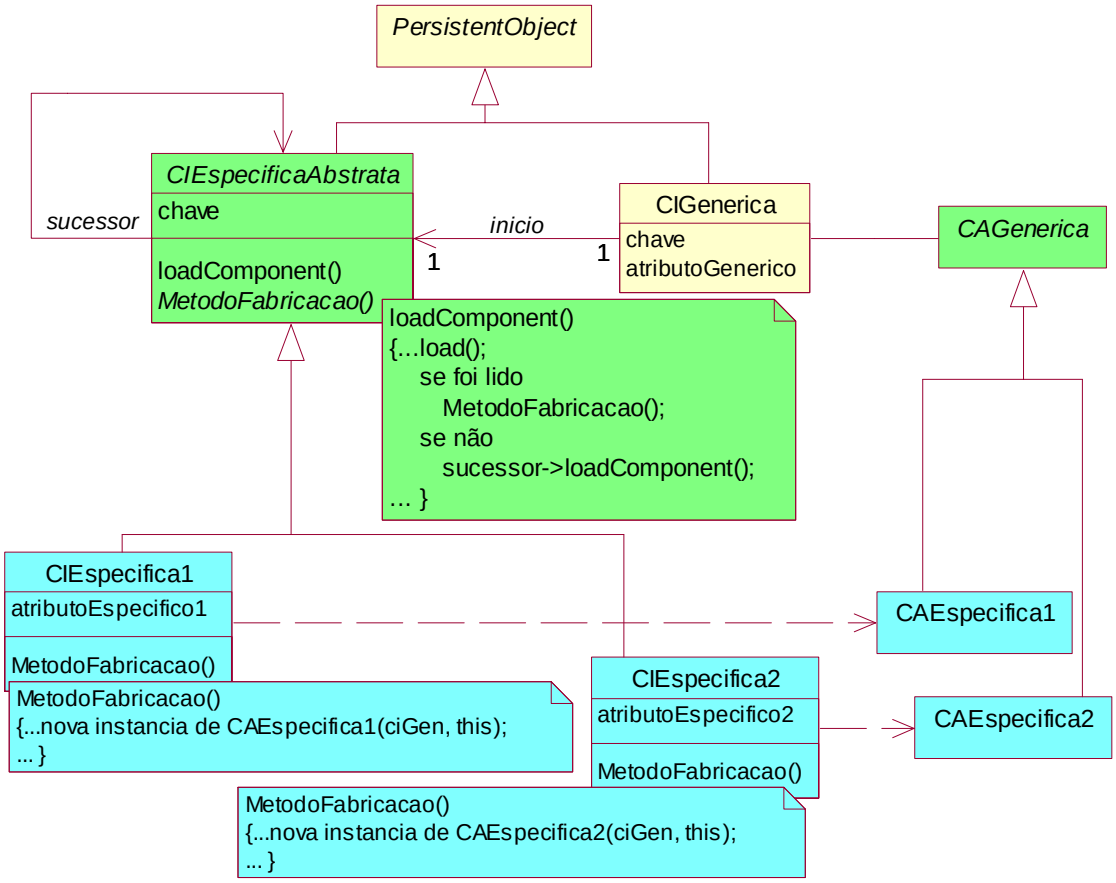


Figura 5.10 – Criação flexível de objetos da aplicação

Com as três transformações apresentadas acima é possível implementar uma nova variabilidade em um ponto flexível do *framework* que faça parte da integração entre os módulos, sem que o seu código base precise ser alterado. É necessário apenas implementar as classes específicas na interface e nos três módulos como subclasses da interface do *framework*

e o método de fabricação na classe da interface, para criar um objeto da classe correspondente da aplicação.

5.5 Projeto do *Framework*

A seção 2.5.3 descreve algumas das formas de desenvolvimento de um *framework*. Os especialistas dividem as formas de desenvolvimento de *frameworks* em dois grandes grupos. O desenvolvimento segundo o primeiro grupo é realizado após uma profunda análise do domínio para identificação de seus pontos variáveis. Essa análise deve ser realizada em conjunto pelo especialista do domínio, que conhece as características desse domínio, e pelo engenheiro de software que organiza as informações de acordo com os requisitos para o projeto do *framework*. O desenvolvimento de acordo com o segundo grupo é fundamentado na generalização de aplicações semelhantes no mesmo domínio. Roberts e Johnson (1998) sugerem que três aplicações concretas sejam inicialmente desenvolvidas, a fim de garantir o conhecimento do domínio necessário para a generalização das aplicações.

As duas técnicas estão presentes no projeto do *framework* desenvolvido no escopo deste trabalho. O autor já desenvolveu aplicações específicas para solução de diversos problemas típicos do domínio da manufatura e tem formação acadêmica na área, o que o qualifica como especialista do domínio neste desenvolvimento. Alguns aspectos do *framework* são generalizações de aplicações semelhantes e outros somente da análise do domínio. A integração destas técnicas é realizada com base na generalização sistemática proposta por Schmid (1997; 1999). O modelo para simular o fluxo de peças proposto no capítulo anterior deve ser visto como o modelo de uma aplicação específica no domínio e não como um *framework*. Em vários aspectos não existe a flexibilidade necessária para caracterizar um *framework*. Nas seções seguintes descrevem-se as transformações realizadas no modelo anterior para que o sistema computacional seja aplicável ao maior espectro possível de fábricas. A generalização das aplicações e a análise do domínio indicam que é possível desenvolver-se um *framework* de uso abrangente e que, para tal, deve-se prever variabilidades nos seguintes pontos:

- na seqüência de equipamentos que compõem o fluxo principal,
- na quantidade, na variedade e no tipo de equipamento de fabricação,
- na quantidade, na variedade e no tipo de equipamento de manipulação,
- na quantidade, na variedade e no tipo de equipamento de transporte,

- na quantidade, na variedade e no tipo de equipamento de teste,
- na quantidade, na variedade e no tipo de equipamento de armazenagem,
- na utilização de um mesmo equipamento para realizar duas ou mais funções do fluxo da peça,
- na associação entre os equipamentos de movimentação (manipulação ou transporte) e os demais equipamentos para formar conexões e realizar etapas do fluxo da peça,
- na quantidade, na variedade e no tipo de placas de fixação,
- na quantidade, na variedade e no tipo de paletes de transporte,
- na quantidade, na variedade, no tipo e na associação de longarinas para montagem nos paletes de transporte,
- na quantidade, na variedade e no tipo de garras de robôs, e
- na quantidade, na variedade e no tipo de pinças para montagem nas garras de pinças.

A seguir descrevem-se as alterações necessárias no modelo para que as variabilidades acima listadas possam ser atingidas.

5.5.1 Aspectos Gerais do Projeto

No modelo apresentado no capítulo anterior o relacionamento entre várias classes é caracterizado por uma agregação. O relacionamento direto entre classes agregadora e agregada concretas impossibilita que tipos diferentes da classe agregada façam parte do mesmo relacionamento. Logo, a classe agregadora não deve relacionar-se diretamente com a agregada. A estrutura de classes deve ser alterada para que o relacionamento da classe agregadora ocorra com uma classe base abstrata que define uma interface comum a todos os possíveis tipos da classe agregada.

Os dispositivos descritos na seção 4.2.3 são aqueles que ocorrem com maior freqüência em sistemas de fabricação. Entretanto, as descrições utilizadas não estão aptas a representar todos as possíveis formas construtivas de dispositivos. A flexibilização deste aspecto é obtida ao se introduzir uma classe base abstrata para criar uma interface comum de acesso a cada um dos tipos de dispositivos. Além das classes referentes aos dispositivos, outras também participam de relacionamentos M::N. Por meio de uma análise simples do domínio conclui-se que os dispositivos são montados nos equipamentos e não o inverso. Em outros relacionamentos a identificação da classe que pode ser colocada em um nível hierárquico

superior, como os equipamentos no caso anterior, pode não ser tão simples. O método de simulação ocorre sempre no sentido "*top-down*", ou seja, a simulação de um equipamento é realizada por meio da simulação de seus dispositivos. Com isso, o relacionamento da classe referente ao nível hierárquico superior deve ser também com uma classe base abstrata que define uma interface comum para todos as subclasses que representam os diversos tipos da classe do nível hierárquico inferior.

Além do detalhamento dos relacionamentos de agregação e dos relacionamentos tipo M::N, nas seções seguintes apresentam-se também as opções de projeto realizadas para determinação e simulação das rotas deste *framework*.

5.5.2 Transformações dos Relacionamentos de Agregação

Tomando-se o exemplo da Figura 4.1, em que o sistema é dado por um conjunto de células e essas por um conjunto de equipamentos, altera-se o relacionamento de agregação entre elas de acordo com o diagrama de classes da **Figura 5.11**. Esta transformação permite que cada tipo de célula tenha um comportamento diferente e que novos tipos de células sejam criados pelo usuário do *framework*, de acordo com a necessidade específica da aplicação.

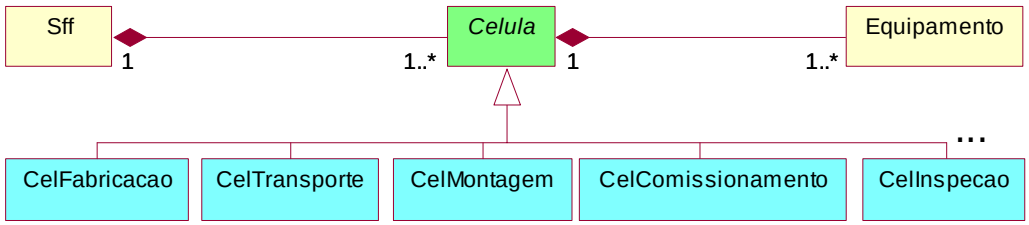


Figura 5.11 – Generalização dos tipos de células

Esta transformação não é aplicada a todos os relacionamentos de agregação. Como cada tipo de equipamento não é descrito diretamente, mas sim por meio da relação com outras classes (Figura 4.2), a classe equipamento não sofre a mesma transformação. As classes para descrição dos detalhes dos equipamentos também são transformadas do modo apresentado. As classes relativas às funções do fluxo de material (nível intermediário da Figura 4.2) e as classes relativas às funções técnicas de cada tipo de equipamento (nível inferior da figura) tornam-se subclasses de classes base abstratas que definem uma interface comum para acesso a partir da classe agregadora. Também a relação entre as classes garra e pinça (Figura 4.9) e base de palete e palete (Figura 4.13) são transformadas da mesma forma. Os relacionamentos entre as classes rota, seção e conexão apresentados na Figura 4.20 já são realizados com a

utilização de classes abstratas para definição de interface comum e, portanto, não precisam ser assim transformados.

5.5.3 Transformações para Generalização dos Dispositivos

O diagrama de classes relativo à placa de três castanhas mostrado na Figura 4.6 é transformado para que a classe torno tenha um relacionamento com uma classe base abstrata que define uma interface comum a todos os tipos de placa. A **Figura 5.12** apresenta como as classes relativas a outros tipos de placas, como a placa de quatro castanhas ou a placa de arraste, entre outras, são alocadas no diagrama de classes. Simultaneamente, o relacionamento de agregação entre o sistema flexível de fabricação (Sff) e a placa de 3 castanhas é substituído pelo relacionamento com a classe base abstrata, de acordo com os requisitos de flexibilidade expostos na seção anterior.

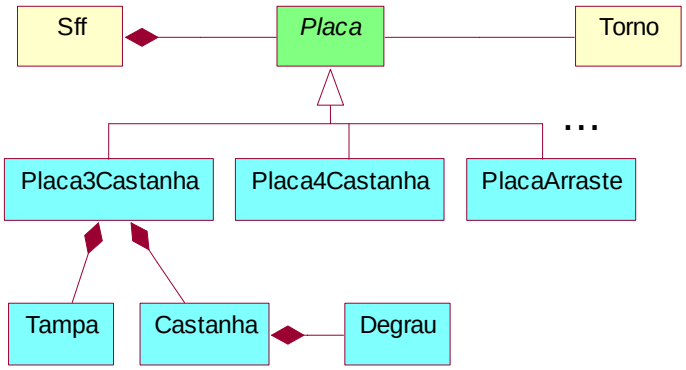


Figura 5.12 – Generalização dos tipos de placas de fixação

A transformação do diagrama de classes relativo à garra (Figura 4.9) é mais complexa. Inicialmente, cria-se a classe base abstrata denominada garra, a partir da qual todos os tipos são derivados. Além da garra de pinças, a **Figura 5.13** mostra também a classe relativa a garras magnéticas, um exemplo de outro tipo de garra. A classe relativa à garra de pinças também é abstrata, pois existem garras com duas ou com três pinças que devem ser tratadas distintamente. As garras de pinças possuem pinças de diversos tipos. Por isso, o relacionamento da classe garra de pinça é com a classe base abstrata pinça, da qual deriva-se a classe pinça para fixação perpendicular da peça com assentos prismáticos (representada na Figura 4.28). Existem vários outros tipos de garras e de pinças que não estão representados no diagramas de classes. Na figura observa-se que a garra magnética não é composta por pinças, ou seja, a existência da pinça não é uma flexibilidade exigida pelo *framework*, mas sim por uma aplicação específica que utiliza garra de pinças. Entretanto, como a utilização de garras

de pinças para manipulação de peças na indústria metal-mecânica é comum, implementou-se também no *framework* as interfaces relativas à garra de pinça e à pinça.

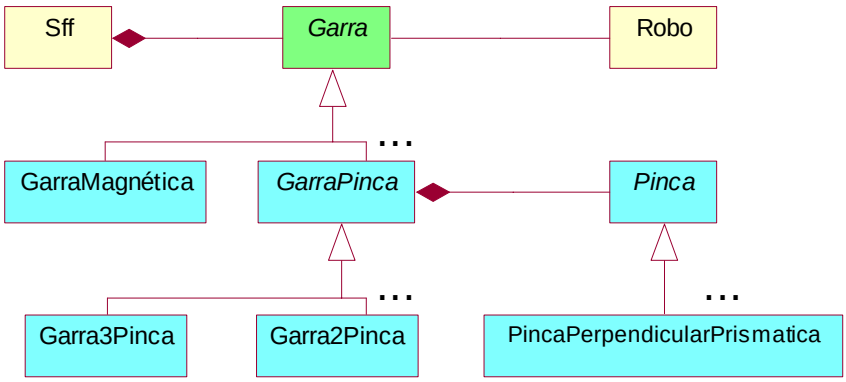


Figura 5.13 – Generalização dos tipos de garras e pinças

O diagrama de classes referente ao palete de transporte (Figura 4.13) também deve sofrer uma transformação para atender aos requisitos de um *framework*. A classe palete torna-se abstrata para definir uma interface comum a todos os tipos de paletes. A **Figura 5.14** mostra que, além do palete de longarinas, outros tipos de paletes podem ser definidos como subclasses da classe base palete. A classe palete de longarina relaciona-se com a classe base abstrata longarina, que define uma interface comum para todos os tipos de longarinas. Na figura são apresentados apenas dois tipos de longarinas com assento prismático para apoio de peças na horizontal e na vertical (mostradas nas Figuras 4.11 e 4.12). A linha tracejada de divisão mostra que as classes relativas às longarinas fazem parte de uma aplicação específica e não do *framework*, pois outros tipos de paletes não as utilizam.

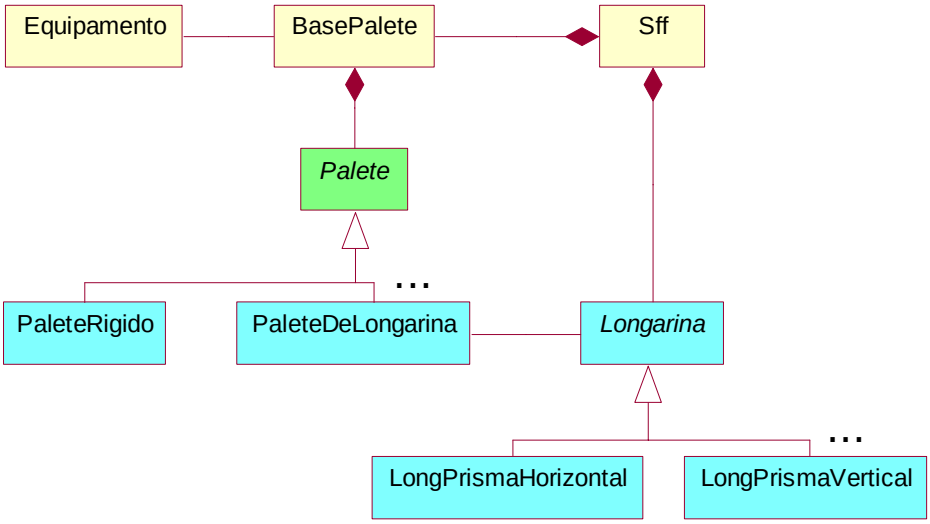


Figura 5.14 – Generalização dos tipos de paletes e longarinas

Nas três figuras anteriores estão também representados os relacionamentos entre cada um dos dispositivos e os equipamentos onde eles são utilizados. Assim, o relacionamento entre o torno e a placa e o relacionamento entre o robô e a garra são realizados com as respectivas classes base abstratas. O relacionamento entre um equipamento qualquer e a base do palete é mantido como anteriormente, pois se considera que existe apenas um tipo de base de palete e que a variabilidade é dada pela possibilidade de definição de tipos de paletes.

Os diagramas de classes mostrados anteriormente estão presentes nos três módulos. Os métodos associados a cada uma das classes são distintos nos módulos de acordo com o objetivo de cada um deles. No Módulo de Configuração, as classes representam janelas de diálogos para manipulação das informações nas respectivas tabelas relacionais do banco de dados. No Módulo de Simulação, os métodos das classes são responsáveis por simular a utilização dos dispositivos atuando sobre a peça. No Módulo de Escolha, a cada classe está associado um método para realizar a representação gráfica da atuação do dispositivo sobre a peça e para auxiliar o processo de escolha da forma de atuação a ser utilizada na fabricação real da peça. Nos três módulos existe uma estrutura paralela entre as classes da aplicação e as classes de interface com o banco de dados, semelhante à Figura 5.10. A solução adotada, dada pela associação dos padrões herança de interface, cadeia de responsabilidade e método de fabricação, é utilizada para realizar a interface entre o banco de dados e as estruturas de classes referentes à placa, à garra, à pinça, ao palete e à longarina, como mostrado na seção 5.4, e também entre o banco de dados e as estruturas de classes referentes ao material em bruto e à peça em bruto.

5.5.4 Transformações para Generalização das Rotas, Seções e Conexões

Na seção 4.3.1 apresentou-se o procedimento para determinação das possíveis rotas, pelas quais pode ocorrer o fluxo da peça. O procedimento mostrado considera a simulação da fabricação de uma peça com simetria rotacional. Conseqüentemente, os equipamentos de fabricação presentes no fluxo principal restringem-se a uma serra e a um torno. Na produção de peças com outra geometria utilizam-se equipamentos de fabricação específicos, que não estão presentes no fluxo principal apresentado. Portanto, é necessário transformar-se a forma como as conexões são identificadas e agrupadas para formação de seções e rotas.

A **Figura 5.15** mostra a solução adotada para que uma aplicação tenha o fluxo principal definido por outros equipamentos e tenha conexões distintas daquelas mostradas no capítulo anterior, sem que seja necessário a alteração do código base do *framework*. O procedimento

de simulação inicia pela determinação do fluxo principal. Para peças com simetria rotacional foi mostrada a seqüência de equipamentos pelos quais a peça deve fluir durante a sua produção. Para outros tipos de peças os equipamentos presentes no fluxo principal são outros. O fluxo é determinado pelo envio de uma mensagem a subclasses de uma classe base abstrata denominada processo de fabricação. No caso de peças rotacionais, a mensagem é enviada à subclasse torneamento, que determina o fluxo principal como foi mostrado na Figura 4.17. Para outros tipos de peças, a subclasse que responde a mensagem é outra e, portanto, o fluxo principal é em princípio distinto. A estrutura de classe que implementa o procedimento de escolha do fluxo principal é realizada com base no padrão de projeto Estratégia (Gamma *et al.*, 1995).

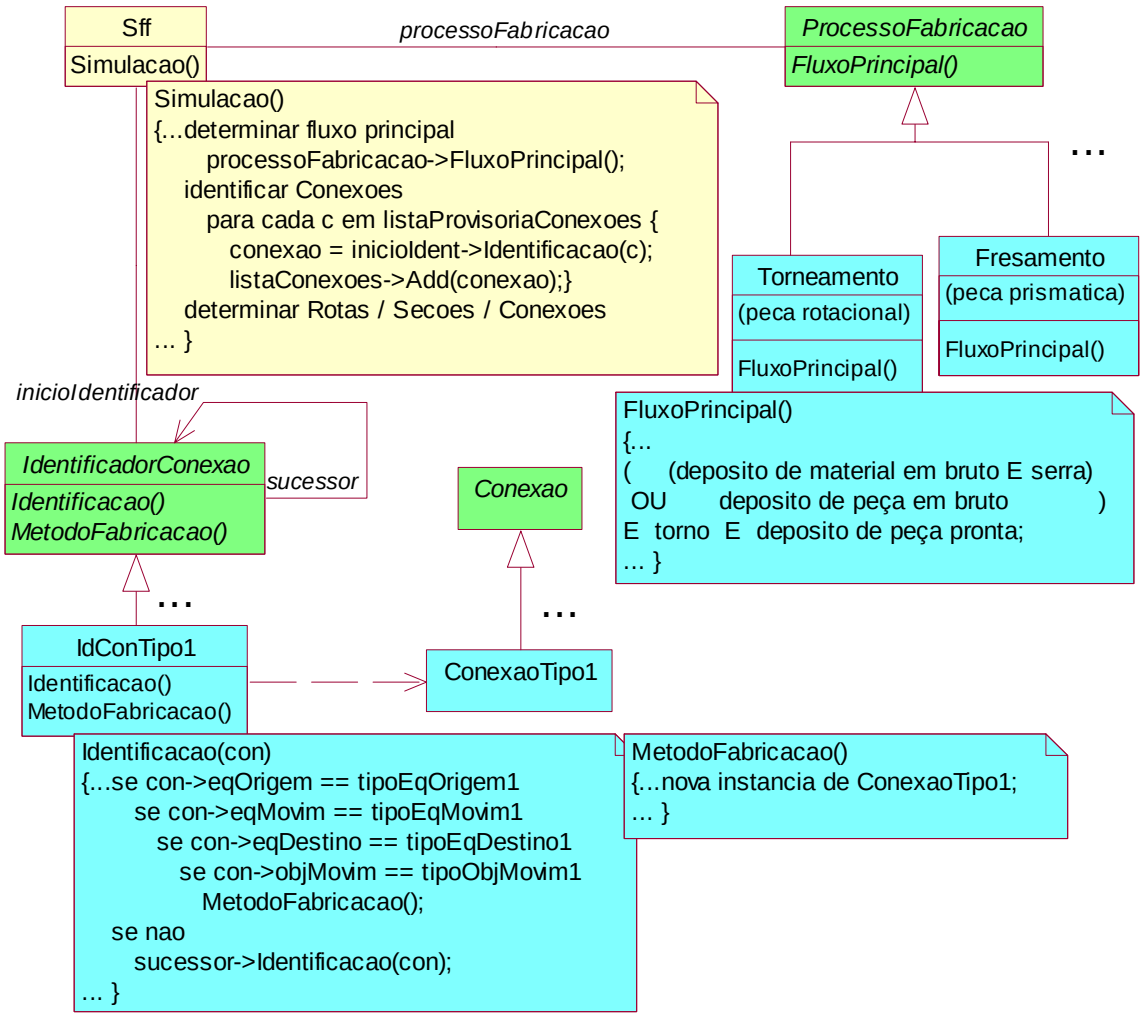


Figura 5.15 – Generalização da identificação das rotas

O passo seguinte consiste em identificar cada uma das conexões presentes no sistema de fabricação. As conexões armazenadas no banco de dados são constituídas apenas de chaves estrangeiras para os equipamentos de origem, movimentação e destino. Conseqüentemente, não existe distinção de tipos entre elas. A identificação de um tipo de conexão é realizada pela comparação entre os tipos de seus atributos e os tipos de atributos que determinam cada tipo de conexão. São comparados os equipamentos de origem, de movimentação e de destino e o objeto movimentado. Os diversos tipos de conexões possíveis são implementados em subclasse da classe base abstrata denominada identificador de conexão segundo os padrões Herança de Interface e Cadeia de Responsabilidade. Os elementos de uma conexão presente no sistema de fabricação são comparados com os elementos de cada um dos identificadores da cadeia até que ocorra a sua identificação. Após a identificação, a subclasse do tipo de conexão correspondente é instanciada de acordo com o padrão Método de Fabricação. A associação de padrões utilizada para a instanciação flexível da subclasse correta da classe abstrata conexão é a mesma que foi utilizada para a persistência de pontos variáveis do *framework* (seção 5.4).

O último passo consiste em montar a estrutura em árvore dos relacionamentos entre as rotas, seções e conexões. Essa estrutura é baseada na linguagem de padrões de sistemas de transporte (pontos e rotas) proposta por Zhao e Foster (1998). A montagem da estrutura é realizada sempre da mesma forma, pois basta comparar os equipamentos das conexões existentes no sistema de fabricação com aqueles do fluxo principal. A simulação da fabricação da peça é realizada pela simulação de cada uma das conexões presentes nas rotas. Como cada conexão foi individualmente identificada e um objeto da respectiva classe foi criado, o método de simulação de cada conexão é implementado de acordo com os equipamentos presentes em cada conexão na classe correspondente. Logo, o procedimento básico de simulação mostrado na Figura 4.21 e suas variações mostradas nas seções seguintes são utilizáveis para peças de qualquer geometria, pois são implementados sob medida.

5.6 Comparação entre este *Framework* e aqueles para controle de SFF

Nesta seção compara-se o *framework* proposto no escopo deste trabalho com os dois *frameworks* apresentados na seção 3.6. Os dois *frameworks* mostrados anteriormente foram desenvolvidos para o controle de sistemas flexíveis de fabricação, enquanto que o *framework* desenvolvido neste trabalho, efetua simulações do fluxo da peça por um destes sistemas com o objetivo de determinar a seqüência de operações pela qual ela deve passar durante a sua produção. Tanto o *framework* proposto por Schmid (2000a) como este são específicos para

subdomínios da manufatura, enquanto que a linguagem de padrão proposta pelo grupo de Turim (Aarsten *et al*., 1995) já foi utilizada para desenvolver *frameworks* em outros domínios.

Schmid apresenta o seu *framework* em conjunto com uma célula de fabricação real para peças rotacionais. O grupo de Turim não menciona em suas publicações o tipo de peça que pode ser produzido no sistema de fabricação utilizado para testar o *framework*. Entretanto, a instanciação de um *framework* para controle de sistemas de fabricação que produzam outros tipos de peças é muito semelhante. Em contrapartida, a instanciação deste *framework* para simular sistemas de fabricação que produzam outros tipos de peças é significativamente mais trabalhosa.

Os modelos básicos de classes dos dois *framework* encontrados na literatura são semelhantes e são detalhados até o nível de equipamento, enquanto que o proposto nesta tese modela também os dispositivos utilizados pelos equipamentos.

Como os *frameworks* apresentados são para controle de sistemas reativos reais, eles implementam um mecanismo de difusão para que um elemento do nível hierárquico inferior comunique a outro do nível superior o término de uma atividade, para que esse possa requisitar a atividade seguinte. Neste *framework* de simulação não é necessário o mecanismo de difusão, pois após o término de uma requisição a execução da aplicação retorna automaticamente após o ponto em que ela foi efetuada.

Quando os dois *frameworks* da literatura são instanciados, cada um deles gera uma única aplacação, enquanto que este *framework* gera três aplicações relativas aos módulos de que ele é composto. Conseqüentemente, a persistência das informações para a integração dos módulos é importante neste *framework*. A seção 5.4 deste trabalho é dedicada a apresentar o problema relacionado com essa opção e a solução adotada para a persistência de pontos variáveis de *frameworks*. Os dois *frameworks* da literatura recebem os dados de entrada relativos à produção a ser executada de sistemas de persistência implementados de forma convencional, tanto que seus autores não tecem nenhum comentário a este respeito.

A linguagem de padrão G++ foi estendida para incorporar a possibilidade de controle simultâneo de várias unidades fabris que interagem na produção de um conjunto mecânico qualquer. Tanto o *framework* de Schmid como o proposto nesta tese são previstos, respectivamente, para o controle e a simulação de apenas um sistema de fabricação.

5.7 Instanciação do *Framework*

Para simular o fluxo de uma peça em um sistema de fabricação real deve-se instanciar o *framework* para criar uma aplicação que simule as características individuais desse sistema. Como o *framework* proposto neste trabalho é modular, na realidade cria-se uma aplicação referente a cada um de seus módulos. O *framework* é do tipo caixa branca e, por isso, a sua instanciação está associada à criação de subclasses em suas interfaces que representam os equipamentos, a interação entre eles, os dispositivos e os demais elementos que compõem o sistema flexível de fabricação. Vários exemplos das classes que devem ser implementadas foram mostrados na seção anterior. Para testar o *framework* é necessário implementar pelo menos uma variabilidade em cada um de seus pontos flexíveis, pois eles representam os elementos pelos quais a peça passa durante o fluxo. Existem casos em que a peça não é processada em alguns dos elementos previstos e, portanto, se não for implementada uma variabilidade nesses pontos a aplicação gerada pelo *framework* simula normalmente o fluxo da peça.

A seguir apresenta-se sucintamente como é realizada a primeira utilização do *framework* para instanciação de uma aplicação, como utilizá-lo para instanciar aplicações semelhantes a outras já implementadas e para aplicações substancialmente diferentes daquelas existentes.

5.7.1 Instanciação de uma aplicação a partir do *framework*

A primeira aplicação desenvolvida a partir do *framework*, assim como a que foi realizada no escopo deste trabalho, não possui nenhum desenvolvimento anterior que possa ser entendido para que esforço necessário seja diminuído. Deve-se implementar subclasses em todas as interfaces do *framework* para gerar a aplicação. Inicialmente, implementa-se uma subclasse da classe abstrata Processo de Fabricação, como a classe Torneamento mostrada na Figura 5.15. Na subclasse implementa-se o método Fluxo Principal que ao ser acionado apresenta como resposta as possíveis seqüências de equipamentos que caracterizam o fluxo principal pelos quais a peça deve fluir para ser produzida. Na aplicação aqui desenvolvida a resposta do método Fluxo Principal da classe Torneamento é dada pelos equipamentos apresentados na Figura 4.17 (depósito de material em bruto, serra, torno e depósito de peça pronta ou depósito de peça em bruto, torno e depósito de peça pronta) de acordo com a sintaxe mostrada na Figura 5.15.

Em uma segunda fase deve-se implementar as conexões presentes no sistema de fabricação e os seus respectivos identificadores. Para cada equipamento de movimentação (manipulação ou transporte) presente no sistema de fabricação relacionam-se todos os possíveis pares de equipamentos (um no início e outro no fim) entre os quais pode ser realizada uma operação de movimentação. Para cada operação relacionam-se também os possíveis objetos que podem ser movimentados. Para cada conjunto com diferentes tipos de equipamento de origem, de movimentação e de destino e diferentes tipos de objeto movimentado implementam-se a classe Identificador de Conexão Tipo i como subclasse da classe abstrata Identificador de Conexão e a classe Conexão Tipo i como subclasse da classe abstrata Conexão, como mostrado na Figura 5.15. Na classe Identificador de Conexão Tipo i implementa-se o método Identificação, que compara os três tipos de equipamentos e o tipo de objeto movimentado de cada conexão presente no sistema de fabricação com aqueles que a caracterizam. Implementa-se também o Método de Fabricação, que cria um objeto da classe Conexão Tipo i toda vez que ocorre a identificação de uma conexão do respectivo tipo. O procedimento de simulação do processo produtivo da peça é realizado para cada uma das possíveis rotas. A simulação de cada rota é subdivida na simulação de todas as suas conexões, como mostrado na Figura 4.21. O método de Simulação da classe Conexão Tipo i deve ser implementado seguindo o padrão apresentado na figura, mas sem desconsiderar as particularidades de cada processo de movimentação. Inicialmente, simula-se a utilização dos três equipamentos, em seguida, a transferência do objeto movimentado entre o equipamento de origem e o de movimentação e, também, entre o de movimentação e o de destino e, finalmente, o processo completo de movimentação realizado na conexão. No exemplo mostrado neste trabalho, as simulações dos equipamentos, das transferências entre eles e do processo de movimentação em cada conexão são respectivamente efetuadas de acordo com as seções 4.3.3, 4.3.4 e 4.3.5. Nas conexões cujo equipamento de destino é de fabricação deve-se também simular a fabricação da peça. No exemplo mostrado simula-se o processo de serramento e de torneamento nas conexões correspondentes.

A simulação dos equipamentos não é normalmente realizada diretamente, mas sim por meio da simulação de seus dispositivos. Na terceira fase implementam-se os dispositivos que são utilizados pelos equipamentos que podem tomar parte no fluxo da peça durante o seu processo produtivo no sistema de fabricação. Pelo menos os dispositivos relativos aos equipamentos que fazem parte do fluxo principal precisam ser implementados. No exemplo mostrado neste trabalho somente a placa de fixação do torno representa um dispositivo

pertencente a um equipamento do fluxo principal. Entretanto, para realizar o fluxo da peça são utilizados robôs, que possuem garras, e transportadores, que utilizam paletes, como dispositivos, ou seja, como interface entre o equipamento e a peça durante o seu processo produtivo. Para representar um tipo específico de dispositivo deve-se implementar uma tabela no banco de dados relacional em que são inseridas as informações relativas ao dispositivo que precisam ser trocadas entre os módulos. Implementa-se também uma classe na interface entre o banco de dados e cada um dos módulos para manipular a respectiva tabela relacional. Em cada um dos três módulos deve-se implementar uma classe relativa ao dispositivo, que representa, respectivamente, uma janela de diálogo para manipular as informações contidas no banco de dados, para simular as possíveis formas de utilização do dispositivo e para auxiliar o procedimento de escolha da forma de utilização do dispositivo a ser empregada no processo produtivo. Normalmente, não é possível representar a estrutura de dados utilizada para descrever um dispositivo em uma única tabela relacional ou uma única classe. Portanto, na realidade são implementadas algumas tabelas relacionais, classes na interface e classes em cada um dos três módulos.

Para, por exemplo, implementar o tipo mais comum de placa de fixação utilizada em tornos, a placa de três castanhas mostrada na Figura 4.4 e modelada de acordo com a estrutura de classes da Figura 4.6, deve-se seguir os seguintes passos. Inicialmente, cria-se tabela relacional referente à placa de três castanhas e as tabelas relativas à castanha, ao degrau e à tampa, que complementam a descrição da placa. Na fase seguinte, criam-se as classes relativas à interface com o banco de dados. A classe denominada CIPlaca3Castanha é criada como subclasse da classe CIPlacaAbstrata que define a interface do *framework* para definição dos diversos tipos de placa, como mostrado esquematicamente na Figura 5.10. As classes CICastanha, CIDegrau e CITampa são criadas como subclasse de *PersistentObject*. Os relacionamentos entre essas classes são implementados de acordo com a Figura 5.12. Criam-se métodos para desenhar os componentes da placa e o Método de Fabricação para criar uma instância da classe correspondente na aplicação. Na última fase implementam-se as classes correspondentes nos três módulos do *framework*. No Módulo de Configuração cria-se a classe Placa3CastanhaDlg como subclasse de PlacaDlg, para implementar a janela de diálogo correspondente. A instanciação da classe Placa3CastanhaDlg como subclasse de PlacaDlg gera automaticamente um novo item na subdivisão do subitem Placa do menu principal mostrado na Figura 5.2 e o associa à abertura da janela de diálogo da placa de três castanhas. Criam-se também as classes CastanhaDlg, DegrauDlg e TampaDlg para implementar as

janelas de diálogos correspondentes. O acesso a essas janelas de diálogos é implementado na janela de diálogo da própria placa. Implementam-se métodos para relacionar cada um dos campos de introdução e/ou visualização de informações com os respectivos atributos da tabela relacional.

No Módulo de Simulação implementa-se a classe Placa3CastanhaSim como subclasse da classe da interface do *framework* denominada PlacaSim e as classes CastanhaSim, DegrauSim e TampaSim. Na classe Placa3CastanhaSim implementa-se o método Fixação que determina as diversas formas de fixação de uma peça pela placa de três castanhas. O método Fixação utiliza as demais classes para cumprir a sua responsabilidade. A cada fixação encontrada gera-se uma instância da classe FixacaoPlaca (Figura 4.31). No módulo de Escolha implementa-se a classe Placa3CastanhaEsc como subclasse da classe PlacaEsc, que define a interface comum do *framework* para escolha da utilização de todos os tipos de placas e as classes CastanhaEsc, DegrauEsc e TampaEsc. Na classe Placa3CastanhaEsc implementa-se o método Desenha, que acessa os métodos de desenho das demais classes para representar graficamente cada um dos componentes da placa de três castanhas para auxiliar o procedimento de escolha da fixação a ser utilizada durante o processo de torneamento. O posicionamento relativo de cada um dos componentes da placa de três castanhas é definido pelos dados da fixação.

De forma semelhante à apresentada para a placa de três castanhas implementam-se as tabelas relacionais e as classes da interface e da aplicação em cada um dos três módulos para cada um dos tipos de equipamentos, de dispositivos, de peças em bruto e de materiais em bruto utilizados durante o processo de produção de uma peça em um sistema de fabricação. Da mesma forma que a placa de três castanhas não é implementada por uma única classe, os demais itens normalmente são descritos por mais de uma tabela relacional e classes relacionadas entre si, tanto na interface como em cada um dos módulos. Os detalhes da instanciação completa do *framework* para criação da aplicação mostrada neste trabalho podem ser visto em seu manual de instruções (*cookbook*) (Massaroppi, 2002).

A estratégia de determinação do fluxo principal, as conexões, os equipamentos, os dispositivos e outros elementos descritos no capítulo anterior foram implementados para simular um sistema flexível de fabricação real, denominado IFMS (*Integriertes Fertigungs- und Montagesystem* – do alemão Sistema Integrado de Fabricação e Montagem) (Weck e Goedeke, 1987), no qual o autor teve a oportunidade de atuar.

Após implementar-se todas as tabelas relacionais e classes referentes aos pontos variáveis que caracterizam os três módulos de uma determinada aplicação instanciada a partir do *framework*, deve-se utilizá-los na mesma ordem em que se descreveram as suas instanciações. Inicialmente, executa-se o Módulo de Configuração para definir cada um dos elementos que constituem a fábrica modelada. Aconselha-se que sejam definidas, pela ordem, as células, os equipamentos, as conexões entre os equipamentos, os materiais e peças em bruto e os dispositivos, na mesma seqüência em que estes elementos aparecem no submenu de configuração mostrado na Figura 5.2. A execução do Módulo de Simulação é realizada automaticamente após a escolha do sistema de fabricação e da peça a ser simulada. O Módulo de Escolha deve ser executado diversas vezes, para que cada um de seus usuários faça a escolha relativa ao equipamento pelo qual é responsável. A execução é simples, pois cada usuário recebe em forma gráfica e tabular as informações relativas a cada uma das utilizações de um equipamento e seus dispositivos e, em seguida, indica qual a opção a ser utilizada na produção da peça.

A modelagem deste sistema de fabricação foi implementada e a simulação foi executada para algumas peças. O resultado da simulação de uma delas é apresentado em maiores detalhes no Capítulo 6.

5.7.2 Instanciação de uma aplicação semelhante a outra já realizada

As implementações realizadas para simular um determinado sistema de fabricação podem ser utilizadas como base para efetuar-se a simulação de outros sistemas de fabricação semelhantes. Se um sistema de fabricação possuir equipamentos de mesmo tipo que estejam agrupados de acordo com os mesmos tipos de conexões para produção de uma peça com a mesma estratégia que aqueles já implementados em desenvolvimentos anteriores, a implementação já realizada deve ser utilizada como ponto de partida para o novo desenvolvimento. Caso também os dispositivos, materiais em bruto e peças em bruto utilizados no novo sistema de fabricação sejam da mesma topologia que os já existentes, mesmo que possuam dimensões e outras características diferentes, pode-se efetuar a simulação com os módulos instanciados para o sistema de fabricação já modelado. Para tal, basta utilizar os três módulos da aplicação já desenvolvida de acordo com a descrição feita na seção anterior para descrever, simular e escolher a melhor alternativa de fabricação da peça. Desta forma amplia-se o espectro de sistemas de fabricação que podem ser simulados pela aplicação já instanciada a partir do *framework*. A aplicação criada pode simular qualquer

sistema de fabricação para a produção de peças segundo a mesma estratégia, que seja descrito por uma combinação qualquer das conexões, dos equipamentos e dos dispositivos implementados.

Se um sistema de fabricação é utilizado para produzir peças com simetria rotacional, mas que utilize, por exemplo, um processo final de retificação, a instanciação dos módulos da aplicação a partir do desenvolvimento realizado neste trabalho é significativamente mais simples do que se ela fosse realizada a partir do *framework*. Inicialmente, deve-se implementar uma nova subclasse da classe abstrata Processo de Fabricação, cujo método Fluxo Principal apresente como resposta a seqüência de equipamentos que inclua também a retificadora. Em seguida, implementam-se as classes Identificador de Conexão Tipo j como subclasses da classe Identificador de Conexão e as classes Conexão Tipo j como subclasses da classe Conexão para cada uma das conexões existentes no sistema de fabricação cujo equipamento de origem ou de destino seja a retificadora. Em cada classe Identificador de Conexão Tipo j implementa-se o método Identificação que compara os tipos de equipamentos e o objeto movimentado de cada conexão do sistema de fabricação com aqueles que a caracterizam. Implementa-se também o Método de Fabricação que gera uma nova instância da classe Conexão Tipo j toda vez que se identifica uma conexão do respectivo tipo. O método de Simulação implementado na classe Conexão Tipo j segue o mesmo padrão, simulando a utilização dos três equipamentos, das transferências no início e no fim do processo e o processo completo. Entretanto, se necessário, deve-se alterá-lo levando-se em consideração as particularidades de cada conexão. Como parte destas conexões tem a retificadora como equipamento de destino, um equipamento de fabricação, a conexão deve também simular o processo de retificação. Caso nesta instanciação do *framework* tenha-se também dispositivos diferentes daqueles já existentes, eles devem ser implementados de acordo com a descrição feita para a placa de três castanhas na seção anterior.

Finalmente, utilizam-se os três novos módulos instanciados com as extensões necessárias para descrever o sistema de fabricação, simular o fluxo da peça durante o seu processo produtivo e escolher a melhor forma de produzi-la.

Desta forma o espectro de sistemas de fabricação que pode ser simulado é expandido com um esforço de desenvolvimento relativamente pequeno ao implementar-se outras conexões, outros equipamentos e outros dispositivos para a produção de peças de acordo com estratégias semelhantes àquelas já implementadas. A existência de uma nova conexão ou um

novo dispositivo aumenta as combinações possíveis de sistemas de fabricação que podem ser simulados.

5.7.3 Instanciação de uma aplicação distinta de outras já realizadas

Entende-se por aplicações distintas aquelas que implementam estratégias de produção da peça significativamente distintas. A instanciação de aplicações aptas a simular sistemas de fabricação para peças, por exemplo, com outras geometrias, que só podem ser produzidas por processos de fabricação que são realizados em máquinas operatrizes distintas daquelas pertencentes a fluxos principais já implementados está associada a um grande esforço para a sua instanciação. Portanto, a viabilidade desta implementação deve ser bem avaliada. Neste caso deve-se implementar inicialmente a estratégia para determinação do fluxo principal, os identificadores das novas conexões e as conexões correspondentes. Em uma segunda fase, implementam-se os métodos de simulação dos novos equipamentos e dispositivos, dos novos tipos de transferências entre os equipamentos e dos processos que são realizados em cada conexão. Espera-se que poucas subclasses já implementadas possam ser reutilizadas neste tipo de instanciação. Após realizar as implementações necessárias, a utilização dos três módulos para simular a produção da peça é realizada da mesma maneira. Provavelmente, a maior dificuldade neste caso seja a implementação dos métodos de interação entre os dispositivos e a peça. A simulação da fixação no dispositivo da máquina-ferramenta ou na garra de um robô e a simulação da armazenagem em um palete de transporte de peças com outras geometrias são consideravelmente mais complexas do que aquelas descritas no capítulo anterior.

Caso queira-se instanciar aplicações que utilizem processos não convencionais de fabricação ou peças com geometrias com grande complexidade, deve-se avaliar mais rigorosamente a viabilidade do desenvolvimento a partir do *framework*. Neste caso, provavelmente o desenvolvimento de uma aplicação específica de forma convencional seja mais simples e rápido.

5.8 Implementação

Tanto o *framework* como as aplicações geradas a partir dele foram implementadas no sistema operacional Windows 98 da Microsoft. Ocasionalmente, utilizaram-se também as versões 95 e 2000 do mesmo sistema operacional sem que ocorresse qualquer problema, mas testes mais intensivos não foram realizados nessas versões. Como todos os módulos possuem janelas de diálogos, optou-se por efetuar o desenvolvimento com a ferramenta C++ Builder

Client/Server da Borland (1997), versão 1.0. Esta ferramenta utiliza a linguagem orientada a objetos C++ e o gerenciador de banco de dados DBase. A utilização de um banco de dados relacional, que pode ser manipulado pela linguagem SQL, facilita a implementação do padrão de persistência proposto por Yoder *et al*. (1998).

A plataforma de hardware utilizada variou entre pentium 200 MHz e pentium III 1 GHz. Com exceção da velocidade de compilação, as diferenças de desempenho não foram significativas, pois todos os módulos possuem interfaces gráficas para interação com o usuário. A interação representa em todos os casos o tempo mais significativo até a obtenção do resultado final.

A implementação não foi realizada como na arquitetura final proposta na Figura 5.5. Implementou-se o Módulo de Configuração como descrito mas o Módulo de Escolha não foi implementado por falta de tempo. Entretanto, a apresentação das informações gráficas e tabulares referentes à utilização de cada dispositivo e equipamento, às transferências entre os equipamentos e ao processo completo realizado em cada conexão, que são funções do Módulo de Escolha, foi implementada no Módulo de Simulação. Com isso, após o Módulo de Simulação realizar a sua tarefa, apresenta os seus resultados na tela. O usuário pode efetuar a escolha dos resultados a serem utilizados, mas as informações correspondentes não ficam armazenadas no banco de dados. Essas informações devem ser introduzidas nos sistemas externos para programação dos equipamentos (Figura 5.5) na forma tradicionalmente utilizada e não por interfaces apropriadas de acesso ao banco de dados. Porém, garante-se que as informações conduzam a utilizações compatíveis dos equipamentos e dispositivos.

O esforço de programação realizado para implementar o *framework* e instanciar uma aplicação pode ser visto na **Tabela 5.1**. Os módulos do *framework* compartilham arquivos comuns, o padrão de persistência de Yoder e as classes da interface com o banco de dados, assim como os módulos de uma aplicação compartilham as classes da interface com o banco de dados. Além dos pontos comuns, os módulos têm implementações específicas tanto no *framework* como na aplicação. A quantidade de linhas de comando apresentada na tabela é cerca de dez por cento menor que o número total de linhas para compensar linhas em branco e comentários existentes nos arquivos. Cada um dos módulos é constituído de aproximadamente vinte e cinco mil linhas de comando, sendo que pouco mais de dez mil linhas são comuns entre os dois módulos.

Tabela 5.1 – Tamanho do *framework* e da aplicação

		Módulo (número de linhas)	
		Configuração	Simulação
Framework	Arquivos Comuns	1.600	
	Padrão Persistência	1.800	
	Interface com o Banco de Dados	4.300	
	Arquivos Específicos	8.600	7.500
Aplicação	Interface com o Banco de Dados	2.800	
	Arquivos Específicos	5.800	7.000
Total	Linhas de Comando	24.900	25.000

5.9 Considerações Finais

Neste capítulo apresentou-se como foi realizada a transformação de um modelo conceitual de uma aplicação específica do domínio em um *framework*. A sua arquitetura, a necessidade da sua divisão funcional em três módulos e a integração entre eles com a utilização de um banco de dados relacional foram também apresentadas. Mostrou-se que a divisão de *frameworks* em módulos integrados por banco de dados é problemática, pois a persistência das informações contidas em seus pontos variáveis não é trivial. A associação de padrões utilizada para solucionar o problema foi também empregada em outro ponto do *framework*, o que pode vir a caracterizar um novo padrão. Comparou-se o *framework* proposto com dois *frameworks* desenvolvidos para o controle de sistemas flexíveis de fabricação. Finalmente, apresentou-se o projeto do *framework*, como instanciá-lo para a geração de aplicações e a implementação realizada no escopo deste trabalho.

6 Avaliação do Sistema Proposto

6.1 Considerações Iniciais

O teste de um *framework* é realizado não só para determinar se ele atende à funcionalidade pretendida mas também para avaliar se ele pode ser utilizado. Entretanto, não é trivial decidir se um *framework* é utilizável ou não. Os autores da área concordam que a única forma de saber se um software é reutilizável é reusando-o. No caso de *frameworks*, o projetista deve instanciar aplicações a partir dele para poder avaliá-lo. Dependendo do tipo de aplicação, o projetista pode testar diferentes aspectos do *framework*. A partir das aplicações instanciadas pode-se decidir se o *framework* deve ser reprojetado ou está suficientemente desenvolvido para ser liberado para uso.

Na seção 6.2 mostram-se os resultados de uma aplicação desenvolvida a partir do *framework*, que modela um sistema de fabricação real. Na seção 6.3 comparam-se os resultados da simulação obtidos com a aplicação instanciada a partir do *framework* com a forma utilizada para produção da peça. Na seção 6.4 apresentam-se as considerações finais deste capítulo

6.2 Uso do *Framework*

Para ilustrar e testar a instanciação do *framework* apresentado no capítulo anterior, foi usado como exemplo de um sistema flexível de fabricação o Sistema Integrado de Fabricação e Montagem (IFMS - do alemão ***I**ntegriertes **F**ertigungs- und **M**ontage**s**ystem*), instalado no Laboratório de Máquinas-Ferramenta da Universidade de Aachen na Alemanha (*Laboratorium für Werkzeugmaschinen und Betriebslehre der RWTH Aachen*) (Weck & Goedeke, 1987). O sistema foi construído para a fabricação e montagem de peças com simetria rotacional, prismáticas e algumas formas especiais. Este estudo de caso é realizado para peças com simetria rotacional que utilizam um torno durante a sua produção.

6.2.1 Descrição do Sistema Flexível de Fabricação

O sistema é composto por mais de quarenta equipamentos divididos em cinco células. Este exemplo é realizado apenas para a produção de peças e, portanto, as células de montagem e de medição não são relevantes. As três células relevantes são compostas por mais de vinte equipamentos mas nem todos são utilizados neste exemplo. A célula de comissionamento possui um depósito de material em bruto, um transportador de material em

bruto, uma serra, dois *buffers* de paletes (um de descarga da serra e outro de saída), um robô de superfície para manipulação de peças em bruto, um robô linear para transporte de paletes e outros equipamentos não utilizados neste estudo de caso. A célula de transporte é composta por um veículo autoguiado e pela estação de recarga de suas baterias. A célula de fabricação é composta por seis *buffers* de paletes (um de entrada e saída da célula, dois para armazenagem de peças manipuladas pelo robô de alimentação e descarga das máquinas-ferramenta e três intermediários), um transportador de paletes, um robô de joelho, um torno e outros equipamentos que não participam do fluxo de peças com simetria rotacional. A **Figura 6.1** mostra uma vista panorâmica e um esquema do arranjo físico (*layout*) do sistema modelado.

6.2.2 Instanciação de uma Aplicação

Para instanciar uma aplicação que simule a produção de uma peça pelo sistema de flexível de fabricação descrito na seção anterior, deve-se, inicialmente, efetuar a modelagem dos equipamentos, dos dispositivos e das conexões que individualizam esse sistema. Para gerar as aplicações correspondentes a cada um dos módulos, o primeiro passo consiste em criar as tabelas relacionais para persistência da informação das características individuais do sistema e as classes correspondentes na interface entre os módulos e o banco de dados. No caso de uma longarina para apoio na horizontal, por exemplo, cria-se uma tabela relacional com os atributos correspondentes aos dados geométricos mostrados na Figura 4.11. A classe da interface com o banco de dados CILongarinaPrismaHorizontal é implementada com os mesmos atributos que a tabela relacional, como subclasse da classe base CILongarinaAbstrata, que define a interface do *framework* para definição dos diversos tipos de longarinas.

Em seguida implementam-se as classes específicas de cada um dos módulos. Para uma mesma instância de uma determinada variabilidade criam-se classes que especificam uma janela de diálogo no Módulo de Configuração, que descrevem o comportamento, por exemplo, de um dispositivo no Módulo de Simulação e que auxiliam a realizar a opção da forma de emprego desse dispositivo no Módulo de Escolha. Para o exemplo acima, cria-se a classe LongarinaPrismaHorizontalDlg, como subclasse da classe base LongarinaDlg, de acordo com a Figura 5.3, a classe LongarianPrismaHorizontalSim como subclasse da classe base LongarinaSim e a classe LongarinaPrismaHorizontalEsc como subclasse da classe base LongarinaEsc, que definem as interfaces do *framework* nos três módulos.

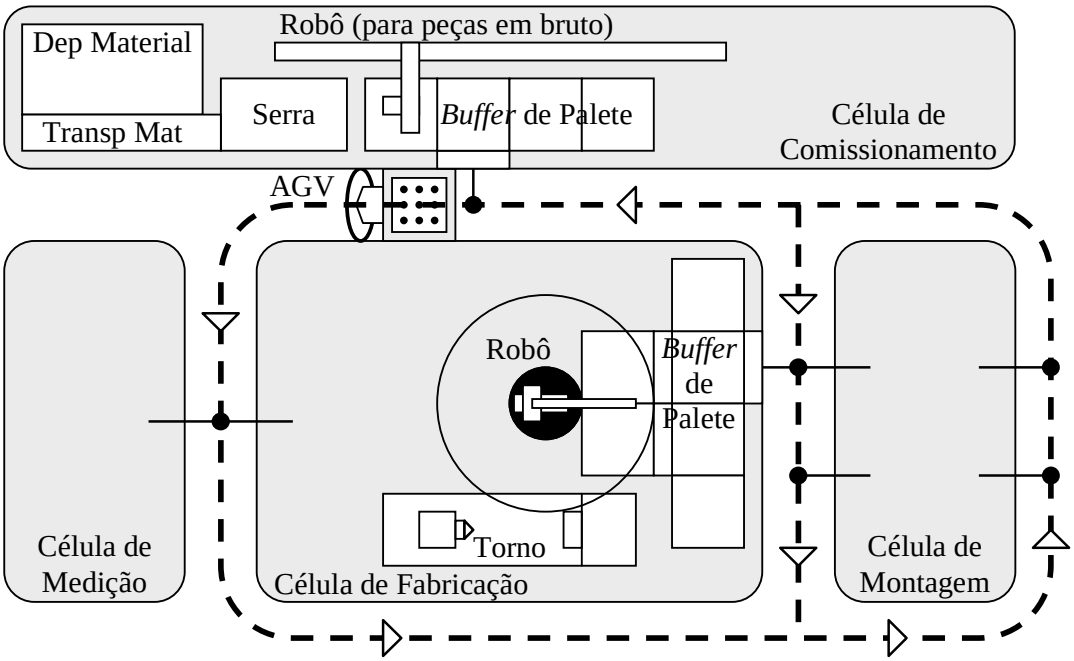


Figura 6.1 – Sistema Integrado de Fabricação e Montagem (IFMS)

Após gerar as aplicações referentes aos três módulos, utiliza-se o Módulo de Configuração para introduzir no banco de dados as informações específicas do sistema

flexível de fabricação modelado. Para o exemplo apresentado, deve-se preencher as informações correspondentes na janela de diálogo da Figura 5.3.

6.2.3 Dados de Entrada para a Simulação

Para efetuar a simulação do fluxo da peça, o usuário do Módulo de Simulação deve escolher o sistema de fabricação onde a simulação deve ocorrer, pois no banco de dados podem estar modelados mais de um sistema de fabricação, a peça e a estratégia de produção. Neste exemplo escolhe-se o sistema mostrado na Figura 6.1. A peça a ser produzida é um parafuso especial, com comprimento de 107 mm e diâmetro máximo de 45 mm, mostrado na **Figura 6.2**. A estratégia de produção é a de torneamento, ou seja, os equipamentos que participam do fluxo principal são, na seqüência, um deposito de material em bruto, uma serra, um torno e um depósito de peças prontas, conforme já foi apresentado na Figura 4.17.

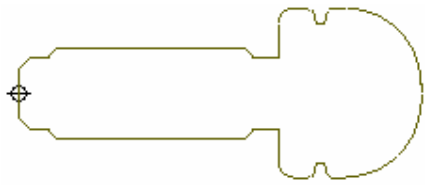


Figura 6.2 – Parafuso especial

6.2.4 Resultados da Simulação

A simulação inicia pela busca de rotas que fluam pelos equipamentos do fluxo principal. Encontram-se quatro rotas, que diferem apenas nos *buffers* de paletes intermediários utilizados durante o encaminhamento do palete de peças em bruto da entrada da célula de fabricação até o *buffer* próximo ao torno e do palete de peças prontas no sentido inverso. Como os paletes utilizados são os mesmos e todos os *buffers* de paletes estão aptos a recebê-los, os demais resultados são exatamente iguais e, por isso, será mostrado apenas o resultados da simulação referente a uma das rotas. A estrutura em árvore da rota, composta por seções, subseções e conexões, pela qual flui a peça, é mostrada na **Figura 6.3**. A figura apresenta as conexões nos quadros em tom mais claro de cinza. Assim como na Figura 4.19, as conexões possuem uma seta direcionada do equipamento de origem para o equipamento de destino. O equipamento de movimentação é representado abaixo da seta e o objeto movimentado acima dela. Com exceção do primeiro e do último equipamento de origem e de destino, todos participam de duas conexões consecutivas. Em tonalidade média estão representadas as subseções, de acordo com a Figura 4.18. As seções do fluxo principal, conforme a Figura

4.17, são mostradas em tonalidade mais escura. Na última conexão, a seta é mostrada em linha tracejada, pois esta parte da simulação não é realizada. A simulação é efetuada até que a peça pronta seja encaminha à saída da célula de fabricação. O passo seguinte não é simulado, pois a peça pode seguir tanto para o depósito de peças prontas, como também para a célula de montagem ou de medição.

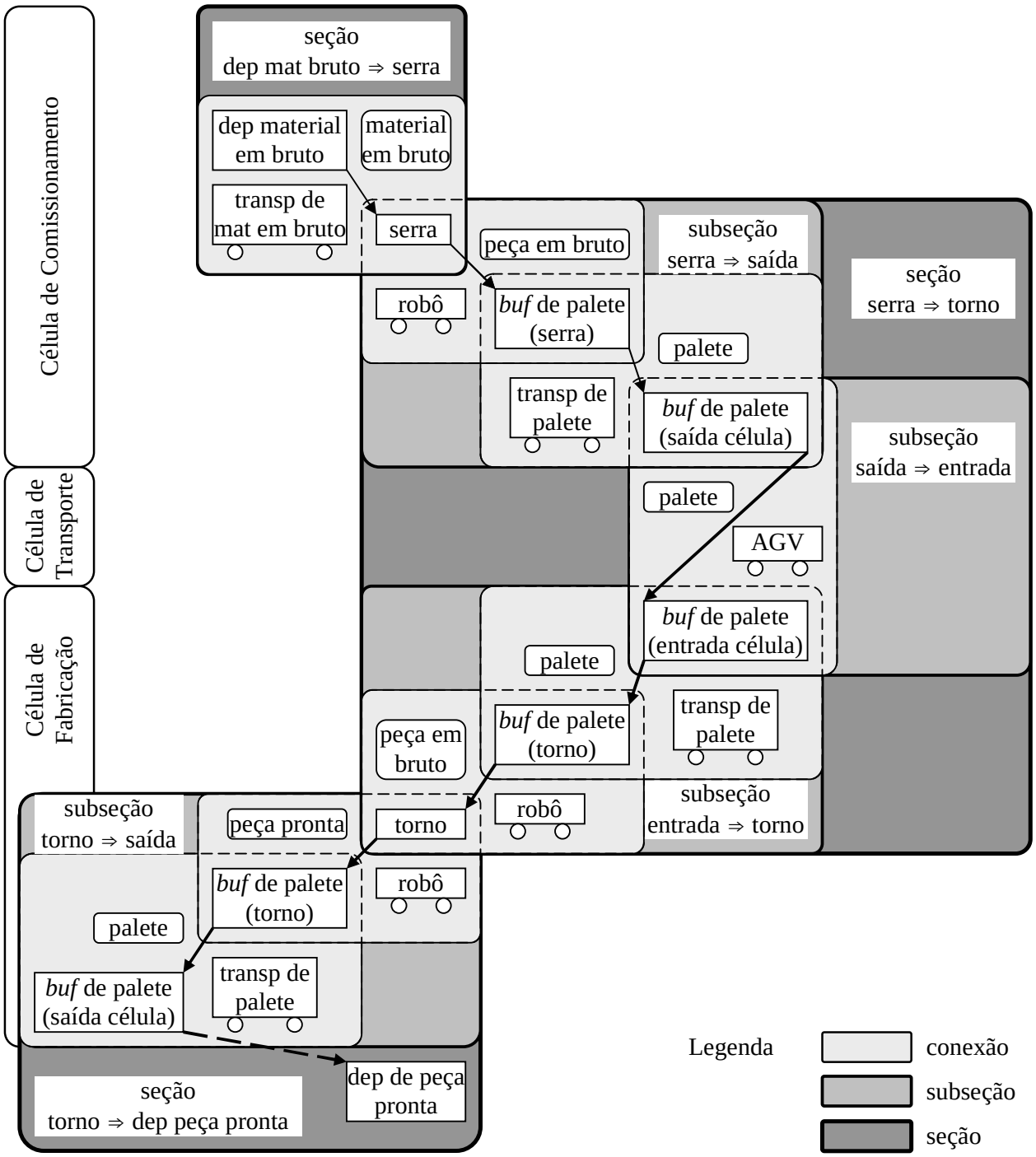


Figura 6.3 – Rota de produção da peça

O passo seguinte consiste em simular cada uma das conexões de acordo com a estrutura da rota mostrada na Figura 6.3. Os resultados da simulação de parte dessas conexões não são únicos. A escolha do resultado mais viável é realizada idealmente pelos usuários do Módulo de Escolha. Entretanto, esse módulo não foi implementado e, por isso, a escolha é realizada durante a execução do Módulo de Simulação, sem que as opções sejam registradas no banco de dados. Os resultados do procedimento de escolha relativa a cada equipamento presente no fluxo são mostrados a seguir.

a) Conexão 1:

Equipamento de origem:	Deposito de Material em Bruto
Equipamento de destino:	Serra Kasto
Equipamento de transporte:	Transportador da Serra
Objeto transportado:	Material em Bruto

O teste correspondente inicia com a verificação da existência de materiais em bruto adequados (dimensões e material) à produção da peça no Deposito de Material em Bruto. Encontra-se o material em bruto denominado “barra de alumínio d 55”, com diâmetro externo de 55 mm e com diâmetro interno de 0 mm, ou seja, trata-se de uma barra maciça e não de um tubo. Em seguida, verifica-se se a capacidade da Serra Kasto é compatível com as dimensões do material em bruto. O diâmetro externo do material em bruto pode ser cortado pela serra. A resolução da capacidade de corte da serra permite que o material em bruto seja cortado com o comprimento mínimo de 111 mm e com uma precisão de ±0,1 mm, para uma peça com comprimento final de 107 mm. A diferença deve-se à necessidade de sobremetal nas duas extremidades da peça. O Transportador da Serra é considerado sempre apto a conduzir materiais em bruto do Depósito até a Serra, ou seja, as seguintes verificações não precisam ser realizadas: i) se o Transportador pode transportar o material em bruto; ii) se as transferências do material em bruto, tanto entre o Deposito e o Transportador como entre o Transportador e a Serra, podem ser efetuadas; e iii) se o processo completo pode ser realizado. Essas verificações não são realizadas, pois se considera que o Transportador seja construído especificamente para a capacidade da serra.

A peça em bruto obtida no processo de serramento pode ser vista na **Figura 6.4**. A figura mostra também como a peça fica apoiada sobre a mesa da serra. O círculo preto com uma cruz representa o ponto de referência da peça. Em outras figuras aparece também o ponto de referência de outros equipamentos.

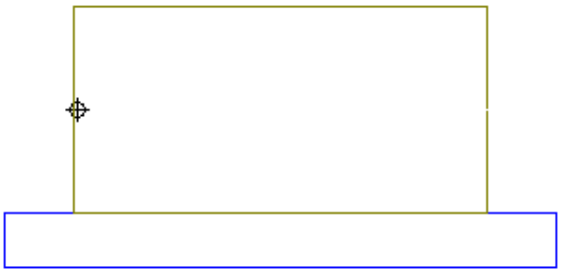


Figura 6.4 – Disposição da peça em bruto na mesa da Serra Kasto

b) Conexão 2:

Equipamento de origem:	Serra Kasto
Equipamento de destino:	*Buffer* de Palete de Descarga da Serra
Equipamento de manipulação:	Pórtico Mannesmann Demag (robô)
Objeto manipulado:	Peça em Bruto

A disponibilidade de peça em bruto na mesa da Serra é aquela encontrada na simulação da conexão anterior (Figura 6.4). Nesta conexão simulam-se inicialmente as possíveis fixações com todas as garras do Robô de Pórtico e as armazenagens em todos os paletes que podem ser colocados no *Buffer*. Uma fixação com a garra e cinco armazenagens sobre o palete de longarinas são encontradas. Em um segundo passo simulam-se as transferências da peça em bruto tanto da Serra para o Robô, como do Robô para o *Buffer*. A transferência da única forma da peça apoiada sobre a mesa da serra para a única fixação com a garra é possível. Das cinco armazenagens sobre o palete encontradas, somente uma é compatível para a transferência da peça proveniente da fixação com a garra.

Finalmente, simula-se o processo completo relativo a essa conexão ao confrontarem-se as possíveis transferências no início e no fim do processo que são caracterizadas pela mesma fixação com a garra. A posição da peça em bruto em relação à garra fica limitada à intersecção dos valores das posições relativas obtidos no início e fim do processo. Caso a intersecção seja nula, significa que a fixação correspondente com a garra não pode ser utilizada. No caso, a intersecção resulta em um intervalo que representa os valores da posição relativa entre peça em bruto e a garra que podem ser utilizados para realização do processo.

A **Figura 6.5** mostra a fixação com a garra da peça em bruto. Na **Figura 6.6** apresenta-se a armazenagem da peça em bruto sobre o palete de longarinas. O processo completo realizado nesta conexão é representado pela **Figura 6.7**. A figura mostra uma interferência

entre a mesa da serra e as longarinas do palete de transporte. Fisicamente essa interferência não ocorre, pois o robô retira a peça em bruto da mesa da serra e manipula-a até as longarinas. A representação utilizada facilita a verificação visual da qualidade de todo o processo, pois em uma única figura têm-se informações sobre a utilização dos três dispositivos dos equipamentos que participam da conexão, sobre a forma como é realizada a transferência entre eles no início e no fim do processo e sobre o intervalo de intersecção em que a garra pode fixar a peça em bruto.

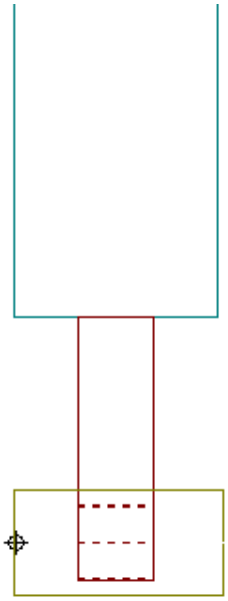


Figura 6.5 – Fixação da peça em bruto com a garra do robô

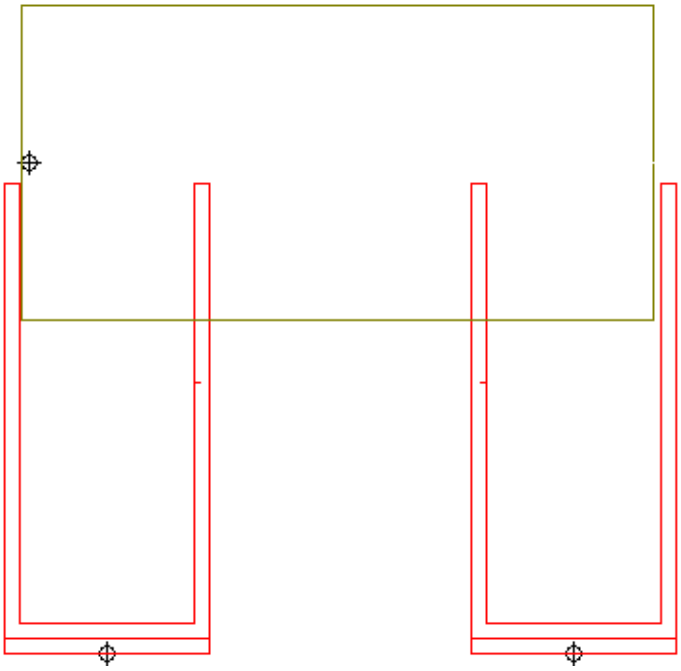


Figura 6.6 – Armazenagem da peça em bruto no palete de longarinas

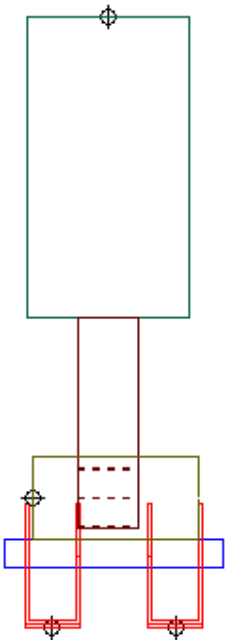


Figura 6.7 – Processo de descarga da serra Kasto

c) Conexão 3:

Equipamento de origem:	*Buffer* de Palete de Descarga da Serra
Equipamento de destino:	*Buffer* de Saída de Palete
Equipamento de transporte:	Transportador de Paletes
Objeto transportado:	Palete

As armazenagens de peça em bruto no palete de transporte colocado no *Buffer* de Palete de Descarga da Serra são aquelas encontradas na verificação da conexão anterior, que são compatíveis com a manipulação correspondente, ou seja, apesar de ser possível armazenar a peça em bruto de cinco diferentes formas, a garra existente só é apta a colocar a peça de acordo com a armazenagem mostrada na Figura 6.6. Portanto, para a simulação desta conexão só se admite que a peça em bruto esteja assim armazenada.

Essa conexão transporta o palete que realiza a armazenagem da peça em bruto. Os equipamentos dessa conexão não têm contato direto com a peça, o que torna a sua simulação mais simples. Só é necessário verificar se os paletes (no caso um único palete), onde a peça em bruto está armazenada, pode ser transportado pelo Transportador de Paletes (equipamento de transporte) e se o equipamento de destino (*Buffer* de Saída de Paletes) pode recebê-lo. É

uma verificação simples, pois para cada equipamento existe a lista de paletes que ele pode receber. O palete pode ser transportado pelo Transportador de Paletes para o *Buffer* de Saída de Paletes e a armazenagem da peça em bruto permanece a mesma sobre o palete de longarinas.

d) Conexão 4:

Equipamento de origem:	*Buffer* de Saída de Palete
Equipamento de destino:	*Buffer* de Entrada e Saída de Palete
Equipamento de transporte:	AGV Mannesmann
Objeto transportado:	Palete

e

e) Conexão 5:

Equipamento de origem:	*Buffer* de Entrada e Saída de Palete
Equipamento de destino:	*Buffer* de Palete (traseiro direito)
Equipamento de transporte:	Transportador de Paletes
Objeto transportado:	Palete

As conexões 4 e 5 são semelhantes à conexão 3. A simulação segue a mesma estratégia. Destaca-se que a conexão 4 é um transporte da célula de comissionamento para a célula de fabricação. Para que essa conexão possa ocorrer é necessário que os equipamentos de origem e destino sejam interfaces de suas células e que o equipamento de transporte pertença a uma célula de transporte. Essa verificação é realizada no Módulo de Configuração para garantir a consistência das informações armazenados no banco de dados.

f) Conexão 6:

Equipamento de origem:	*Buffer* de Palete (traseiro direito)
Equipamento de destino:	Torno Monforts
Equipamento de manipulação:	Robô Kuka
Objeto manipulado:	Peça em Bruto

As armazenagens da peça em bruto no palete de transporte colocado no *Buffer* de Palete são aquelas encontradas durante a simulação das conexões anteriores, pois desde a segunda conexão o palete está sendo transportado, sem alteração da armazenagem da peça em bruto. Conseqüentemente, a peça em bruto está armazenada como foi mostrado na Figura 6.6, que foi a única possibilidade encontrada naquela conexão.

Inicialmente, verificam-se as possibilidades de fixação da peça em bruto com todas as garras do Robô Kuka e as possibilidades de fixação da peça em bruto com todas as placas do Torno Monforts. Uma fixação com a garra do Robô Kuka e quatro fixações com a placa do Torno Monforts são encontradas. Em um segundo passo verificam-se as possibilidades de transferência da peça em bruto entre os dispositivos, tanto do palete do *Buffer* de Palete (traseiro direito) para a garra do Robô Kuka, como desta para a placa do Torno Monforts. A transferência da única forma de armazenagem da peça para a única fixação com a garra é possível. Todas as quatro fixações com a placa são compatíveis para a transferência da peça proveniente da fixação com a garra.

Finalmente, simula-se o processo completo de alimentação do torno ao confrontarem-se as possíveis transferências no início e no fim do processo que utilizam a mesma fixação com a garra. A posição da peça em bruto em relação à garra fica limitada à intersecção dos valores das posições relativas no início e fim do processo. A transferência do palete para a garra é compatível com as quatro transferências da garra para a placa. Conseqüentemente, existem quatro possibilidades distintas de efetuar a alimentação do torno.

A **Figura 6.8** mostra a fixação da peça em bruto com a garra do Robô Kuka. Na **Figura 6.9** apresenta-se a única fixação da peça em bruto na placa do Torno Monforts que é compatível com os passos seguintes do fluxo da peça. As demais fixações são invalidadas quando se efetua a simulação inversa da rota (seção 4.3.6). O processo de alimentação do torno é representado na **Figura 6.10**. Da mesma forma que a figura correspondente foi apresentada na segunda conexão, também nessa figura apresenta-se a atuação simultânea dos três dispositivos para facilitar a verificação visual da qualidade de todo o processo.

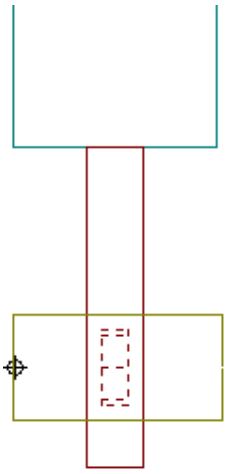


Figura 6.8 – Fixação da peça em bruto com a garra do robô Kuka

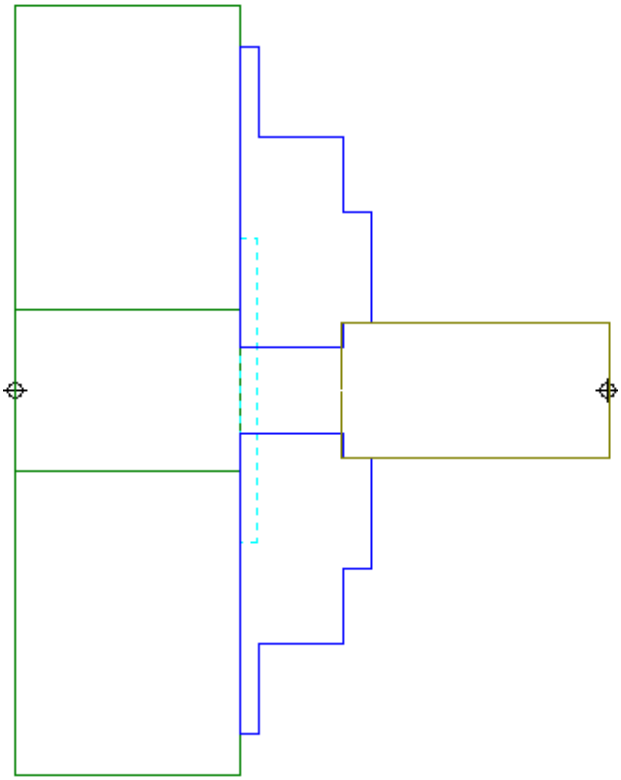


Figura 6.9 – Fixação da peça em bruto com a placa do torno Monforts

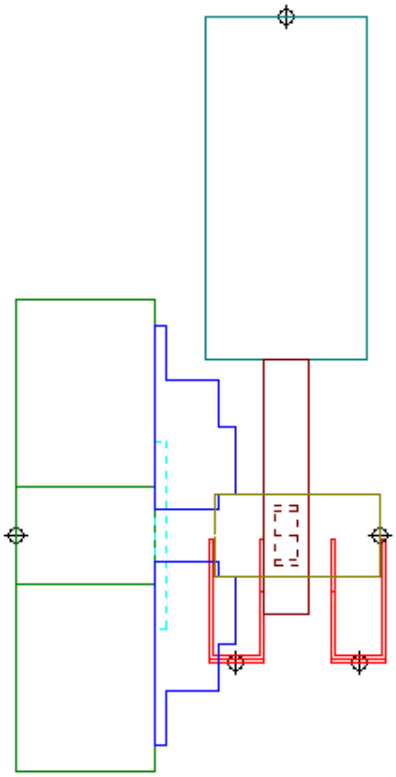


Figura 6.10 – Processo de alimentação do torno Monforts

Nesta conexão simula-se também o processo de torneamento. O processo é dividido em três etapas. Inicialmente, simula-se a usinagem do primeiro lado da peça, em seguida, a sua refixação e, finalmente, a usinagem do seu segundo lado. Antes de simular a primeira usinagem deve-se determinar quais as possíveis geometrias da peça semipronta. Analisando-se o contorno da peça pronta encontram-se quatro pontos de inversão da usinagem. Dois são relativos à primeira usinagem em um dos lados da peça e os outros dois relativos à usinagem no lado oposto da peça. A primeira usinagem referente às geometrias associadas aos dois primeiros pontos de inversão pode ser realizada, mas é incompatível com passos seguintes do fluxo da peça e é invalidada. A primeira usinagem referente às outras duas possíveis geometrias é compatível com todo o fluxo da peça. Somente no Módulo de Escolha é que é feita a opção sobre qual das duas geometrias deve ser utilizada. A opção escolhida é aquela sugerida pelo sistema, por estar relacionada com o menor índice de fixação com a garra (Massaroppi & Masiero, 2002), durante o processo de refixação. A geometria escolhida é mostrada na **Figura 6.11** e a **Figura 6.12** apresenta a peça semipronta fixada na placa após a simulação da primeira usinagem.

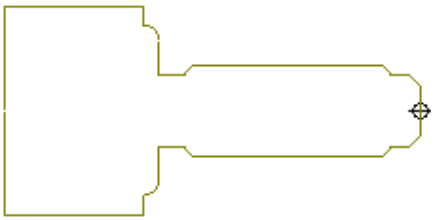


Figura 6.11 – Geometria da peça semipronta

A simulação da refixação é semelhante à simulação de uma conexão, pois a peça semipronta encontra-se na placa do torno (equipamento de origem) e um robô (equipamento de manipulação) deve retirá-la e colocá-la novamente na mesma placa do torno, que é interpretada na segunda fixação como sendo o equipamento de destino. Neste processo existe a restrição que a primeira fixação deve necessariamente ser realizada no lado não usinado da peça semipronta e a segunda fixação no lado já usinado. A garra do Robô Kuka pode fixar a peça semipronta de três modos diferentes. A segunda fixação da peça semipronta com a placa do torno Monforts pode ser realizada de três maneiras distintas. Dentre as possíveis fixações da peça semipronta em seu lado já usinado, uma delas apresenta o índice de fixação (Massaroppi & Masiero, 2001a; 2001b) significativamente menor que as demais e, portanto, é escolhida como a forma de fixação a ser utilizada. Das três formas de fixações com a garra, somente uma é compatível com a segunda fixação com a placa escolhida. A **Figura 6.13**

mostra a fixação com a garra da peça semipronta. Na **Figura 6.14** é mostrada a segunda fixação com a placa da peça semipronta. A refixação da peça semipronta é mostrada na **Figura 6.15**. A figura também mostra a atuação simultânea dos três dispositivos. Na realidade são dois, pois a placa do Torno Monforts é apresentada duas vezes, fixando cada um dos lados da peça semipronta.

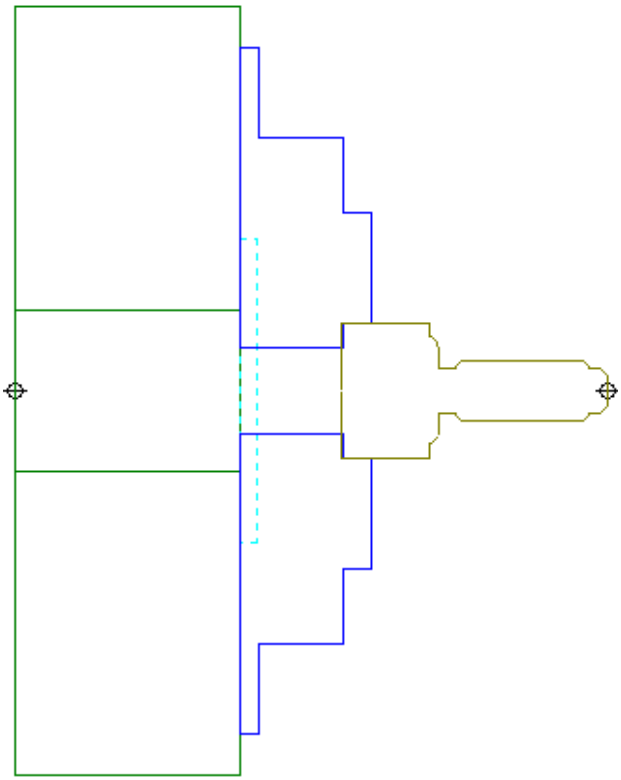


Figura 6.12 – Fixação da peça semipronta com a placa do torno Monforts após a primeira usinagem

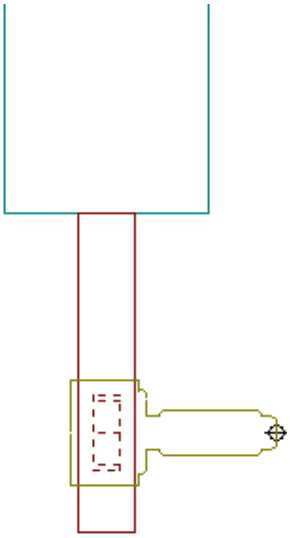


Figura 6.13 – Fixação da peça semipronta com a garra do robô Kuka

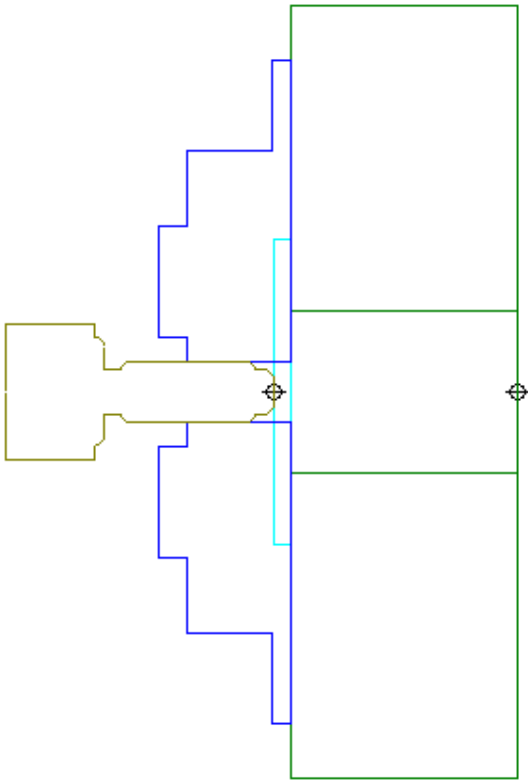


Figura 6.14 – Segunda fixação da peça semipronta com a placa do torno Monforts

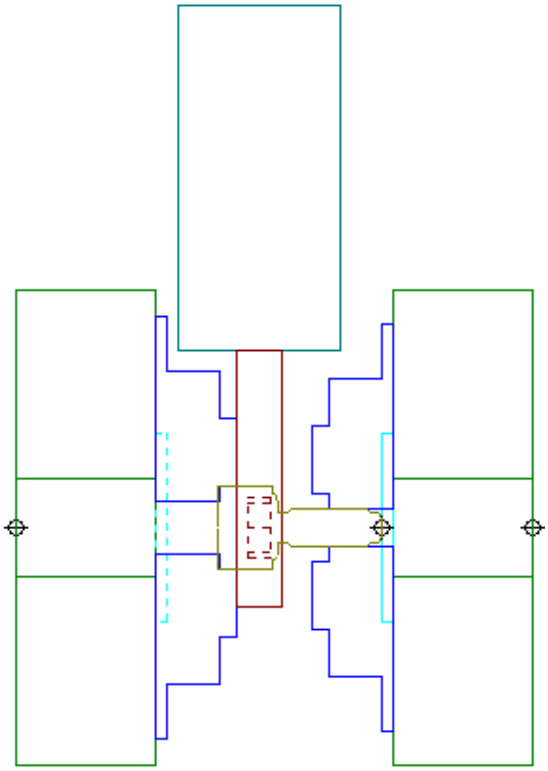


Figura 6.15 – Processo de refixação da peça semipronta no torno Monforts

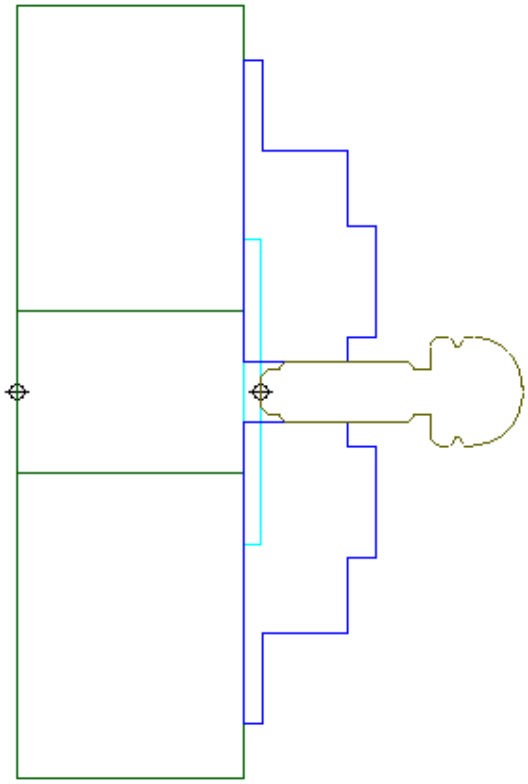


Figura 6.16 – Fixação da peça pronta com a placa do torno Monforts

A simulação do processo de torneamento é encerrada com a simulação da segunda usinagem. A simulação é bem simples, pois basta remover do lado ainda não usinado o material excedente. A fixação da peça pronta é idêntica à segunda fixação da peça semipronta. Só a geometria da peça é alterada. A **Figura 6.16** mostra a fixação da peça pronta, após a segunda usinagem. Existem três possibilidades de fixação da peça pronta correspondentes às três fixações da peça semipronta. Entretanto, como já foi mencionado, uma é significativamente melhor que as demais.

g) Conexão 7:

Equipamento de origem:	Torno Monforts
Equipamento de destino:	*Buffer* de Palete (traseiro direito)
Equipamento de manipulação:	Robô Kuka
Objeto manipulado:	Peça Pronta

As possíveis fixações com a placa do Torno Monforts são aquelas provenientes da conexão anterior. Das cinco possibilidades de fixação da peça pronta somente três são compatíveis com o processo de torneamento já simulado. A Figura 6.16 mostrou a fixação

com a placa escolhida, mas no âmbito desta conexão a simulação é realizada com todas as fixações possíveis. Nesta conexão simulam-se inicialmente as possíveis fixações com todas as garras do Robô Kuka e as armazenagens em todos os paletes que podem ser colocados no *Buffer*. Encontram-se cinco fixações com as duas garras do robô e três armazenagens sobre o palete de longarinas. No segundo passo simulam-se as transferências da peça pronta tanto da placa do torno para as garras do robô, como dessas para o palete de longarinas. Existem oito possibilidades de transferência da peça pronta da placa do torno para as garras do robô e três possibilidades de transferências das garras do robô para o palete de longarinas. Das cinco possibilidades de fixação com as garras do robô, somente três são compatíveis com alguma transferência proveniente da placa do torno e somente duas com alguma transferência para o palete de longarinas. Essa restrição facilita o passo final da simulação do processo de descarga do torno.

Finalmente, simula-se o processo completo de descarga do torno utilizando o confronto das possíveis transferências no início e no fim do processo, que são realizadas com a mesma fixação com a garra. A posição da peça pronta em relação à garra fica limitada à intersecção das posições relativas no início e no fim do processo. Existem quatro possibilidades de realização do processo associadas à segunda fixação com a garra e três possibilidades associadas à quarta fixação com a garra. Entretanto, a fixação com a placa do Torno Monforts escolhida na conexão anterior (Figura 6.16) é compatível com apenas uma forma de realização do processo de descarga do torno.

A **Figura 6.17** mostra a fixação da peça pronta com a garra do Robô Kuka. Na **Figura 6.18** apresenta-se a armazenagem da peça pronta sobre o palete de longarinas. O processo de descarga do torno é representado pela **Figura 6.19**. Ao contrário das conexões anteriores, em que se apresentou em uma única figura a atuação simultânea dos três equipamentos, na Figura 6.19 mostram-se as duas transferências, uma da placa do torno para a garra do robô no início do processo e outra da garra do robô para as longarinas do palete de transporte no fim do processo, pois ocorre uma mudança da orientação da peça. Para desenhar em uma única figura, o palete de transporte precisaria sofrer uma rotação de noventa graus em relação à sua orientação normal de trabalho. Mesmo em duas imagens é possível efetuar a verificação visual de todo o processo.

Os resultados do processo de descarga do torno podem ser visto de forma abstrata na Figura 4.22. A figura não apresenta a forma de utilização de cada dispositivo mas mostra qual delas são compatíveis para transferência e para a realização da descarga do torno.

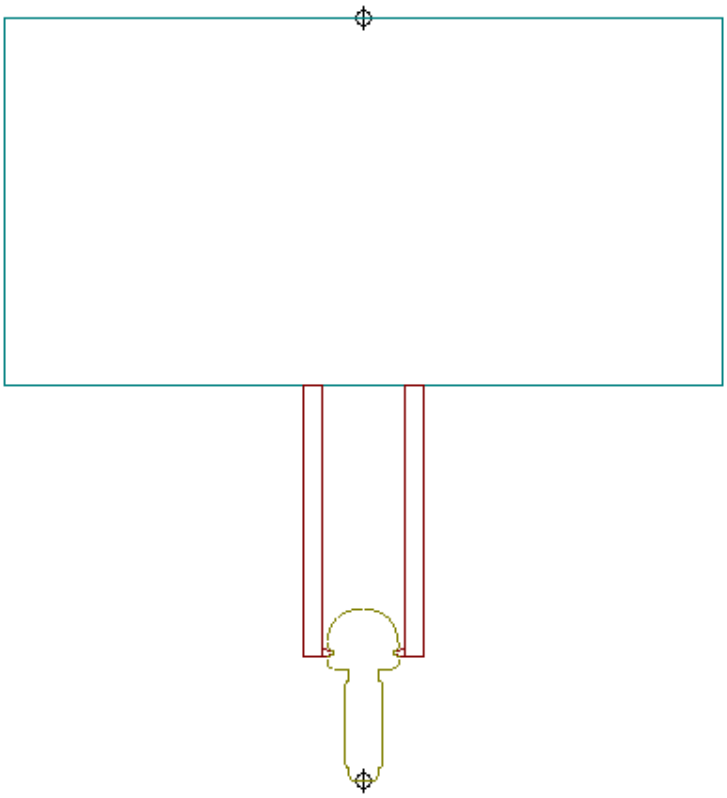


Figura 6.17 – Fixação da peça pronta com a garra do robô Kuka

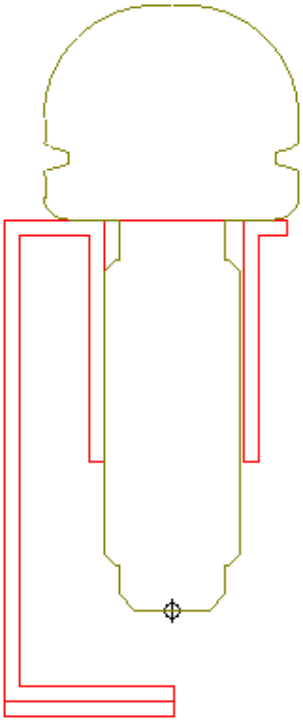


Figura 6.18 – Armazenagem da peça pronta sobre o palete de longarinas

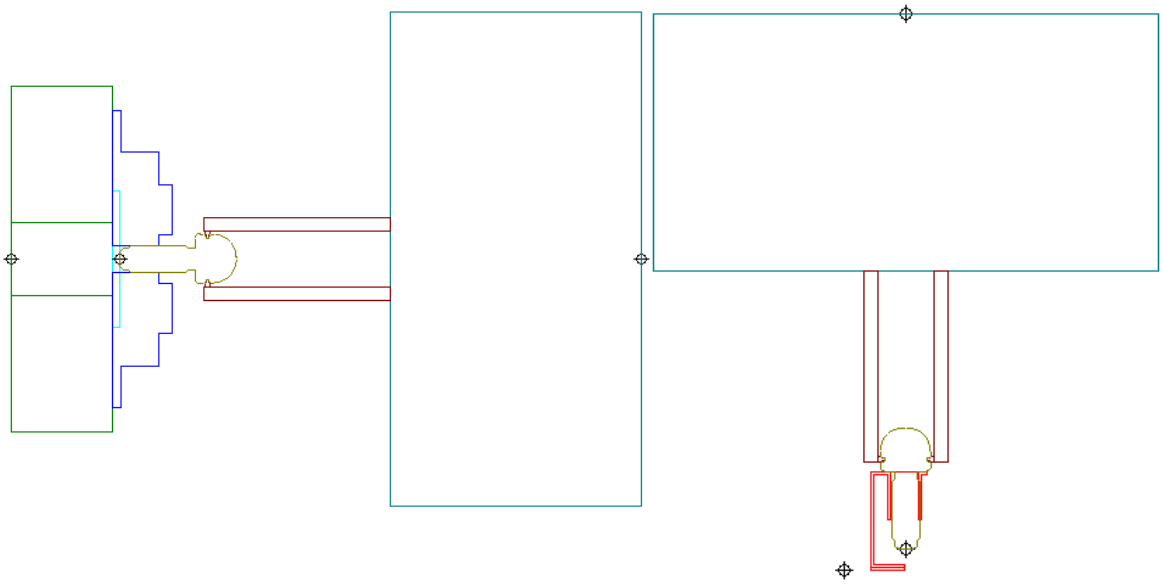


Figura 6.19 – Processo de descarga do torno Monforts

h) Conexão 8:

Equipamento de origem:	*Buffer* de Palete (traseiro direito)
Equipamento de destino:	*Buffer* de Entrada e Saída de Palete
Equipamento de transporte:	Transportador de Paletes
Objeto transportado:	Palete

Esta conexão, utilizada para o transporte da peça pronta desde o *buffer* de paletes próximo ao torno até o *buffer* na saída da célula, é semelhante às conexões 3, 4 e 5. A simulação segue a mesma estratégia. Verifica-se apenas se o palete pode ser recebido pelos três equipamentos que fazem parte da conexão. Todos os equipamentos estão aptos a receber o palete e, portanto, a peça pronta permanece armazenada sobre ele, como antes da realização deste passo do fluxo.

i) Demais Conexões:

A peça pronta encontra-se no palete de transporte no *buffer* para paletes na saída da célula de fabricação. A partir deste ponto o veículo autoguiado pode transportar o palete para outra célula qualquer. O palete pode, por exemplo, ir para a célula de medição a fim de efetuar-se algum controle de qualidade sobre a peça pronta ou para a célula de montagem,

caso a peça seja utilizada para a formação de um conjunto ou ainda para a célula de comissionamento para que a peça seja armazenada em um depósito de peças prontas. As simulações correspondentes não são realizadas, pois o método e o *framework* propostos no escopo deste trabalho são direcionados à produção das peças e não à montagem ou ao controle de qualidade.

A descrição completa do fluxo desta peça pelo sistema de fabricação pode ser vista no manual de instruções do *framework* (Massaroppi, 2002).

6.3 Avaliação do Método de Simulação e do *Framework*

O exemplo de uso do sistema proposto apresentado na seção anterior foi utilizado para avaliar tanto o método de simulação como o *framework*. A aplicação encontrou resultados que refletem o comportamento físico do sistema flexível de fabricação modelado. As simulações dos três tipos de dispositivos modelados apresentam resultados compatíveis com as suas utilizações típicas. Os resultados encontrados da simulação da fixação das peças em bruto, semipronta e pronta com a placa do torno são geometricamente realizáveis. O índice de fixação calculado representa a qualidade da fixação e é utilizado para ordená-las de acordo com o critério proposto. Também os resultados da simulação da fixação da peça com as garras de robôs modeladas correspondem àqueles fisicamente realizáveis. Alguns dos resultados encontrados não foram previstos por este autor. Eles são de qualidade significativamente inferior e, provavelmente, nunca seriam utilizados em um sistema de fabricação real. Porém, este fato evidencia que o método é suficientemente abrangente e pode ser utilizado para apoiar a decisão de um especialista do domínio. Da mesma forma os resultados encontrados na simulação da armazenagem da peça nos paletes de longarinas modelados são ordenados de acordo com o critério de qualidade proposto e podem ser utilizados no sistema de fabricação real.

Os resultados da simulação das transferências e dos processos realizados em cada uma das conexões são compatíveis com as possíveis formas como eles podem ocorrer. A simulação do fluxo total da peça desde o depósito de material em bruto até o depósito de peça pronta apresenta como resultado exatamente a forma utilizada para a produção da peça, escolhida por uma equipe de especialistas do domínio.

A validação de um método não pode ser realizada com os poucos exemplos simulados no escopo deste trabalho. Entretanto, o fato de uma simulação do fluxo da peça realizada em poucos minutos apresentar o mesmo resultado que uma equipe de especialistas demorou

semanas para obter, mostra que o método proposto apresenta resultados corretos e deve ser mais bem testado a fim de ter a validação confirmada.

O mesmo ocorre com o *framework* e a aplicação instanciada a partir dele. O fluxo de controle do *framework* foi executado de acordo com o seu projeto. Os métodos referentes aos dispositivos e a outros elementos implementados nos subsistemas de pontos variáveis do *framework* foram corretamente requisitados pelos respectivos clientes. As respostas às requisições dos clientes são genéricas o suficiente para que eles não precisem conhecer os detalhes dos subsistemas de pontos variáveis para poder utilizá-las.

6.4 Considerações Finais

Neste capítulo apresentou-se sucintamente uma aplicação realizada para avaliar o *framework* proposto, bem como o método de simulação. Mostraram-se também os resultados dessa aplicação. A aplicação implementa um método que simula o fluxo de uma peça em sistemas flexíveis de fabricação. Os resultados obtidos na simulação correspondem àqueles utilizados por especialistas do domínio. Como ocorreu a coincidência dos resultados avalia-se que o método proposto, o *framework* que o implementa e a aplicação que simula um sistema de fabricação específico são bastante promissores para apoiar o planejamento do processo de produção de uma peça em sistemas flexíveis de fabricação realizado pelos especialistas do domínio da manufatura. Entende-se também que novas investigações devem ser realizadas a fim de efetuar uma melhor avaliação do sistema.

7 Conclusões

7.1 Considerações Iniciais

Esta tese apresentou os conceitos de padrões, subsistemas de pontos variáveis e *frameworks*. Mostrou-se como se podem introduzir flexibilidades no desenvolvimento de software quando se empregam essas formas de reuso. Em seguida apresentou-se sucintamente o domínio da manufatura e em maior grau de detalhamento os subdomínios do planejamento do processo para produção de peças e do controle de sistemas flexíveis de fabricação. Dois *frameworks* desse subdomínio freqüentemente presentes na literatura foram discutidos e comparados entre si.

Um método de simulação do fluxo de peça em sistemas de fabricação baseado na geometria e na funcionalidade dos dispositivos que entram em contato direto com a peça durante o seu fluxo foi proposto. Em conjunto com o método apresentou-se o processo de modelagem de um sistema de fabricação e o modelo conceitual resultante. O modelo foi generalizado e transformado em um *framework* para captar as variabilidades do domínio da manufatura.

Finalmente, utilizou-se o *framework* para instanciar uma aplicação que simula o fluxo da peça em um sistema flexível de fabricação real. Compararam-se os resultados obtidos pelo sistema de simulação instanciado a partir do *framework* com aqueles utilizados pelos especialistas do domínio para a produção da peça. Devido à similaridade dos resultados avaliou-se que o método proposto e o *framework* que o implementa são bastante promissores e que novas investigações nesta linha de trabalho devem ser realizadas.

7.2 Contribuições

Como o trabalho é multidisciplinar, em uma parte as contribuições e as propostas de trabalhos futuros são em Engenharia de Software, em outra parte em Engenharia Mecânica e em uma terceira parte envolve as duas áreas.

Uma contribuição desta tese é um novo método de simulação do fluxo da peça em sistemas flexíveis de fabricação. O emprego do método resulta no planejamento macroscópico das operações realizadas sobre a peça durante o processo produtivo. Como o método é baseado na forma como o dispositivo atua sobre a peça, ele pode ser facilmente estendido ao incorporar outros tipos de dispositivos não modelados no escopo deste trabalho. Para cada

tipo de dispositivo criou-se um critério que qualifica a sua utilização e, por isso, o sistema pode oferecer automaticamente a melhor forma de realizar o fluxo da peça. A integração em um único sistema que simula todos os equipamentos e seus dispositivos, presentes no fluxo da peça, como realizado neste trabalho, garante a compatibilidade entre os resultados gerados para cada dispositivo.

Outra contribuição desta pesquisa é a modelagem de parte do domínio da manufatura, em especial dos dispositivos e da forma como eles atuam durante o fluxo de uma peça em um sistema de fabricação. Modelou-se tanto a atuação individual dos dispositivos sobre a peça, em função de suas características geométricas e funcionais, como também a atuação coletiva dos dispositivos no instante em que a peça é transferida entre eles. Apresentou-se não só o modelo conceitual mas também o processo de modelagem utilizado, o que permite que outros desenvolvedores possam utilizá-lo para modelar tanto outros dispositivos, como outros aspectos do domínio da manufatura. O modelo foi gerado sob o paradigma da orientação a objetos que facilita a sua reutilização na modelagem ou na implementação de sistemas computacionais relacionados.

O *framework* que implementa o modelo orientado a objetos do método de simulação proposto também é uma contribuição importante deste trabalho. É um *framework* inovador, desenvolvido em um subdomínio da manufatura em que não foram encontradas propostas de soluções baseadas em *frameworks*. O *framework* também foi utilizado como prova dos conceitos incorporados no método de simulação e no modelo conceitual desenvolvidos neste trabalho. Além do reuso do *framework* para o desenvolvimento de aplicações específicas deste domínio, todo o processo, desde a proposta do método, a sua modelagem e o projeto e a implementação do *framework*, pode ser reutilizado por engenheiros de software como um guia para desenvolvimentos semelhantes.

Um subproduto desta contribuição está nos estudos realizados para o desenvolvimento de *frameworks* que gerem aplicações modulares integradas por sistemas de persistência. A persistência das informações relativas aos pontos variáveis de um *framework* não é trivial. O acesso às informações armazenadas não é simples, pois em princípio elas mudam para cada possível implementação do ponto variável, de forma que não pode ser prevista pelos desenvolvedores do *framework*. As formas tradicionais de adaptação a essas alterações implicam em alterações do código base do *framework*. Contribuiu-se neste ponto com uma proposta de associação de padrões para que a forma de acesso a informações desconhecidas

em um banco de dados relacional fosse realizada de um modo genérico. Com isso, o código base do *framework* não precisa ser alterado a cada nova instanciação do ponto variável.

7.3 Trabalhos Futuros

A seguir apresentam-se algumas propostas de trabalhos futuros que complementam o desenvolvimento realizado nesta tese e que podem trazer novas contribuições ao estado da arte:

Inicialmente é recomendável que se instancie novas aplicações a partir do *framework* e que se implemente o Módulo de Escolha como previsto em sua arquitetura para melhor avaliar o método e a implementação realizada. Para realizar esta tarefa é necessário que se modele e implemente outros dispositivos utilizados nos equipamentos que constituem os sistemas flexíveis de fabricação. A definição de novos equipamentos permite que se modele e implemente novas conexões entre eles e novas estratégias de produção para peças com outras geometrias. Assim o espectro de sistemas de fabricação que podem ser simulados com este *framework* torna-se bem mais abrangente.

Outra linha de trabalho é constituída pela integração deste *framework* com outros sistemas. Pode-se integrá-lo ao modelador geométrico tridimensional, como os existentes em sistemas CAD (*Computer Aided Design*), para por um lado permitir uma modelagem mais complexa dos dispositivos e por outro melhorar a análise visual dos resultados da simulação. Uma forma de integração mais interessante pode ser obtida se for realizada com os sistemas que utilizem os resultados da simulação. Seria desejável efetuar a integração com sistemas de programação dos equipamentos para auxiliar o detalhamento das operações e com os *frameworks* para controle de sistemas flexíveis de fabricação.

Para solucionar o problema de persistência das informações referentes a pontos variáveis do *framework*, propôs-se uma associação de padrões que permite a implementação de novas variabilidades sem a alteração do código básico do *framework*. No escopo deste trabalho utilizou-se essa mesma associação de padrões para implementar a variabilidade requerida em outra parte do *framework*. Sugere-se que outros estudos sobre a associação de padrões sejam efetuados com o objetivo de formalizá-la como um novo padrão.

Um estudo mais amplo pode ser realizado no campo da simulação. Atualmente existem diversos sistemas de simulação em diferentes áreas do conhecimento. Propõe-se a realização de estudos para a generalização deste *framework* e de outros sistemas de simulação para avaliar a viabilidade do desenvolvimento de *frameworks* para simulações genéricas.

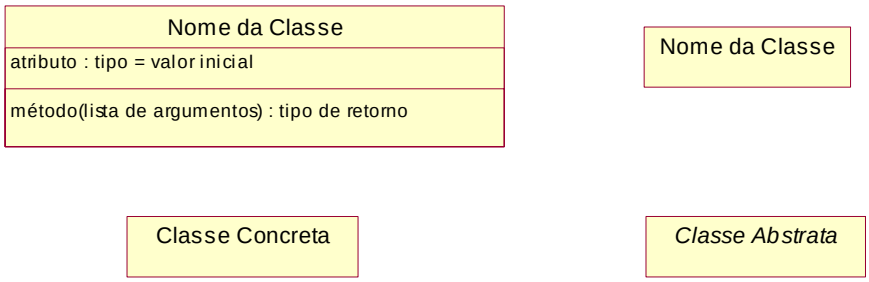
Anexo

UML

Unified Modeling Language

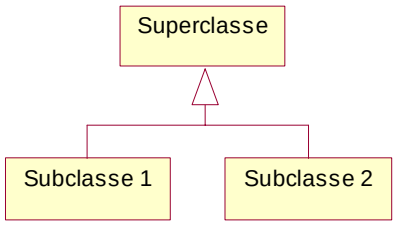
1 Classe

Uma classe é representada por um retângulo. O retângulo é dividido em três retângulos menores, sendo que o superior é utilizado para designar o nome, o intermediário os atributos e o inferior os métodos da classe. Pode-se simplificar a representação omitindo-se os atributos, os métodos ou ambos. Uma classe concreta é representada com letras normais e uma classe abstrata com letras em itálico.



2 Generalização

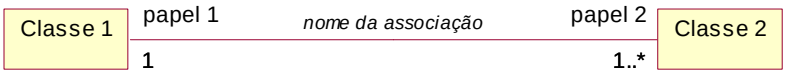
A generalização é uma relação entre classes, representada por um triângulo com um de seus vértices posicionado sobre a superclasse. Uma linha une a face oposta ao vértice do triângulo localizado sobre a superclasse até a subclasse. Quando uma superclasse possui mais de uma subclasse, normalmente, o triângulo e parte da linha de união até as subclasses são representados uma única vez.



3 Associação

A associação entre duas classes é representada por uma linha unindo-as. Pode-se denotar a associação com um nome específico. Os objetos das duas classes que participam da

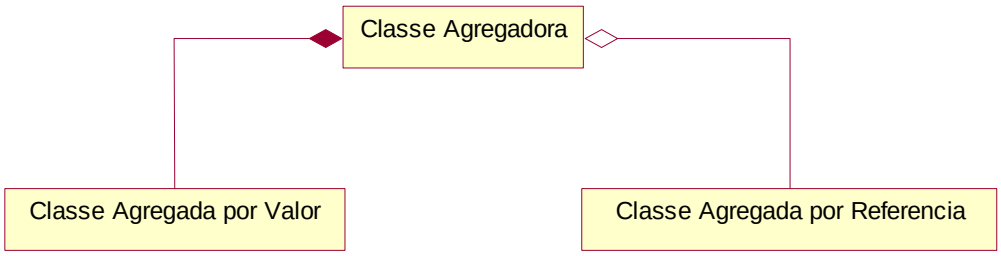
associação desempenham um determinado papel na associação, que pode ser descrito ao lado das classes próximo à linha que representa a associação. Pode-se especificar a quantidade de objetos de cada classe que participa da associação. Essa quantidade é denominada multiplicidade e é representada por um valor ou uma faixa de valores (dois valores separados por dois pontos). Quando uma associação por relacionar uma quantidade qualquer de objetos o valor é substituído por um asterisco (*).



As classes de uma associação podem ter visibilidade mútua ou apenas uma delas pode visualizar a outra. Quando a visibilidade é mútua ou indeterminada a associação é representa apenas pela linha que une as classes. Quando a visibilidade é em sentido único a associação é dita unidirecional ou denomina-se de sentido de navegabilidade da associação. Neste caso ela é representada por uma linha com uma seta na extremidade apontando da classe que visualiza para a classe que é vista. Esta representação indica que a classe que visualiza pode passar mensagens para a outra e, portanto, acessar os seus métodos. A classe visualizada em uma associação unidirecional apenas responde a requisições de outras classes. A nomenclatura de classe cliente e servidora são freqüentemente utilizadas para denominar respectivamente as classes que visualiza e que é vista na associação.

4 Agregação

A agregação é um tipo especial de associação entre classes. Ela representa o conceito de “todo – parte”. Os objetos da classe agregadora possuem objetos da classe agregada. A agregação pode ser por valor, quando o objeto agregado pertence a exatamente um único objeto agregador, ou por referência, quando um objeto agregado pode pertencer a um número qualquer de objetos agregadores. A agregação é representa por uma linha com um losango na extremidade junto à classe agregadora. Se a agregação é por valor, o losango deve ser preenchido, se for por referência ele deve ser vazio.

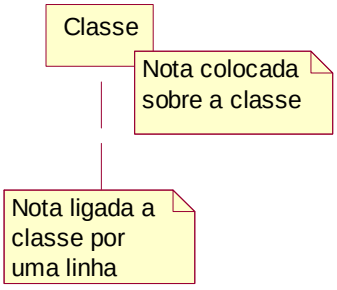


5 Dependência

A dependência é uma forma de relacionamento entre dois objetos. Quando um objeto sofre alteração o objeto dependente deve verificar se ele também precisa ser alterado. Dentre as formas de dependência existe a instanciação. Neste caso um objeto é responsável pela criação de objetos usando o mecanismo de instanciação da outra classe. A instanciação é representada por uma linha tracejada unindo as duas classes, com uma seta apontando para a classe instanciável.

6 Nota

A nota é representada por um retângulo em que canto superior direito tem o aspecto semelhante ao de uma folha de papel com a ponta dobrada. Ela é colocada diretamente sobre parte do retângulo que representa a classe ou unida a ela por uma linha traceja, denominada ancora da nota à classe, até a informação contida na classe que se pretende esclarecer ou detalhar.



Referências Bibliográficas

AARSTEN, A.; BRUGALI, D.; MENGA, G. (1996a). Designing Concurrent and Distributed Control Architectures. *Communications of the ACM*, v. 39, n. 10, p. 50-58, Oct.

AARSTEN, A.; BRUGALI, D.; MENGA, G. (2000). A CIM Framework and Pattern Language. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 3, p. 21-42.

AARSTEN, A.; ELIA, G.; MENGA, G. (1995). G++: A Pattern Language for Computer-Integrated Manufacturing. In: COPLIEN, J.O.; SCHMIDT, D.C., eds. (1995). *Pattern Languages of Program Design.* Reading-MA: Addison-Wesley. Cap. 6, p. 91-118.

AARSTEN, A.; MENGA, G.; MOSCONI, L. (1996b). Object-Oriented Design Patterns in Reactive Systems. In: VLISSIDES, J.M.; COPLIEN, J.O.; KERTH, N.L., eds. (1996). *Pattern Languages of Program Design 2*. Reading-MA: Addison-Wesley. Cap. 32, p. 537-548.

ALEXANDER, C.; ISHIKAWA, S.; SILVERSTEIN, M.; JACOBSON, M.; FIKSDAHL-KING, I.; ANGEL, S. (1977). *A Pattern Language*. New York: Oxford University Press.

APPLETON, B. (1997). *Patterns and Software:* Essential Concepts and Terminology. Disponível em <http://www.enteract.com/~bradappdocpatterns-intro.html>. Acesso em 9 out. 2000.

AUER, H.B. (1977). *Beitrag zur Steigerung der Flexibilität von Handhabungseinrichtungen im Bereich der Einzel- und Kleinserienfertigung*. Berlin. 199 p. Dissertation (Doktor-Ingenieur) – Technische Universität Berlin.

BANKS, J.; CARSON, J.S.; NELSON, B.L. (1996). *Discrete-Event System Simulation*. 2.ed. Englewood Cliffs-NJ: Prentice Hall.

BLOKSMA, R.G. (1988). Palettensysteme für die integrierte Fertigung: Metallwerkstückträger, die Grundvoraussetzung für automatisiertes, funktionssicheres Teile-Handling. In: EUROPÄISCHE KONGREßMESSE FÜR TECHNISCHE AUTOMATION, 5. "kommtech 88", Essen, 7.-10. Juni. *Proceedings...*, v. 1, Cap. 5.4.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. (2000). *UML:* Guia do Usuário. Rio de Janeiro: Campos.

BORLAND INTERNATIONAL INC. (1997). *Borland C++ Builder for Windows 95 & Windows NT:* User's Guide. Scotts Valley-CA: [s.n.].

BOSCH, J. (2000). Measurement Systems Framework. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 11, p. 177-205.

BOSCH, J.; MOLIN, P.; MATTSSON, M. BENGTSSON, P.; FAYAD, M. (1999). Framework Problems and Experiences. In: FAYAD, M.E.; SCHMIDT, D.C.; JOHNSON, R.E., eds. (1999). *Building Application Frameworks:* Object-Oriented Foundations of Framework Design. New York: John Wiley & Sons. Cap. 3, p. 55-82.

BOYLE, W. (2000). Distributed Manufacturing Execution Systems Framework. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 8, p. 121-138.

BRUGALI, M. (1998). *From Objects to Agents:* Software Reuse for Distributed Systems. Torino Italia. 108 p. Dissertation (Doctor of Philosophy) - Politecnico di Torino. Disponível em <http://www.polito.it/~brugali>. Acesso em 3 maio 2001.

BRUGALI, D.; MENGA, G. (1999). Framework and Pattern Languages: An Intriguing Relationship. *ACM Computing Surveys*. Disponível em <http://www.polito.it/~brugali>. Acesso em 3 maio 2001.

BRUGALI, D.; MENGA, G.; AARSTEN, A. (1997). The Framework Life Span. *Communications of the ACM*, v. 40, n. 10, p. 65-68, Oct.

BRUGALI, D.; MENGA, G.; AARSTEN, A. (2000). A Case Study for Flexible Manufacturing Systems. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 6, p. 85-99.

BÜNNING, T. (1990). Kein Umfüllen mehr am Arbeitzplatz: Materialträgersysteme für manuelle und automatische Fertigung. *Industrie-Anzeiger*, v. 112, n. 28, p. 10-12.

BUSCHMANN, F.; MEUNIER, R.; ROHNERT, H.; SOMMERLAD, P.; STAL, M. (1996). *Pattern-Oriented Software Architecture:* A system of Patterns. Chichester, West Sussex, England: John Wiley & Sons.

BUSCHULTE, J. (1990). Wirrwarr beim Greiferwechsel: Standardschnittstelle für Roboter und Greifer. *Roboter*, n. 1, p. 37-40, Feb.

CARDAUN, U. (1981). *Systematische Auswahl von Greiferkonzepten für die Werkstückhandhabung*. Hannover. Dissertation (Doktor-Ingenieur) – Universität Hannover.

CHAM, S.M.; LAMMERS, T.L. (2000). Developing Domain Frameworks. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 10, p. 159-176.

COAD, P. (1992). Object-Oriented Patterns. *Communications of the ACM*, v. 35, n. 9, p. 152-159, Sept.

COAD, P.; NORTH, D.; MAYFIELD, M. (1995). *Object Models:* Strategies, Patterns and Applications. Upper Saddle River-NJ: Prentice-Hall.

COAD, P.; NORTH, D.; MAYFIELD, M. (1997). *Object Models:* Strategies, Patterns and Applications, 2.ed. Upper Saddle River-NJ: Prentice-Hall.

COPLIEN, J.O.; SCHMIDT, D.C., eds. (1995). *Pattern Languages of Program Design*. Reading-MA: Addison-Wesley.

DEUTSCHES INSTITUT FÜR NORMUNG (1986). *DIN Norm 15141 Teil 1 – Paletten:* Formen und Hauptmaße von Flachpaletten. Berlin: Beuth Verlag, Jan. 3 p.

DEUTSCHES INSTITUT FÜR NORMUNG (1986). *DIN Norm 24601 Entwurf - Mechanische Schnittstelle Form A, rund*. Berlin: Beuth Verlag, Dez.

DEUTSCHES INSTITUT FÜR NORMUNG (1989). *DIN Norm 30781 Teil 1 – Transportkette:* Grundbegriffe. Berlin: Beuth Verlag, Mai. 3 p.

DIE SACHE mit der Ordnung! Werkstückträger in der flexiblen Fertigung und Montage (1989). *Flexible Fertigung*, v. 7, n. 1, p. 113-116.

DIETRICH, W.C. *et al*. (2000). Production Resource Manager (PRM) Framework. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 9, p. 139-158.

DILLING, H.J. (1978). *Methodisches Rationalisieren von Fertigungsprozessen am Beispiel montagegerechter Produktgestaltung*. Darmstadt. 250 p. Dissertation (Doktor-Ingenieur) – Fachbereich Maschinenbau, Technische Hochschule Darmstadt.

DOSCHER, D.; HODGES, R. (1997). SEMATECH's Experiences with the CIM Framework. *Communications of the ACM*, v. 40, n. 10, p. 82-84, Oct.

DOSCHER, D.; HODGES, R. (2000). SEMATECH CIM Framework. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 2, p. 7-19.

DREHER, H.; WEISENER, T. (1990). Greifbarer Boom: Große Steigerungen auf dem Greifermarkt. *Roboter*, n. 5, p. 55-60, Okt.

ENZLER, H. (1989). System-Werkstückträger: Einführung in die Problematik. In: HANDBUCH: System-Werkstückträger für durchgehende Ordnung in flexibler Fertigung und Montage. Düsseldorf: VDI-Verlag. Cap. 1. VDI-Seminar. Stuttgart, 13. und 14. März.

ERIKSSON, H.; PENKER, M. (1998). *UML Toolkit*. New York: Wiley Computer Publishing.

FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons.

FAYAD, M.E.; SCHMIDT, D.C. (1997). Object-Oriented Application Framework. *Communications of the ACM*, v. 40, n. 10, p. 32-38, Oct.

FAYAD, M.E.; SCHMIDT, D.C.; JOHNSON, R.E. (1999). Application Frameworks. In: FAYAD, M.E.; SCHMIDT, D.C.; JOHNSON, R.E., eds. (1999). *Building Application Frameworks:* Object-Oriented Foundations of Framework Design. New York: John Wiley & Sons. Cap. 1, p. 3-27.

FOWLER, M. (1997). *Analysis Patterns:* Reusable Object Models. Menlo Park-CA: Addison-Wesley.

FOWLER, M.; SCOTT, K. (1998). *UML Distilled:* Applying the standard object modeling language. Reading-MA: Addison-Wesley.

FORKARDT (1983). FORDSAVE-D-System. Druckschrift Nr. 620.01 der Firma Paul Forkardt GmbH & Co. KG. Düsseldorf. (prospecto)

FREITAS Fo., P.J. (2001). *Introdução à Modelagem e Simulação de Sistemas*: Com Aplicações em Arena. Florianópolis-SC: Visual Books.

FRICKE, W. (1985). *Rechnergestützte Planung von Übergabesystemen zwischen Transport und Fertigung*. Dortmund. 143 p. Dissertation (Doktor-Ingenieur) – Universität Dortmund.

GALL, H.; KLÖSH, R.R.; MITTERMEIER, R.T. (1996). Application Patterns in Re-engineering: Identifying and Using Reusable Concepts. INTERNATIONAL CONFERENCE ON INFORMATION PROCESSING AND MANAGEMENT OF UNCERTAINTY IN KNOWLEDGE-BASED SYSTEMS, 6. "IPMU", July. *Proceedings...*, v. 3, p. 1099-1106.

GAMMA, E.; HELM, R.; JOHNSON, R.E.; VLISSIDES, J. (1995). *Design Patterns:* Elements of Reusable Object-Oriented Software. Reading-MA: Addison-Wesley.

GANIYUSUFOGLU, Ö.S. (1984). Materialbereitstellungssysteme für flexible Fertigungszellen. *Zeitschrift für wirtschaftliche Fertigung*, v. 79, n. 4, p. 159-170, Apr.

GANIYUSUFOGLU, Ö.S. (1985). *Wirtschaftliche Nutzung flexibler Fertigungszellen am Beispiel der Drehbearbeitung*. Berlin. 139 p. Dissertation (Doktor-Ingenieur) – Technische Universität Berlin.

GANIYUSUFOGLU, Ö.S. (1988). Flexible Drehzellen als autonome Fertigungsinseln. In: EUROPÄISCHE KONGREßMESSE FÜR TECHNISCHE AUTOMATION, 5. "kommtech 88", Essen, 7.-10. Juni. *Proceedings...*, v. 1, Cap. 3.3.

GERTOSIO, C.; MEBARKI, N.; DUSSAUCHOY, A. (2000). Modeling and Simulation of the Control Framework on a Flexible Manufacturing System. *International Journal of Production Economics*, n. 64, p. 285-293.

GOLDBERG, A.; ROBSOND. (1983). *Smaltalk80*: The language and its Implementation. Reading-MA: Addison-Wesley.

GRAF, B. (1984). *Flexibilität und Kapazität von Werkstückspeichersystemen*. Stuttgart. 115 p. Dissertation (Doktor-Ingenieur) – Fakultät Fertigungstechnik, Universität Stuttgart.

HAHNER, W. (1989). Werkstücke und Werkzeuge sicher und präzise spannen. *Werkstatt und Betrieb*, v. 122, n. 12, p. 1067-1070.

HALEVI, G.; WEILL, R.D. (1995). *Principles of Process Planning*: A Logical Approach.1.ed. London: Chapman & Hall.

HARDOCK, G. (1988). Werkstückträger als Systemfamilie. *Roboter*, n. 3, p. 32-38.

HESSE, S.; MITTAG, G. (1989). *Handhabetechnik:* Technische Lösung für Konstrukteure. Heidelberg: Hüthig Verlag.

HIBBELER, R.C. (1999). *Mecânica*. Vol. 2: Dinâmica. Rio de Janeiro: Ed. LTC.

IBM (1986). CATIA - Computerunterstütztes graphisch interaktives dreidimensionales Anwendungssystem. Druckschrift Nr. GT12-3002-4 der Firma IBM Deutschland GmbH. Stuttgart. (Catálogo)

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (1988). *ISO 9409-1 - Manipulating industrial robots:* Mechanical interfaces - Part 1: Circular (form A). 15 Dec. 3 p.

JACOBSON, E.E.; NOWACK, P. (1999). Frameworks and Patterns: Architectural Abstractions. In: FAYAD, M.E.; SCHMIDT, D.C.; JOHNSON, R.E., eds. (1999). *Building Application Frameworks:* Object-Oriented Foundations of Framework Design. New York: John Wiley & Sons. Cap. 2, p. 29-54.

JOHNSON, R.E. (1997). Framework = (Components + Patterns). *Communications of the ACM*, v. 40, n. 10, p. 39-42, Oct.

JOHNSON, R.E.; FOOTE, B. (1988). Designing Reusable Classes. *Journal of Object-Oriented Programming*, v. 1, n. 5, p. 22-35, June/July.

KASTO MASCHINENBAU GMBH (2002). The Name for Economic Cutting and Storing of Metal. Disponível em <http://www.kasto.de/e/index.html>. Acesso em 21 de agosto de 2002.

KELLER, V. (1989). Werkstückträger aus Stahl und Aluminium. In: HANDBUCH: System-Werkstückträger für durchgehende Ordnung in flexibler Fertigung und Montage. Düsseldorf: VDI-Verlag. Cap. 8. VDI-Seminar. Stuttgart, 13. und 14. März.

KELLER, W; COLDEWEY, J. (1998). Accessing Relational Databases. In: MARTIN, R.C.; RIEHLE, D.; BUSCHMANN, F., eds. (1998). *Pattern Languages of Program Design 3*. Reading-MA: Addison-Wesley. Cap. 18, p. 313-343.

KOGAN, K.; KHMELNITSKY, E.; SHTUB, A.; MAIMON, O. (1997). Optimal Flow Control of Flexible Manufacturing Systems: Setup Localization by an Interactive Procedure. *International Journal of Production Economics*, n. 51, p. 37-46.

KUK, K.H. (1988). *Modulares Simulationsmodell für die Abläufe in verketteten Fertigungszellen mit Industrierobotern*. Stuttgart. 130 p. Dissertation (Doktor-Ingenieur) – Universität Stuttgart.

LAW, A.M.; KELTON, W.D. (1991). *Simulation Modeling and Analysis*. 2.ed. New York: McGraw-Hill.

LIPPMAN, S.B.; LAJOIE, J. (1998). *C++ Primer*. 3.ed. Reading-MA: Addison-Wesley.

MARTIN, R.C.; RIEHLE, D.; BUSCHMANN, F., eds. (1998). *Pattern Languages of Program Design 3*. Reading-MA: Addison-Wesley.

MASSAROPPI JR., E. (1998). *Monografia sobre o Artigo: "Creating Applications from Components: A Manufacturing Framework Design"*. São Carlos. 16 p. Monografia (Exame de Qualificação) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.

MASSAROPPI JR., E. (2002). *TornoFix*: Um Framework para Simulação do Fluxo de Peças: Guia do Usuário.

MASSAROPPI JR., E.; MASIERO, P.C. (2001a). Modelagem Geométrica e Funcional de Placas de Torno para Auxiliar a Determinação do Fluxo de Peças em Sistemas Produtivos. In: CONGRESSO BRASILEIRO DE ENGENHARIA DE FABRICAÇÃO, 1., Curitiba-PR. *Anais...*, 1 CD, 10 p.

MASSAROPPI JR., E.; MASIERO, P.C. (2001b). Fluxo de Peça Determinado pela Modelagem de Placas de Tornos. *Máquinas e Metais*, A. 38, n. 429, p. 100-111, Out.

MASSAROPPI JR., E.; MASIERO, P.C. (2002). Modelagem Geométrica e Funcional de Garras de Robôs para Auxiliar a Determinação do Fluxo de Peças em Sistemas Produtivos. In: CONGRESSO NACIONAL DE ENGENHARIA MECÂNICA, 2., João Pessoa-PB. *Anais...*, 1 CD, 10 p.

MASSAROPPI JR., E.; MASIERO, P.C. (2003). Modelagem Geométrica e Funcional de Paletes de Transporte para Auxiliar a Determinação do Fluxo de Peças em Sistemas Produtivos. In: CONGRESSO BRASILEIRO DE ENGENHARIA FABRICAÇÃO, 2., Uberlândia-MG. 11 p. (a ser publicado).

MATTSSON, M.; BOSCH, J.; FAYAD, M.E. (1999). Framework Integration: Problems, Causes, Solutions. *Communications of the ACM*, v. 42, n. 10, p. 81-87, Oct.

MICHAELIS, D. (1982). *Rechnerunterstützte Konstruktion von Funktionssystemen zur flexiblen Handhabung rotationssymmetrischer Werkstücke*. Berlin. 187 p. Dissertation (Doktor-Ingenieur) – Technische Universität Berlin.

MICHAELIS, D. (1984). Rechnerunterstüzte Planung der Einspannung, der Bereitstellung und des Greifens von Drehteilen. *HGF-Kurzbericht*, n. 23.

MONDAY, P.; CAREY, J.; DANGLER, M. (2000). *SanFrancisco Component Framework:* An Introduction. Reading-MA: Addison-Wesley.

MOSER, O. (1991). *3D-Echtzeitkollisionsschutz für Drehmaschinen*. München. 165 p. Dissertation (Doktor-Ingenieur) – Fakultät für Maschinenwesen, Technische Universität München.

NIEMANN. G. (1981). *Maschinenelemente*. Band 1. Berlin: Springer Verlag.

NOSKE, H. (1991a). Spannkraftüberwachung an Drehmaschinen: Meßsystem zur Erfassung der dynamischen Spannkraft. *VDI-Zeitschrift*, v. 133, n. 6, p. 113-121, Juni.

NOSKE, H. (1991b). Spannkraftüberwachung an Drehmaschinen: Reibzustandüberwachung von Spannfuttern. *VDI-Zeitschrift*, v. 133, n. 7, p. 71-74, Juli.

OMNI (2000). OMNISPHERE INFORMATION SYSTEMS CORPORATION. OmniBuilder: Open Application Generator. Disponível em <http://www.omni-sphere.com/overview/overview.htm>. Acesso em 3 maio 2001.

OSTERWINTER, M. (1991). *Steuerungsorientiertes Simulationssystem zum Test von Industrieroboterapplikationen*. Aachen. 129 p. Dissertation (Doktor-Ingenieur) – Fakultät für Maschinenwesen, RWTH-Aachen.

PHAM, D.T.; HEGINBOTHAM, W.B. (1986). *Robot Grippers*. Berlin: Springer Verlag.

PREE, W. (1995). *Design Patterns for Object-Oriented Software Development*. Reading-MA: Addison-Wesley.

PREE, W. (1999). Hot-Spot-Driven Development. In: FAYAD, M.E.; SCHMIDT, D.C.; JOHNSON, R.E., eds. (1999). *Building Application Frameworks:* Object-Oriented Foundations of Framework Design. New York: John Wiley & Sons. Cap. 16, p. 379-393.

QUADROS, E.M. (1997). *Uma Abordagem Orientada a Objetos para Programação Distribuída Confiável*. Campinas. 120 p. Dissertação (Mestrado em Ciência da Computação) – Instituto de Computação, Universidade Estadual de Campinas.

QUADROS, E.M.; RUBIRA, C.M.F. (1997). Construção de um Framework para Sistemas Controladores de Trens Utilizando Padrões de Projeto e Metapadrões. In: SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE, 11., Fortaleza-CE *Anais*... Fortaleza-CE: Universidade Federal do Ceará.

RATIONAL CORPORATION (2000). *Unified Modeling Language*. Disponível em <http://www.rational.com/uml/resources/documentation/index.jsp>. Acesso em 3 maio 2001.

RITTINGHAUSEN, H. (1980). *Integrierte Materialflußautomatisierung in der Einzel- und Serienfertigung*. Berlin. 159 p. Dissertation (Doktor-Ingenieur) – Technische Universität Berlin.

RITTINGHAUSEN, H.; SINNING, H. (1979). Flexible Materialbereitstellung mit Handhabungsgeräten. Teil 1: *HGF-Kurzbericht*, n. 34, Teil 2: *HGF-Kurzbericht*, n. 36.

ROBERTS, D.; JOHNSON, R.E. (1998). Patterns for Evolving Frameworks. In: MARTIN, R.C.; RIEHLE, D.; BUSCHMANN, F., eds. (1998). *Pattern Languages of Program Design 3*. Reading-MA: Addison-Wesley. Cap. 26, p. 471-486.

ROSENBAUER, T. (1988). Work Piece Specific Interfaces Determine the Flexibility of FMS. In: *Integrated European Course in Mechatronics*. 10.-14. Oktober. Aachen: RWTH-Aachen.

ROZENFELD, H. (1992). *Implantação Distribuída do Planejamento de Processo Assistido por Computador na Manufatura Integrada*. São Carlos. 128 p. Tese (Livre Docência) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

RÜSTEN in der Zelle: Werkstück-Trägersysteme für den Weg zu CIM. (1990). *Flexible Automation*, v. 8, n. 4, p. 39-41.

SCHMID, H.A. (1995). Creating the Architecture of a Manufacturing Framework by Design Patterns. *ACM Sigplan Notices*, v. 30, n. 10, p. 370-384, Oct.

SCHMID, H.A. (1996a). Design Patterns for Constructing the Hot Spots of a Manufacturing Framework. *Journal of Object-Oriented Programming*, v. 9, n. 3, p. 25-37, June.

SCHMID, H.A. (1996b). Creating Applications from Components: A Manufacturing Framework Design. *IEEE Software*, v. 13, n. 6, p. 67-75, Nov.

SCHMID, H.A. (1997). Systematic Framework Design by Generalization. *Communications of the ACM*, v. 40, n. 10, p. 48-51, Oct.

SCHMID, H.A. (1999). Framework Design by Systematic Generalization. In: FAYAD, M.E.; SCHMIDT, D.C.; JOHNSON, R.E., eds. (1999). *Building Application Frameworks:* Object-Oriented Foundations of Framework Design. New York: John Wiley & Sons. Cap. 15, p. 353-378.

SCHMID, H.A. (2000a). OSEFA: Framework for Manufacturing. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 4, p. 43-65.

SCHMID, H.A. (2000b). Framework Reuse over Different CIM Subdomains. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 5, p. 67-84.

SCHMID, H.A.; MUELLER, F. (1998). Patterns for Extending Black-Box Frameworks. *Journal of Object-Oriented Programming*, v. 11, n. 3, p. 38-47, June.

SCHULER, J. (1987). *Integration von Förder- und Handhabungseinrichtungen*. Stuttgart. 153 p. Dissertation (Doktor-Ingenieur) – Fakultät Fertigungstechnik, Universität Stuttgart.

SCHULER, J.; HARDOCK, G. (1986). Standardisierte Magazinpalette für flexible Fertigung: Handhabungsgerechte Werkstückbereitstellung. *Industrie-Anzeiger*, v. 108, n. 5, p. 18-21.

SCHUNK Spann- und Greiftechnik. (1999). *Gripping Systems*: Schunk offers more! (Catálogo)

SCHÜTZE, P. (1988). *Kollisionskontrolle bei der Offline-Programmierung von Industrierobotern*. Aachen. 155 p. Dissertation (Doktor-Ingenieur) - Fakultät für Maschinenwesen, RWTH-Aachen.

SEVERIN, F. (1987). *Planung der Flexibilität von roboterintegrierten Bearbeitungs- und Montagezellen*. Berlin. 143 p. Dissertation (Doktor-Ingenieur) – Technische Universität Berlin.

SIEMENS, K.J. (1983). *Konstruktive Lösungswege zur Erhöhung der Flexibilität von Werkzeugen für Handhabungsgeräte*. Hannover. 127 p. Dissertation (Doktor-Ingenieur) – Institut für Fertigungstechnik und Spanende Werkzeugmaschinen, Universität Hannover.

SPUR, G.; ABELEIN, G.; STELZER, C. (1993). Force-Controled Clamping Using Three-Jaw Chucks. *Production Engineering*, v.1, n. 1, p. 111-116.

SPUR, G.; METTE, U. (1998). Clamping-Force Optimization Allows High-Speed Turning. *Production Engineering*, v. 5, n. 1, p. 55-58.

SPUR, G.; SEVERIN, F. (1982). Analyse und Entwicklung flexibler Greifereinrichtungen für Handhabungsgeräte. *Zeitschrift für wirtschaftliche Fertigung*, v. 77, n. 2, p. 75-81, Feb.

STAVE, H. (1988). Grippers and Mechanical Interfaces for Robot Systems. In: *Integrated European Course in Mechatronics*. 10.-14. Oktober. Aachen: RWTH-Aachen.

STEINBERGER, J. (1989). Richtig Spannen beim Hochgeschwindigkeits-Drehen. *Werkstattstechnik*, v. 79, n. 9, p. 521-525.

STÖCK, H.P. (1986). *On-line Kollisionsvermeidung bei Handhabungssystemen:* Ein Beitrag zur Steigerung der Einsatzsicherheit bei Handhabungsgeräten. Aachen. 132 p. Dissertation (Doktor-Ingenieur) – Fakultät für Maschinenwesen, RWTH-Aachen.

STOLZ, A.; BOLEY, D. (1988). Wechselspiel am Greiferflansch. *Roboter*, n. 2, p. 68-76.

TALIGENT INC. (1994). Building Object-Oriented Frameworks. A Taligent White Paper, Disponível em <http://www.ibm.com/java/education/oobuilding/index.html>. Acesso em 7 maio 1998.

TALIGENT INC. (1995). Levereging Object-Oriented Frameworks. A Taligent White Paper, Disponível em <http://www.ibm.com/java/education/oolevearing/index.html>. Acesso em 7 maio 1998.

THIM, C. (1992). *Rechnerunterstützte Optimierung von Materialflußstrukturen in der Elektronikmontage durch Simulation*. Nürnberg. 169 p. Dissertation (Doktor-Ingenieur) – Technischen Fakultät der Friedrich-Alexander, Universität Erlangen-Nürnberg.

TONSHOFF, H.K., KREUTZFELDT, J.; HOFSCHNEIDER, D. (1992). Concurrent Process Planning and Workshop Control in Batch Production. CIRP International Seminar on Manufacturing Systems, 24. Copenhagen. p. 315-328.

UETZ, H.; SCHULER, J. (1984). Modulare Handhabungsgeräte und Werkstückmagazine: Schlüsselelemente für die flexibel automatisierte Rotationsteilebearbeitung. *VDI-Zeitschrift*, v. 126, n. 19, p. 687-694, Okt.

VEREIN DEUTSCHER INGENIEURE (1970). *VDI-Richtlinie 2411 - Begriffe und Erläuterungen im Förderwesen*. Düsseldorf: Beuth Verlag, Juni. 26 p.

VEREIN DEUTSCHER INGENIEURE (1969). *VDI-Richtlinie 2496 - Stahlpalette*. Düsseldorf: Beuth Verlag, Okt. 3 p.

VEREIN DEUTSCHER INGENIEURE (1991). *VDI-Richtlinie 2740 Blatt 1 Entwurf - Mechanische Einrichtungen in der Automatisierungstechnik:* Greifer für Handhabungsgeräte und Industrieroboter. Düsseldorf: Beuth Verlag, Juli. 47 p.

VEREIN DEUTSCHER INGENIEURE (1990). *VDI-Richtlinie 2860 - Montage- und Handhabungstechnik:* Handhabungsfunktionen, Handhabungseinrichtungen; Begriffe, Definitionen, Symbole. Düsseldorf: Beuth Verlag, Mai. 16 p.

VEREIN DEUTSCHER INGENIEURE (1983). *VDI-Richtlinie 3106 - Ermittlung der zulässigen Drehzahl von Drehfuttern (Backenfuttern)*. Düsseldorf: Beuth Verlag, Juli. 8 p.

VEREIN DEUTSCHER INGENIEURE (1973). *VDI-Richtlinie 3300 - Materialfluß-Untersuchungen*. Düsseldorf: Beuth Verlag, Aug. 11 p.

VEREIN DEUTSCHER INGENIEURE (1983). *VDI-Richtlinie 3633 - Anwendung der Simulationstechnik zur Materialflußplanung*. Düsseldorf: Beuth Verlag, Mär. 15 p.

VEREIN DEUTSCHER INGENIEURE (1989). *VDI-Richtlinie 3961 - Planung der Materialflußsteuerung in Fertigungsbetrieben*. Düsseldorf: Beuth Verlag, Sept. 24 p.

VIEHWEGER, B. (1980). Auslegung von flexiblen Werkstückträgersystemen. *HGF-Kurzbericht*, n. 72.

VIEHWEGER, B. (1981). Flexible Transportvorrichtungen für scheibenförmige Werkstücke. *HGF-Kurzbericht*, n. 29.

VLISSIDES, J.M.; COPLIEN, J.O.; KERTH, N.L., eds. (1996). *Pattern Languages of Program Design 2*. Reading-MA: Addison-Wesley.

WAGNER, H.D. (1987a). Ermittlung der Betriebsspannkraft von Drehfuttern unter Berücksichtigung der Systemsteifigkeiten. *HGF-Kurzbericht*, n. 37.

WAGNER, H.D. (1987b). *Spannkraftverluste beim Drehen unter Beachtung der radialen Verformung und Steifigkeit von Werkstücken*. Darmstadt. 160 p. Dissertation (Doktor-Ingenieur) – Institut für Spanende Technologie und Werkzeugmaschinen, Technische Hochschule Darmstadt.

WARNECKE, H.J.; SCHRAFT, R.D. (1991). *Handbuch Handhabungs-, Montage- und Industrierobotertechnik*. Landsberg am Lech: Verlag Moderne Industrie.

WECK, M. (1991). *Werkzeugmaschinen:* Fertigungssysteme. Band 1: Maschinenarten, Bauformen und Anwendungsbereiche. 4.ed. Düsseldorf, Alemanha: VDI-Verlag. (Studium und Praxis).

WECK, M. *et al*. (1987). Bausteine Flexibler Fertigunssysteme. In: *Produktionstechnik: Auf dem Weg zu Integrierten Systemen*. In: Eversheim, W. *et al*., eds. (1987) *Produktionstechnik:* Auf dem Weg zu Integrierten Systemen. (Aachener Werkzeugmaschinen-Kolloquium, Mai) Düsseldorf, Alemanha: VDI-Verlag, Cap. 2.1, p. 119-164.

WECK, M. *et al*. (1990). Realisierung des Materialflusses in Integrierten Systemen: das Schnittstellenproblem. In: WECK, M. *et al*., eds. (1990) *Wettbewerbsfaktor Produktionstechnik*. (Aachener Werkzeugmaschinen-Kolloquium, Mai) Düsseldorf, Alemanha: VDI-Verlag, Cap. 3.2, p. 309-347.

WECK, M.; BEER, C. (1991). Kein Griff in die Kiste: Flexibel automatisierte Werkstückträgermontage. *Roboter*, n. 1, p. 62-64, Feb.

WECK, M.; GOEDEKE, G.G. (1987). Integriertes Fertigungs- und Montagesystem IFMS. *VDI-Zeitschrift*, v. 129, n. 8, p.10-17, Aug.

WECK, M.; KOHEN, E. (1986). Simulation als Hilfsmittel für den Aufbau und Betrieb von flexiblen Fertigungssystemen. In: PRITSCHOW, G.; SPUR, G.; WECK, M., eds. (1986) *Simulationstechnik in der Fertigung*. München: Carl Hanser Verlag. Cap. 11, p. 129-141.

WECK, M.; ROSENBAUER, T. (1988). Automatisierte Werkstückversorgung in FFS: Greifer, Spannvorrichtungen und Transportkomponenten. *Industrie-Anzeiger* v. 110, n. 7, p. 32-36.

WECK, M.; STÖCK, H.P. (1985). Kollisionsvermeidung bei Industrierobotern. *VDI-Zeitschrift*, v. 127, n. 3, p. 71-79, Feb.

WECK, M.; WEEKS, J.K. (1991). Planning and Programming Robot Actions in Space. In: *Space Course*, p. 70.1-70.19. Aachen: RWTH-Aachen. (apostila)

WEEKS, J.K. (1988). Aufgabenorientierte Roboterprogrammierung für die flexibel automatisierte Montage. *Industrie-Anzeiger*, v. 110, n. 13, p. 32-33.

WERKSTÜCKTRÄGER für die Automatisierung (1988). *Roboter*, n. 1, p. 14-16.

WRBA, P. (1990). *Simulation als Werkzeug in der Handhabungstechnik*. München. 169 p. Dissertation (Doktor-Ingenieur) – Fakultät für Maschinenwesen, Technische Universität München.

YANG, D.S.; MEHTA, U. (2000). CEF: A Concurrent Engineering Framework. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 7, p. 105-119.

YASSIN, A.; FAYAD, M. (2000). Application Frameworks: A Survey. In: FAYAD, M.E.; JOHNSON, R.E., eds. (2000). *Domain-Specific Application Frameworks:* Frameworks Experience by Industry. New York: John Wiley & Sons. Cap. 29, p. 615-632.

YODER, J.W.; JOHNSON, R.E.; WILSON, Q.D. (1998). Connecting Business Objects to Relational Databases. In: CONFERENCE ON THE PATTERN LANGUAGES OF PROGRAMS, 5., Monticello-IL, Aug. *Proceedings...* Disponível em <http://jerry.cs.uiuc.edu/~plop/plop98/final_submissions/>. Acesso em 3 maio 2001.

ZHAO, L.; FOSTER, T. (1998). A Pattern Language of Transport Systems: Point and Route. In: MARTIN, R.C.; RIEHLE, D.; BUSCHMANN, F., eds. (1998). *Pattern Languages of Program Design 3*. Reading-MA: Addison-Wesley. Cap. 23, p. 409-430.

ZIPSE, T. (1987). *Konzeption und Auswahl modularer Magazinpaletten*. Stuttgart. 126 p. Dissertation (Doktor-Ingenieur) – Fakultät Fertigungstechnik, Universität Stuttgart.