ESTE EXEMPLAR CORRESPONDE A REDAÇÃO FINAL DA TESE DEFENDIDA POR MAURO FERREIRA KOYAMA E APROVADA PELA COMISSÃO JULGADORA EM 06/03/2001
ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA

Arquitetura de Supervisão e Controle de Chão de Fábrica baseada em Componentes Genéricos

Autor: **Mauro Ferreira Koyama**
Orientador: **João Maurício Rosário**
Co-orientador: **Marcius Fabius H. de Carvalho**

23/01

UNICAMP
BIBLIOTECA CENTRAL

UNICAMP
BIBLIOTECA CENTRAL

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

Arquitetura de Supervisão e Controle de Chão de Fábrica baseada em Componentes Genéricos

Autor: **Mauro Ferreira Koyama**
Orientador: **João Maurício Rosário**
Co-orientador: **Marcius Fabius Henriques de Carvalho**

Curso: Engenharia Mecânica
Área de Concentração: Projeto Mecânico

Tese de doutorado apresentada à comissão de Pós Graduação da Faculdade de Engenharia Mecânica, como requisito para obtenção do título de Doutor em Engenharia Mecânica.

Campinas, 2001
S.P.- Brasil

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

K849a

Koyama, Mauro Ferreira
Arquitetura de supervisão e controle de chão de fábrica baseada em componentes genéricos / Mauro Ferreira Koyama. --Campinas, SP: [s.n.], 2001.

Orientadores: João Maurício Rosário, Marcius Fabius Henriques de Carvalho.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica

1. Automação. 2. Sistemas de fabricação integrada por computador. 3. Sistemas flexíveis de fabricação. I. Rosário, João Maurício. II. Carvalho, Marcius Fabius Henriques de. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. IV. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

TESE DE DOUTORADO

Arquitetura de Supervisão e Controle de Chão de Fábrica baseada em Componentes Genéricos

Autor : **Mauro Ferreira Koyama**
Orientador: **João Mauricio Rosário**
Co-orientador: **Marcius Fabius Henriques de Carvalho**

Prof. Dr. João Mauricio Rosário, Presidente
FEM / UNICAMP

Prof. Dr. François B. Vernadat
Universidade de Metz, França

Prof. Dr. Jean-Marc Faure
ISMCM - CESTI - Paris, França

Prof. Dr.Antonio Batocchio
FEM / UNICAMP

Prof. Dr. Paulo Corrêa Lima
FEM / UNICAMP

Campinas, 6 de Março de 2001

00-1959

Agradecimentos

Um grande número de pessoas colaborou para a realização deste trabalho, em especial gostaria de agradecer :

Aos meus orientadores Prof. João Maurício Rosário e Prof. Marcius Fabius Henriques de Carvalho, no Brasil, e Prof. François B. Vernadat, na França, por organizarem e apoiarem meu trabalho.

Aos Professores Jean-Marc Faure e Manuel de Jesus Mendes por aceitarem a tarefa de julgar meu trabalho na qualidade de relatores e por participarem da banca de tese.

Aos Professores Antonio Batocchio e Paulo Corrêa Lima e ao Prof. Ibrahima Sakho, por me honrarem participando da banca de tese.

Ao Prof. Didier Anciaux e ao Dr. Daniel Roy, da Universidade de Metz, pela orientação e excelente recepção que me deram na França, durante meu estágio naquele país.

Finalmente agradeço à minha esposa Adriana, à minha família e aos meus amigos, os quais me auxiliaram a transpor mais esta etapa profissional.

Resumo

KOYAMA, Mauro Ferreira, *Arquitetura para Controle de Chão de Fábrica baseada em Componentes Genéricos*, Campinas. Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2001. Tese de Doutorado.

O tema enfocado por esta tese é a construção de Arquiteturas de Controle de Chão de Fábrica, dentro do espírito de permitir a integração completa do mesmo à empresa. A idéia central desenvolvida é a de criar elementos construtores básicos, chamados Componentes Genéricos, que possam ser combinados de maneira a suportar o estabelecimento das arquiteturas de controle. Pode-se definir uma estratégia de construção dessas arquiteturas de controle, combinando os Componentes Genéricos e uma Metodologia de Projeto, que deve ser usada para cada caso específico. Da aplicação sucessiva dessa técnica poderão surgir padrões arquiteturais, que serão incorporados à uma biblioteca de modelos parciais *à la* CIMOSA. O processo de elaboração dos Componentes Genéricos e da metodologia associada, buscou reforçar as características de reutilização e modularidade, para permitir a criação de tais bibliotecas. Será apresentada uma aplicação para a geração de uma arquitetura de controle da Plataforma Integrada de Pesquisa, Ensino e Formação em Automação (PIPEFA), que contém uma célula de montagem e desmontagem.

Palavras Chave:

Automação, Sistemas de fabricação integrada por computador, Sistemas flexíveis de fabricação

Abstract

KOYAMA, Mauro Ferreira, *Architecture for Shop Floor Control based on Generic Components*, Campinas, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2001. Doctorate Thesis.

This work focuses on building architectures for shop floor control, aiming at enhancing integration of shop floor to the enterprise. Basic building blocks, called Generic Components, are developed. They can be combined to create control architectures. It is possible to define a building strategy, using the generic components and a project methodology, that shall be used in each specific industrial case. From successive applications of the technique, architectural patterns may emerge, making possible to compose libraries of shop floor's partial models. The process of elaborating Generic Components and associated methodology, enforced reuse and modularity characteristics, to allow creating such libraries. One practical application, on "Plataforma Integrada de Pesquisa, Ensino e Formação em Automação - PIPEFA", that contains one assembly and disassembly cell, will be presented, illustrating the whole process of architecture development.

Keywords:

Automation, Shop Floor Control, Enterprise Integration

Índice

Introdução Geral **1**

Problemática 1

Projeto Integrado de Chão de Fábrica 2

Plano Geral da Tese 3

1 Evolução das empresas **5**

1.1 Conjuntura do Mundo Industrial 6

1.2 Supervisão e controle de chão de fábrica 8

1.3 Conclusão 9

2 Supervisão e Controle de Chão de Fábrica **11**

2.1 Arquiteturas para Supervisão e Controle 12

2.1.1 Tipos de Produção 13
2.1.2 Tipos de Arquitetura 14

2.2 Integração das funções de Supervisão e Controle de Chão de Fábrica. 16

2.2.1 Arquiteturas de Referência 17

2.3 Disciplinas Relacionadas 31

2.3.1 Suporte para a Integração 32
2.3.2 Infra-estruturas de Integração (IEI) 37

2.3.3 Sistemas Multi-agentes 44

3 Uma Arquitetura Experimental baseada em Componentes 54

3.1 Noção de Componentes Genéricos 55

3. 2 Arquitetura Experimental 56

3.2.1 Derivação da Arquitetura Experimental 57
3.2.2 Agentificação 59

3.3 Componentes Básicos da Arquitetura 61

3.3.1 Modelo do Agente 62
3.3.2 Metodologia de construção de aplicações. 68

3.4 Robustez e flexibilidade da arquitetura 74

3.5 Conclusão 75

4 Aplicação da Arquitetura Experimental 77

4.1 Seleção de uma Arquitetura 78

4.1.1 PHOCS / PARSIFAL 78
4.1.2 SYROCO 80
4.1.3 Arquitetura selecionada 82
4.1.4 Derivação da Arquitetura Particular 84

4.2 Seleção de Algoritmo de Escalonamento 85

4.3 Projeto e Implementação 87

4.3.1 Decisões Tecnológicas. 88
4.3.2 Definição de Requisitos 89
4.3.3 Projeto Funcional 89
4.3.4 Implementação - Protótipos dos Agentes 94

4.4 Conclusão 100

5 Análise dos Experimentos e dos Resultados 103

5.1 Influência das Arquiteturas de Controle e de Referência. 103

5.2 Arquiteturas de Supervisão e Controle e Sistemas Multi-agentes. 105

5.3 Influência das Infra-estruturas de Integração (IEIs) 107

5.4 Desenvolvimento do Protótipo e Avaliação da Metodologia 107

5.4.1 Levantamento do Sistema Existente e Especificação dos Requisitos 108
5.4.2 Fase de Projeto 109
5.4.3 Fase de Implementação 110
5.4.4 Conclusão 112

6 Conclusão **114**

6.1 Trabalhos Futuros 115

6.1.1 Criação de novas arquiteturas 116
6.1.2 Simulação distribuída de ambiente de manufatura 116
6.1.3 Modelagem de Infra-estrutura de Integração 117
6.1.4 Desenvolvimento de algoritmos de escalonamento 117
6.1.5 Cooperação e Gestão do Conhecimento no Chão de Fábrica 118

Referências Bibliográficas **119**

Anexo 1 **129**

Lista de Figuras

Figura 2.1: Planejamento da Produção Tradicional 12

Figura 2.2: Classificação de Dilts *et al.* 14

Figura 2.3: Modelo de Referência GRAI 18

Figura 2.4: GRAI *Grid* 19

Figura 2.5: GRAI *Net* 20

Figura 2.6: Arquitetura PERA 21

Figura 2.7: Arquitetura ARIS 23

Figura 2.8: Cadeia de processos ARIS 24

Figura 2.9: Quadro de Modelagem CIMOSA 25

Figura 2.10: Visão Geral dos Conceitos CIMOSA 27

Figura 2.11: Componentes de GERAM 30

Figura 2.12: Relação Cliente/Servidor no MMS 33

Figura 2.13: Infra-Estrutura de Integração CIMOSA 38

Figura 2.14: EMEIS 41

Figura 2.15: Influências sobre os sistemas multi-agentes. 44

Figura 2.16: Vantagens dos Sistemas Multi-agentes 46

Figura 2.17: Aspectos da Arquitetura dos SMAs 47

Figura 2.18: Composição dos SMAs 47

Figura 2.19: Comunicação entre Agentes 48

Figura 2.20: Coordenação/ Negociação entre Agentes 49

Figura 3.1: Relacionamento do Planejamento da Produção e Chão de Fábrica 57

Figura 3.2: Partes Lógica e Física do Chão de Fábrica 58

Figura 3.3: Arquitetura Lógica X PAC [Huguet 95] X PAC [Bauer 94] 59

Figura 3.4: Agentificação e Arquitetura Experimental 60

Figura 3.5: Modelo do Agente 63

Figura 3.6: Estrutura das Mensagens 66

Figura 3.7: Ciclo de Vida de Projeto 69

Figura 3.8: Composição dos Dados 70

Figura 3.9: Diagrama de Uso 71

Figura 3.10: Diagrama de Seqüência 72

Figura 4.1: PHOCS / PARSIFAL 80

Figura 4.2: SYROCO 81

Figura 4.3: Derivação de SYROCO-2 85

Figura 4.4: Diagrama Geral de Syroco-2 90

Figura 4.5: Protótipo dos Agentes 95

Figura 4.6: PIPEFA 97

Figura 4.7: Estratégia de Migração 98

Figura A.1: Diagrama Geral de Syroco-2 129

Figura A.2: Cenário de Falha de Equipamento 135

Figura A.3: Diagrama de Classes de Syroco-2 136

Figura A.4: Rastreamento de mensagens 138

Nomenclatura

ARIS	*Architecture of Integrated Information Systems*
BDI	*Belief, Desire, Intention*
BP	*Business Process*
CESTI	*Centre d'Etudes Supérieures des Techniques Industrielles*
CIM	*Computer Integrated Manufacturing*
CIMOSA	*Open System Architecture for CIM*
COSIMA	*Control System for Integrated Manufacturing*
CSCW	*Computer Supported Cooperative Work*
DFD	Diagrama de Fluxo de Dados
EDI	*Electronic Data Interchange*
EMEIS	*Enterprise Model Execution and Integration Services*
FC/PAC	*Factory Control / Production Activity Control*
FIP	*Flux d'Information Processus*
FIPA	*Foundation of Intelligent Physical Agents*
GERAM	*Generalized Enterprise Reference Architecture and Methodology*
GRAI-GIM	*Graphes à Résultats et Activités Interreliés - GRAI Integrated Methodology*

IEI	Infra-estrutura de Integração
IFAC	*International Federation of Automatic Control*
IFIP	*International Federation for Information Processing*
ISA	*Instruments Society of America*
ISMCM	*Institut Supérieur des Matériaux et de la Construction Mécanique*
ISO	*International Organisation for Standardization*
ITI	Instituto Nacional de Tecnologia de Informação
JIT	*Just in Time*
KIF	*Knowledge Interchange Language*
KQML	*Knowledge Query and Manipulation Language*
LIISI	*Laboratoire d'Ingénierie Intégrée des Systèmes Industriels*
LGIPM	*Laboratoire de Génie Industriel et Production Mécanique*
MAP	*Manufacturing Automation Protocol*
MMS	*Manufacturing Message Specification*
MRP	*Manufacturing Resource Planning / Manufacturing Requirements Planning*
NBS	*National Bureau of Standards*
NIST	*National Institute of Standards and Technology*
PAC	*Production Activity Control*
PIPEFA	Plataforma Industrial para Pesquisa, Ensino e Formação em Automação
PERA	*Purdue Enterprise Reference Architecture*

POA	Projeto Orientado a Agentes
POO	Projeto Orientado a Objetos
PROFIBUS	*Process Field Bus*
SMA	Sistema Multi-agentes
SQL	*Structured Query Language*
STEP	*Standard for the Exchange of Product Model Data*
SGML	*Standard Generalized Markup Language*
SYROCO	*Système Réactif par Objectif de Conduite*
RDA	*Remote DataBase Access*
UML	*Unified Modeling Language*
VMD	Virtual Manufacturing Device

Introdução Geral

Problemática

Esta-se vivendo uma era de grandes mudanças, talvez uma das mais aparentes seja a crescente globalização da economia. Não é incomum a utilização no dia a dia de produtos das mais diversas origens geográficas: feitos no país, nos vizinhos ou em países distantes.

Esta presença de produtos manufaturados importados ilustra bem o aumento da competividade entre as empresas, que passou de um nível regional ou nacional para o nível internacional.

A constatação é de que esse é um movimento sem retorno, principalmente depois que os contra exemplos do capitalismo, as economias soviética e chinesa se transformaram parcialmente e que houve a reunificação da Alemanha, cujo decênio foi comemorado no ano 2000.

A globalização da economia é suportada por uma crescente malha de telecomunicações, cuja Indústria enfrenta, por seu lado, a revolução da Internet, com reflexos que vão desde a derrubada dos custos de conexões internacionais até a mudança de costumes de consumo.

Estas mudanças na economia comandam outras, nas estruturas produtivas e na organização das empresas.

No aspecto gerencial as linhas de mudanças são variadas, passando pelas técnicas incrementais de melhoria de qualidade exemplificadas no *Kaizen* [Imai 86], pela reengenharia de processos de negócios [Hammer 94] e, entre outras tantas pela Inovação de Processos [Davenport 93].

As estruturas produtivas são chamadas a adaptararem-se à esta realidade, por exemplo através do rearranjo do Chão de Fábrica para suportar a produção JIT [Bauer 94]. Ao mesmo tempo surge a necessidade da construção de quadros de referência que permitam facilitar o reprojeto das empresas, visando atender os novos requisitos de mercado. Dentre estas referências destacam-se as arquiteturas para modelagem de empresas, como ARIS [Scheer 92] e CIMOSA [Cimosa 93].

Apesar do grande avanço que estas arquiteturas representam, ainda está por ser resolvido o problema de reprojetar o Chão de Fábrica dentro de uma estratégia de integração de empresa. Acredita-se no entanto que as técnicas de modelagem previstas por estas arquiteturas podem ser adaptadas / estendidas para contemplarem este problema.

Projeto Integrado de Chão de Fábrica

O tema enfocado por esta tese é o projeto e construção de Arquiteturas de Controle de Chão de Fábrica, dentro do espírito de permitir a integração completa do mesmo à empresa.

A idéia central desenvolvida é a de criar elementos construtores básicos, chamados Componentes Genéricos, que possam ser combinados de maneira a suportar o estabelecimento das arquiteturas de controle. Não há soluções universais para a questão de supervisão e controle do chão de fábrica, assim como cada um deles apresenta suas peculiaridades em relação aos mais diversos aspectos, como tipo de produção, configuração física, equipamentos etc.

Foi definida uma estratégia de construção dessas arquiteturas de controle, combinando os Componentes Genéricos e uma Metodologia de Projeto, que deve ser usada para cada caso específico. Evidentemente com a aplicação sucessiva dessa técnica poderão surgir padrões arquiteturais, que serão incorporados à uma biblioteca de modelos parciais *à la* CIMOSA. Com efeito, o processo de elaboração dos Componentes Genéricos e da metodologia associada, buscou reforçar as características de reutilização e modularidade, para permitir a criação de tais bibliotecas.

Será apresentada uma aplicação para a geração de uma arquitetura de controle da Plataforma Integrada de Pesquisa, Ensino e Formação em Automação (PIPEFA), localizada no

Laboratório de Automação e Robótica, Faculdade de Engenharia Mecânica da Universidade Estadual de Campinas, a qual contém uma célula de montagem e desmontagem.

Plano Geral da Tese

No primeiro capítulo apresentar-se-á uma breve descrição da evolução das empresas, mostrando como as mudanças de mercado e a competividade crescente entre elas motiva o experimento e a criação de novas estruturas de produção e de arquiteturas de controle.

O segundo capítulo revisará o estado da arte em controle de chão de fábrica, mostrando as estruturas tradicionais e as propostas mais relevantes de novas estruturas. Serão então apresentadas brevemente as Arquiteturas de Referência para Integração de Sistemas, pois dela se extrairá as idéias básicas para a Metodologia de Projeto e Desenvolvimento e para os Componentes Genéricos.

Serão mostrados os esforços na construção de suporte informático para estas arquiteturas de referência, as chamadas Infra-Estruturas de Integração, bem como os padrões de comunicação industriais relacionados. Serão apresentados os Sistema Multi-Agentes e algumas aplicações para a classe de sistema distribuído representada pelo Chão de Fábrica.

O terceiro capítulo apresentará a proposta deste trabalho, através de uma conceituação de Componentes Genéricos e da descrição de uma Arquitetura Experimental. Para a construção de tal arquitetura foi necessário estabelecer uma Metodologia de Projeto e Desenvolvimento de Aplicações que será descrita.

Os Componentes Genéricos formam a base da Arquitetura Experimental, escolheu-se desenvolvê-los como agentes compondo um sistema multi-agentes. Serão avaliados os aspectos relevantes de composição do sistema, coordenação dos agentes, formas de comunicação e estruturação de protocolos de comunicação. A Metodologia de Projeto será o elemento de ligação entre estes diversos aspectos, para permitir a construção de uma arquitetura que seja ao mesmo tempo robusta e flexível, características estas que serão avaliadas.

No quarto capítulo mostrar-se-á como pode ser gerada uma Arquitetura Particular, a partir da Arquitetura Experimental. O problema escolhido foi o controle da célula de montagem e desmontagem da PIPEFA. Mostrar-se-ão duas arquiteturas existentes para a solução de

problemas assemelhados e selecionar-se-á a arquitetura SYROCO (*Système Réactif par Objectif de Conduite*) [Anciaux 97], [Roy 98] como referência. Descrever-se-á o projeto, onde foi utilizada a metodologia desenvolvida e gerado um protótipo executável. Discutir-se-ão os dados disponíveis da implementação e os resultados iniciais.

No quinto capítulo todo o processo de desenvolvimento dos Componentes Genéricos e da Metodologia de Projeto será analizado, em face dos resultados colhidos e mostrar-se-á que as propriedades especificadas foram conseguidas, dentro da limitação de tempo e recursos disponíveis. Os aspectos insuficientes, tanto em relação ao projeto, quanto à implementação serão levantados e analisados.

Finalmente, no sexto capítulo, apresentar-se-ão as conclusões e mostrar-se-á que os Componentes Genéricos, organizados através de uma Metodologia de Projeto e Desenvolvimento de Aplicações, são efetivos para a geração de Arquiteturas Genéricas de Controle de Chão de Fábrica. Propor-se-ão trabalhos de continuação à pesquisa, visando tanto aprimorar os elementos já estabelecidos quanto contemplar os aspectos insuficientes levantados no capítulo anterior.

Capítulo 1

Evolução das empresas

O modelo de produção de bens de consumo padronizados em massa encontra-se em declínio. Este descenso é especialmente perceptível a partir de meados da década de setenta quando afirma-se um novo modelo de consumo, onde há uma busca por mais funcionalidade e diversidade.

Alvin Toffler em seu livro "A Empresa Flexível" [Toffler 85] analisa o caso da corporação AT&T e aponta uma série de motivos para esta transição. As mudanças são fruto da evolução da própria sociedade industrial, mesclando efeitos sociais, culturais e de mercado. Dentre os fenômenos de mercado citados aparece claramente a digitalização da informação, que começa a ser transmitida também por meios alternativos para a época, como o satélite e os cabos de banda larga (neste último caso trata-se de sinais de televisão).

Os fenômenos sociais e culturais são exemplificados pela maior participação feminina na composição da mão de obra, pelas reivindicações das minorias étnicas demandando serviços diferenciados e por um crescente conflito de interesses entre as forças de mercado e as organizações governamentais e sociais.

Evidentemente a demanda por bens de consumo não decresceu, muito pelo contrário, o que acontece é uma mudança na qualidade do consumo. Os bens de consumo possuem um ciclo de vida cada vez mais curto, devido à crescente necessidade de inovação e de melhoria de qualidade, exigidas pelos consumidores.

A definição de produto está mudando, para incluir também serviços e informação (ou

conhecimento). Segundo Toffler [Toffler 85] isto caracteriza uma mudança de paradigma, da sociedade industrial para o de sociedade superindustrial (usando seus termos na época).

Outros autores confirmam estas mudanças em seus escritos. Peter Senge em seu livro "*The Fifth Discipline*" [Senge 94] apresenta o conceito da organização que aprende, pontificando que o conhecimento contido em uma empresa é seu maior patrimônio. Esta tese já era sugerida por Toffler quando este mostrava a evolução da produção, que já foi baseada na posse da terra (sociedade pré-industrial), no trinômio trabalho/ matérias-primas / capital (sociedade industrial) e finalmente na informação (sociedade pós-industrial).

Para este trabalho interessa profundamente a ênfase de Senge na "Quinta Disciplina", por ele chamada de "Pensamento Sistêmico" (*Systems Thinking*). Segundo ele, é esta visão sistêmica que permite uma análise mais apurada das empresas. A visão dos componentes de um problema isoladamente geralmente não permite a compreensão do problema como um todo. Seu exemplo didático é o da chuva, percebida antes de ocorrer pelas suas manifestações tais como nuvens negras que se acumulam e folhas tremulantes nas árvores. Antes mesmo da chuva ocorrer já pode-se também antecipar seus efeitos, pode-se supor que o volume dos riachos irá aumentar temporariamente e que, após esta chuva, o céu se limpará.

Tentar ver o todo ao invés de ver apenas os detalhes, ver a floresta e não apenas as árvores. O trabalho de Senge é orientado para as pessoas, pois são elas que detêm o conhecimento, outra disciplina citada por ele são os modelos mentais. Estes modelos podem não ser conscientes mas influenciam fortemente a capacidade das pessoas de lidar com situações diversas.

Mesmo quando a participação humana é intencionalmente minimizada, como acontece nos sistemas automáticos de produção, visão sistêmica e modelos são importantes ferramentas de análise e projeto, como buscar-se-á demonstrar neste trabalho.

1.1 Conjuntura do Mundo Industrial

Em resposta aos estímulos do mercado as empresas de manufatura modificam-se. As estruturas de produção passam das grandes séries - da produção em massa, para as pequenas séries e produção sob medida e sob demanda. Ressalta-se a crescente importância da Tecnologia de Grupo e da Manufatura Celular.

Já não parece futurista a possibilidade da produção de um certo bem (unitário) visando atender as especificações de um dado consumidor, mesmo que este bem nunca mais venha a ser produzido. Dependendo do bem, as limitações para sua produção hoje são apenas econômicas e não técnicas.

Mesmo no momento atual, onde ainda há uma predominância da produção em massa, a produção em pequenas séries e por pequenas empresas, apresenta um crescimento constante. Existe uma necessidade de aprimorar os mecanismos de produção e de controle da produção, para atender os requisitos deste tipo de demanda, a qual apresenta cada vez mais variabilidade e restrições de qualidade e prazo de entrega.

Charles Savage em seu livro "*5th Generation Management*" [Savage 96] aponta a necessidade de se dominar o conhecimento e o processo de conhecer (*knowledging*). Seu enfoque é mais próximo da manufatura, ele tece críticas ao modelo de CIM, que em realidade não promove a integração pois deixa o elemento humano fora da equação.

Seu trabalho apresenta também uma revisão breve das restruturações organizacionais, vista sob diversas óticas e com diversos nomes como: Sistemas Holônicos, Empresa Fractal, Organizações Virtuais para citar apenas os que apresentam maior interesse. Estes são esforços na direção de mudar o paradigma de gerência, outros esforços, provavelmente mais numerosos, são no sentido de informatizar as estruturas existentes.

Uma das conclusões de Savage é de que não basta informatizar o que já existe, é preciso repensar a organização. Ele traça um paralelo com o desenvolvimento dos computadores, que enfrentou o chamado "Gargalo de Von Neumann". Este gargalo foi superado com o advento da computação massivamente distribuída e das redes de computadores, no que foi chamado Computação de Quinta Geração. O foco de Savage é em formar uma rede de conhecimentos (*knowledge networking*) permitindo a passagem para a Gerência de Quinta Geração.

De Savage interessa este enfoque na transmissão de conhecimentos, não basta apenas **conectar** partes, é preciso definir **interfaces** que permitam que elas interoperem de maneira **integrada** (grifou-se os elementos do discurso de Savage).

1.2 Supervisão e controle de chão de fábrica

Muito já foi escrito sobre a Manufatura Integrada por Computador (CIM), hoje sabe-se que algumas das crenças que estavam por trás da sigla CIM foram invalidadas, como por exemplo o da empresa sem papéis e o da fábrica sem operários [Savage 96].

No entanto outros aspectos dele, como a informatização da produção e adoção de sistemas flexíveis de manufatura, foram melhor sucedidos e permitiram um aumento na produtividade das empresas.

Nesta tese há um interesse em pequenas e médias empresas, com uma produção diversificada. Relevante é pois a evolução dos **sistemas flexíveis de manufatura**, decorrente dos trabalhos em Tecnologia de Grupo e dos avanços dos sistemas informáticos (redes de computadores, bancos de dados, interfaces homem-máquina, sistemas de supervisão, etc.), do material de suporte (computadores, controladores programáveis, robôs, etc.) e do material produtivo propriamente.

Smith e Joshi [Smith 95] argumentam que 50-75% dos componentes manufaturados nos Estados Unidos são fabricados em lotes pequenos e médios. Dizem também que apesar da tecnologia CIM surgir como um facilitador para esta produção ela ainda não atingiu os benefícios esperados, devido aparentemente a problemas de integração.

Talvez estes problemas sejam decorrentes da complexidade dos sistemas de manufatura. Fala-se em sistemas Flexíveis de Manufatura, mas mesmo a definição do que é flexibilidade é árdua, como bem apontam Sethi e Sethi [Sethi 90] que indicam que "ao menos 50 termos para vários tipos de flexibilidade podem ser encontrados na literatura de manufatura". Em seu levantamento sobre flexibilidade eles discorrem sobre os aspectos econômicos, organizacionais e de manufatura e propõem uma organização de tipos entre os quais pode-se citar as flexibilidades: de processo, de roteamento, de produto, de volume e de expansão.

Uma definição interessante apresentada por eles é a de **Flexibilidade de Produção**, a qual é descrita como: "Flexibilidade de Produção é o universo de tipos de componentes que o sistema de manufatura pode produzir, sem adicionar maior capital em equipamentos". Argumentam que através de investimentos menores, tais como novas ferramentas pode-se, talvez as custas de

"*setups*" consideráveis, aumentar a produção.

A Flexibilidade de Produção, segundo eles, necessita da concorrência de diversos meios, como "variedade e versatilidade das máquinas, flexibilidade do sistema de transporte e do sistema de informação e controle em uso".

É importante ressaltar a necessidade de integração entre estes meios, para que a produção flexível seja realizada. Integração está intimamente ligada à tomada de decisão, como verifica-se no trinômio "Sistema de Decisão / Sistema Físico/ Sistema Operante" da arquitetura GRAI-GIM (vide Capítulo 2), na qual o elemento "Sistema de Informação" aparece como elemento de suporte destes três componentes.

Thierauf [Thierauf 82] considera que os sistemas de decisão possibilitam a gestão dos recursos disponíveis da empresa (homens, máquinas, materiais e capital) através do uso dos sistemas de informação. Um sistema de controle de chão de fábrica é parte integrante do sistema de decisão da empresa e é fundamental que haja um fluxo de informações entre os níveis mais altos de decisão e este sistema, nos dois sentidos, para que a gestão dos recursos seja efetiva.

Infelizmente os modelos de gerenciamento de empresa, segundo Savage, em sua maior parte são baseadas em hierarquias profundas (*steep hierarchies*). O Chão de fábrica aparece então como camadas inferiores destas hierarquias. Da mesma maneira como as estruturas organizacionais estão sendo forçadas a modificar-se, as estruturas de controle, até sob influência daquelas, também o estão.

Caminha-se para o fortalecimento da capacidade de decisão do chão de fábricam, uma mudança necessária para permitir à empresa uma adaptação rápida e eficiente às variações de demanda e de suprimento do mercado.

1.3 Conclusão

Neste capítulo apresentou-se a evolução das empresas ao longo do tempo, mostrando que está-se em pleno processo de transição, do paradigma da produção em massa para o de produção sob medida, da idéia de produto como bem de consumo físico e palpável à idéia de produto também como serviço ou conhecimento.

Os sistemas de produção apresentam os reflexos desta transição. A Manufatura Integrada por Computador (CIM) trouxe ganhos significativos na década de oitenta, mas não chegou a realizar plenamente seus objetivos naquela época, possivelmente por estar atrelada a estruturas antigas de produção.

Os Sistemas de Controle de Chão de Fábrica, parte da estratégia CIM, é onde se pode observar ganhos concretos em produtividade, devido ao crescente nível de informatização e automação. Mesmo assim estes sistemas não estão adaptados às novas exigências de mercado, que pedem capacidade de resposta rápida, qualidade, diversidade e custo reduzido.

Acredita-se que os trabalhos de Toffler [Toffler 85], Senge [Senge 94] e Savage [Savage 96], mesmo tendo como foco os elementos organizacionais, apontam o caminho da mudança: a gestão do conhecimento da empresa. Para realizar esta gestão é necessário adquirir uma visão sistêmica, que considere o maior número possível de aspectos e as fases de evolução temporal.

O problema de integração da manufatura é extremamente complexo, o que permite abordá-lo é a utilização de **modelos**, que são transformados em elementos de diálogo entre as pessoas que devem realizar este projeto de integração. Como Savage aponta, não basta apenas conectar subsistemas, é preciso integrá-los e para isto são necessárias interfaces.

O chão de fábrica reproduz a problemática com a qual as empresas se deparam; assim como estão sendo pesquisadas novas estruturas organizacionais (Fractal, Holônica, Virtual ...) também devem ser pesquisadas novas arquiteturas de controle, a partir da premissa de que o **conhecimento** (e a capacidade de decisão decorrente dele) juntamente com a partilha (**comunicação**) do mesmo devem ser os elementos básicos de projeto.

No capítulo seguinte será mostrado o estado da arte em sistemas de controle de chão de fábrica, em modelagem de integração e arquiteturas de referência e haverá uma discussão sobre sistemas de informação considerando explicitamente as disciplinas de infra-estrutura de integração, padrões industriais de comunicação e sistemas multi-agentes.

Capítulo 2

Supervisão e Controle de Chão de Fábrica

Como mostrado no capítulo anterior, as mudanças nos perfis de consumo engendram outras nas estruturas produtivas. Nesta tese o interesse é no elemento final de produção, o Chão de Fábrica. Para propor um enfoque de desenvolvimento de estruturas de controle / arquiteturas adequadas à nova realidade, será mostrado um panorama do estado atual de conhecimento sobre "Supervisão e Controle de Chão de Fábrica", sobre a integração das funções de controle dentro do âmbito maior das "Arquiteturas de Referência" e finalmente serão revisadas um conjunto de disciplinas que apresentam interesse para a realização deste trabalho, que são as "Infra-estruturas de Integração", os protocolos de comunicação industriais e os Sistemas Multi-Agentes.

2 Supervisão e Planejamento da Produção

É necessário posicionar o Chão de Fábrica em relação ao problema de gerenciamento de Empresa. Ele está inserido dentro do contexto maior que é o de Gerenciamento da Produção. Bauer *et al.* [Bauer 94] descreve Planejamento e Controle da Produção como consistindo de três níveis de decisão: estratégico, tático e operacional.

O nível estratégico considera aspectos como atendimento às expectativas de mercado e composição da produção e determina quais produtos serão manufaturados.

No nível tático serão gerados planos de produção, baseados na demanda estimada no nível estratégico. Ferramentas de MRP (*Manufacturing Resource Planning, Materials Requirements*

Planning) são suporte nesta fase, trabalhando com "*lead times*" estimados e com a explosão da árvore de produtos geram um conjunto de ordens de produção para o Chão de Fábrica.

No nível operacional as ordens de produção planejadas serão executadas. O Chão de Fábrica está posicionado neste nível gerencial, recebendo ordens de produção e materiais e fornecendo produtos e informações sobre o estado da produção. Na Figura 2.1 mostra-se um sistema simplificado de Planejamento da Produção Tradicional. Uma discussão sobre um modelo mais genérico pode ser encontrada em Howard *et al.* [Howard 99]. A ênfase no modelo apresentado está em mostrar dois níveis de decisão, no retângulo superior estão os níveis estratégico e tático e no retângulo inferior o nível operacional. Ao mesmo tempo mostra-se que os componentes do nível superior são acoplados via informação / dados (pedidos de produção, previsões, níveis de estoque, relação de materiais etc.) e que o nível de controle de chão de fábrica não têm acesso à esses dados.

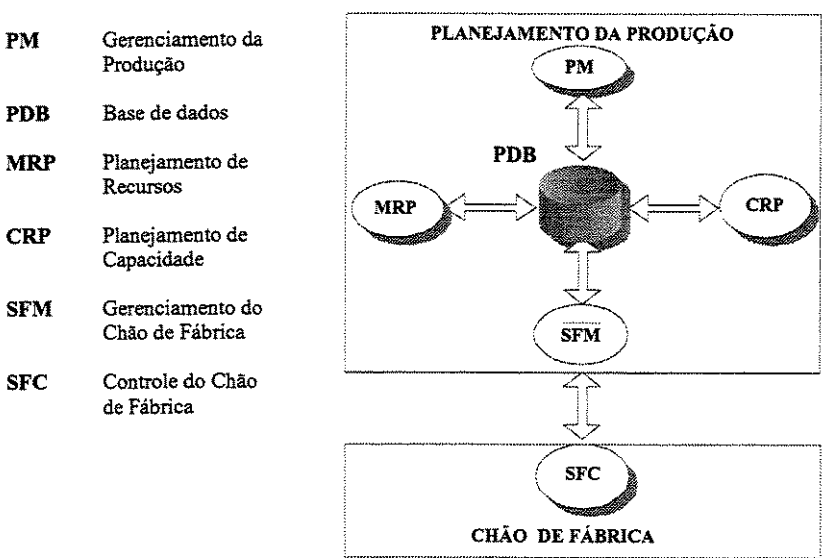


Figura 2.1: Planejamento da Produção Tradicional

2.1 Arquiteturas para Supervisão e Controle

Para produzir é necessário organizar os meios de produção dentro de uma arquitetura. A definição proposta por Bauer *et al* será usada.: "Uma arquitetura estabelece um quadro ou um

conjunto de regras e diretivas para gerenciar o desenvolvimento e operação de sistemas complexos".

Uma conhecida arquitetura para produção é a chamada "Arquitetura NBS" [McLean 83], cujo nome deriva da instituição onde foi concebida, o "*National Bureau of Standards* - NBS", atualmente NIST (*National Institute of Standards and Technology*). A arquitetura NBS é hierárquica e contém cinco níveis: Fábrica, Planta, Célula, Estação e Equipamento. Esta arquitetura prevê a possibilidade da utilização de "Células Virtuais", formadas através da combinação de Estações que podem pertencer a células físicas distintas. As células físicas são habitualmente formadas por intermédio de uma análise baseada em Tecnologia de Grupo, visando agrupar os produtos em famílias e destinando células específicas para a fabricação desta famílias. As células virtuais quebram esta restrição ao permitir que qualquer equipamento do Chão de Fábrica possa ser usado na sua composição, o que pode ser útil quando ocorrerem imprevistos na fabricação.

A Arquitetura NBS teve uma grande projeção e foi estendida através do "Modelo de Automação de Fábrica" da ISO (*International Organisation for Standardization*).

2.1.1 Tipos de Produção

Antes de continuarmos com a discussão sobre arquiteturas é preciso fazer uma distinção breve entre os diferentes tipos de produção. A produção pode ser contínua, quando existe um fluxo contínuo de material entre os meios de produção, ou discreta, quando este fluxo for discreto ou seja, realizar-se por unidades fisicamente reconhecíveis.

Um exemplo de produção contínua é uma Refinaria de Petróleo, que recebe petróleo bruto e fornece derivados como gasolina, querosene e outros.

Para a produção discretizada interessa saber o tamanho dos lotes de produção e a variação no tipo de produtos. A produção em massa consiste em produzir lotes muito grandes. Considera-se neste trabalho uma produção diversificada, que conterá lotes de menor tamanho e com uma grande gama de produtos.

2.1.2 Tipos de Arquitetura

O trabalho de Dilts *et al.* "*The Evolution of Control Architectures for Automation Manufacturing Systems*" [Dilts 91] apresenta e compara diversas arquiteturas, criando um quadro de referência para posicionar arquiteturas existentes e novas propostas. As arquiteturas espelham os mecanismos de tomada de decisões. Na Figura 2.2 é reproduzida a classificação de Dilts *et al.* Duffie e Prabhu [Duffie 96] apresentam também uma classificação que vai da forma hierárquica à heterárquica, passando por formas oligárquicas e semi-heterárquicas.

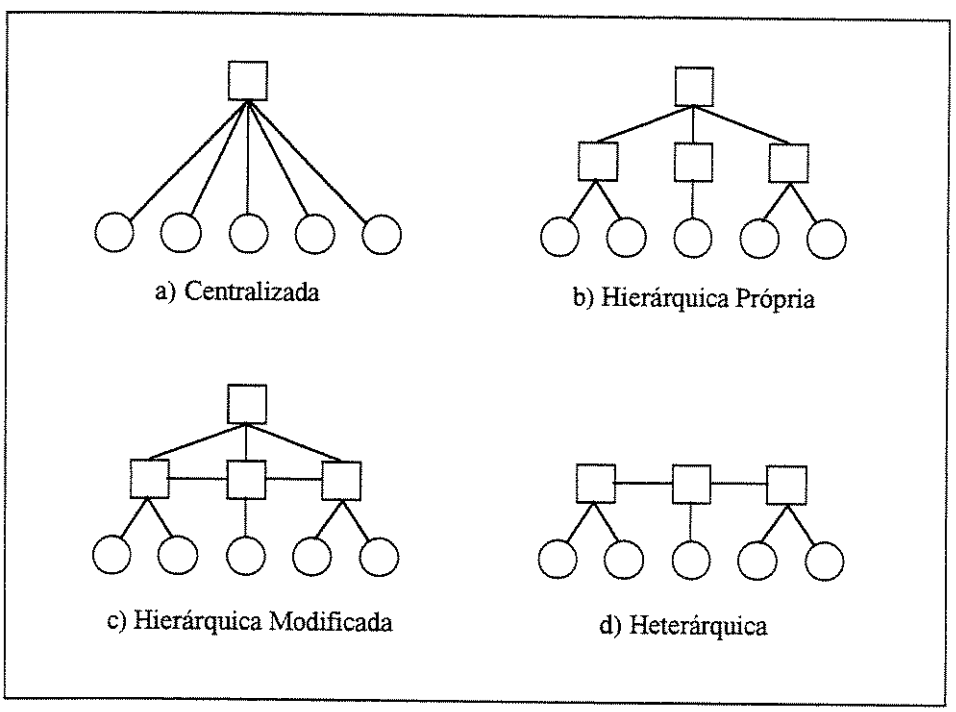


Figura 2.2: Classificação de Dilts *et al.* **Fonte: [Dilts 91]**

a) Centralizada - Nesta classificação a forma Centralizada indica que a capacidade de decisão e controle estão centralizadas em um nó, os demais nós são considerados como sendo destituídos de inteligência.

   A grande vantagem dessa estrutura é: acesso global às informações está disponível, facilitando otimizações e minimização do uso de recursos computacionais. As

desvantagens são: ponto único de controle torna deficiente a resistência à falhas; tempo de resposta lento, não escalabilidade do sistema e problemas de atualização da estrutura.

b) Hierárquica Própria - Esta forma, cujo padrão de referência é a arquitetura NBS anteriormente citada, permite a decomposição do sistema sob controle em diversas camadas, possibilitando a existência de níveis de controle.

   Vantagens desta estrutura são: melhor eficiência do controle; melhorias nas características de confiabilidade e na flexibilidade do sistema; possibilidade de crescimento incremental (escalabilidade) e possibilidade de distribuição dos esforços computacionais. As desvantagens são: a especialização no tratamento de informações, contida em cada nível, pode dificultar o tratamento de falhas por necessitar uma propagação dessas informações através da hierarquia e finalmente a rigidez do sistema dificulta a realização de modificações não previstas.

c) Hierárquica Modificada - Esta forma busca suprir as deficiências da forma Hierárquica Própria ao permitir que dentro do mesmo nível hierárquico haja comunicação entre nós, o que facilita a sincronização de tarefas de maneira independente da estrutura mestre-escravo que permeia as hierarquias. O propósito é afrouxar as relações mestre-escravo, dando mais autoridade a alguns nós, que ganham capacidade local de decisão. Esta característica beneficia-se da crescente capacidade computacional disponível no chão de fábrica através dos dispositivos inteligentes e computadores.

   As vantagens desta forma são aquelas da forma hierárquica própria somadas a um aumento na autonomia dos módulos, proporcionando melhor resistência a falhas e liberando os níveis superiores para tratar de maneira mais completa as informações.

   As desvantagens também são associadas aquelas da forma hierárquica própria, somando-se problemas de conectividade dentro do mesmo nível hierárquico e um possível aumento da complexidade do controle.

d) Heterárquica - A forma heterárquica é um desenvolvimento recente, onde destacam-se as contribuições de Duffie *et al.* [Duffie 94], [Duffie 96]. Segundo Dilts *et al.* nesta estrutura o propósito é obter autonomia plena dos módulos componentes, implicando na minimização ou

eliminação dos repositórios globais de informação. Os módulos usarão localmente os repositórios de informação que necessitem e eventualmente os partilharão com outros módulos. A cooperação entre estes módulos necessita de alguma forma de coordenação, esta coordenação é baseada em protocolos os quais devem prever a possibilidade de um módulo recusar-se a executar uma ação requisitada (isto é a relação é entre pares e não mestre-escravo).

Vantagens propaladas da forma heterárquica são: a complexidade do software de controle decresce devido à modularidade do sistema e do reduzido acoplamento entre módulos; melhoria na capacidade de tratamento de falhas e finalmente novas formas de escalonamento estão em desenvolvimento permitindo ganhos de desempenho.

Algumas desvantagens são: se o sistema for computacionalmente heterogêneo surgem problemas de interconexão e desempenho da comunicação; eventualmente pode-se perder ganhos de otimização global, pela dificuldade em obter os dados necessários para esta otimização e finalmente não há disponibilidade de aplicativos comerciais que suportem esta estrutura.

Há uma forte corrente de investigação de estruturas heterárquicas para controle da produção; o uso destas estruturas necessitará do desenvolvimento de novos algoritmos de escalonamento e controle [Baker 98], [Müller 98].

Dilts *et al.* [Dilts 91] descrevem as arquiteturas segundo sua decomposição em níveis, um trabalho complementar e interessante é a análise dos mecanismos de decisão e operação dentro de cada nível. Este é o assunto do trabalho de Huguet e Grabot [Huguet 95], que descrevem os níveis baseados na inter-relação de quatro funções: planejar, despachar, seguir e reagir. Das possíveis combinações dessas funções desenvolvem um conjunto de sete possíveis variantes, as quais são comparadas com arquiteturas existentes, como por exemplo a de Bauer *et al.* "*FC/PAC*" [Bauer 94].

2.2 Integração das funções de Supervisão e Controle de Chão de Fábrica.

Um Sistema para Supervisão e Controle de Chão de Fábrica é parte efetiva do Sistema de Decisão e Controle da empresa, isto significa que para integrá-lo dentro da empresa é preciso

possuir uma compreensão de como se dá esta inserção.

As empresas, na busca por um aumento em sua eficiência e competitividade, procuraram em um primeiro instante, ao nível da estrutura produtiva, utilizar técnicas isoladas como Tecnologia de Grupo e MRP (*Materials Requirements Planning, Manufacturing Resource Planning*) [Ploss 87], [Agostinho 1991]. Tais técnicas apesar de promover ganhos de produtividade mostraram-se insuficientes, surgiu então um movimento na direção de uma maior integração entre os componentes da empresa, visando eliminar as "ilhas de automação /informatização". Uma primeira fase do esforço de integração foi na linha de buscar os elementos invariantes de uma empresa [Burbidge 87], com uma visão claramente funcional (funções de gerência). O modelo CAM-I, como descrito por Doumeingts [Doumeingts 84], com suas fases de Decidir, Projetar, Adquirir, Produzir, Verificar e Entregar, teve influência neste período [Doumeingts 95].

O modelo NBS [MCLean 83], também fincou raízes, ao propor uma hierarquia para controle de sistemas de manufatura. Este modelo influenciou vários projetos europeus [Doumeingts 95] servindo até os dias atuais como elemento de referência junto ao qual os novos projetos buscam se posicionar. Exemplo deste posicionamento é o modelo PAC, descrito por Bauer *et al.* [Bauer 94] e originário do projeto europeu COSIMA (ESPRIT Project 477).

Numa segunda fase buscou-se uma maior abstração na descrição dos relacionamentos dentro da empresa, quebrando as barreiras funcionais, dentro de uma linha de descrição por processos de negócios (BP). O grande expoente desta tendência é o projeto CIMOSA (ESPRIT *Project* 688) [CIMOSA 93] sendo que este projeto ressalta a importância de se considerar a empresa sob diversas vistas.

2.2.1 Arquiteturas de Referência

Serão descritas brevemente algumas arquiteturas de referência para integração de empresa. Elas foram selecionadas segundo os critérios de importância conceitual e histórica e a descrição irá ressaltar o aspecto da infra-estrutura de integração proposta nas mesmas (quando existente) por ser este o vínculo mais importante com este trabalho. Uma descrição mais abrangente pode

ser encontrada em "*Enterprise Modeling and Integration: Principles and Applications*" [Vernadat 96]. As Arquiteturas são: GRAI-GIM, PERA, ARIS, CIMOSA e GERAM.

2.2.1.1 GRAI-GIM

Esta arquitetura foi desenvolvida na Universidade de Bordeaux, França. Doumeingts, em sua tese de doutorado, descreve o método GRAI (*Graphes à Résultats et Activités Interreliés*) que deu origem à arquitetura [Doumeingts 84]. O nome GIM atualmente significa GRAI Integrated Methodology /ISO 97b] e é uma extensão ao método GRAI.

GRAI é baseada em um modelo conceitual, chamado Modelo GRAI, que é apresentado na Figura 2.3, na qual quatro entidades são distinguidas: Sistema de Informação, Sistema de Decisão, Sistema Físico e Sistema Operante. Tais sistemas são decompostos segundo uma hierarquia de níveis, de tal forma que dentro de cada nível pode-se aplicar recursivamente a decomposição em sistemas informático, de decisão, físico e operante.

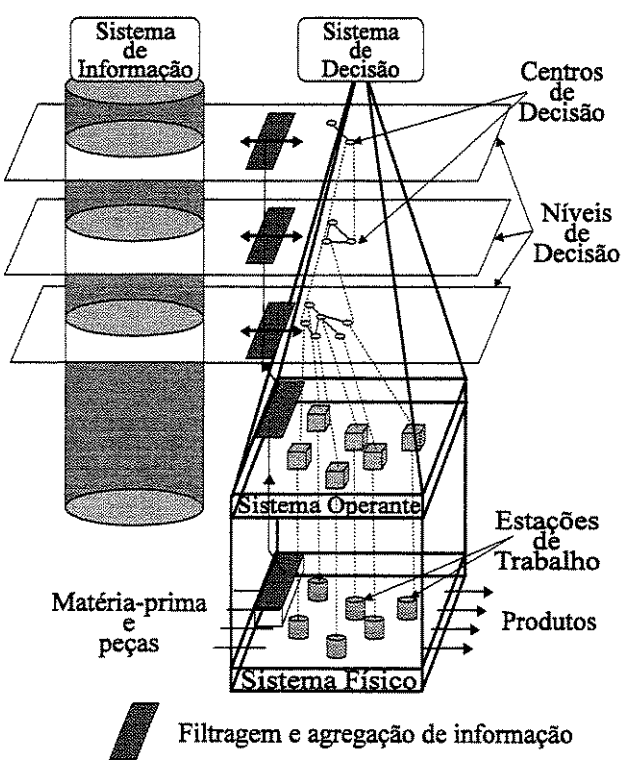


Figura 2.3: Modelo de Referência GRAI **Fonte [Doumeingts 95]**

O método GRAI baseia-se no uso de duas ferramentas, GRAI *Grid* e GRAI *Nets* [Doumeingts 87].

GRAI *Grid* é usada para descrever hierarquicamente a estrutura de decisão de um Sistema de Gerência de Produção, consiste em uma matriz na qual as linhas representam os centros de decisão, classificados de cima para baixo de acordo com os critérios de Horizonte e Período de Decisão. Horizonte de decisão (H) é um intervalo de tempo em que os estados do sistema são conhecidos e está relacionado aos seus ciclos (de fabricação, de compras, etc.). Período (P) é o intervalo de tempo ao fim do qual os estados serão reavaliados com o propósito de reavaliar as decisões de uma maneira regulatória [Doumeingts 84]. A Figura 2.4 apresenta um exemplo de GRAI *Grid*.

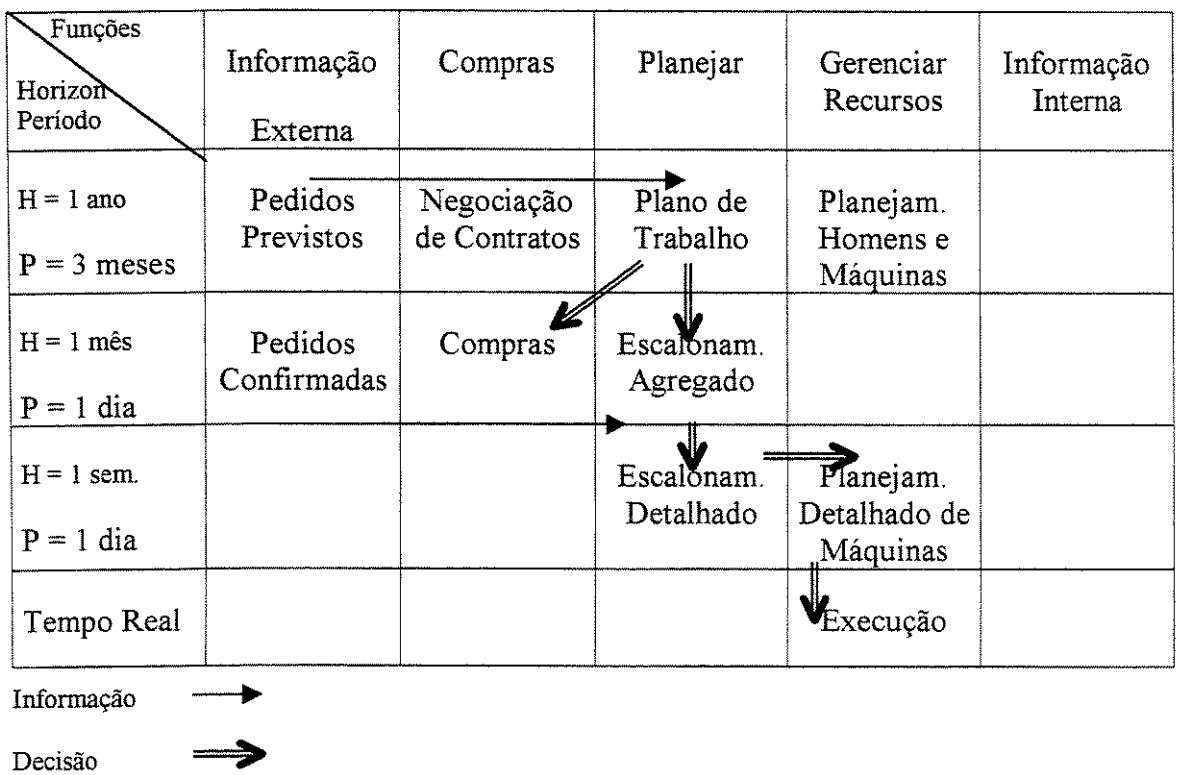


Figura 2.4: GRAI *Grid* **Fonte: [Doumeingts 87]**

GRAI *Nets* são usadas para descrever as atividades de cada centro de decisão, mostrando também quais são as informações e recursos necessários. A Figura 2.5 apresenta um exemplo de GRAI *Net*.

Quanto ao ciclo de vida de projeto de empresa, existe uma Metodologia Estruturada GIM, que comporta as fases de Iniciação, Análise, Projeto e Implementação [Vernadat 96].

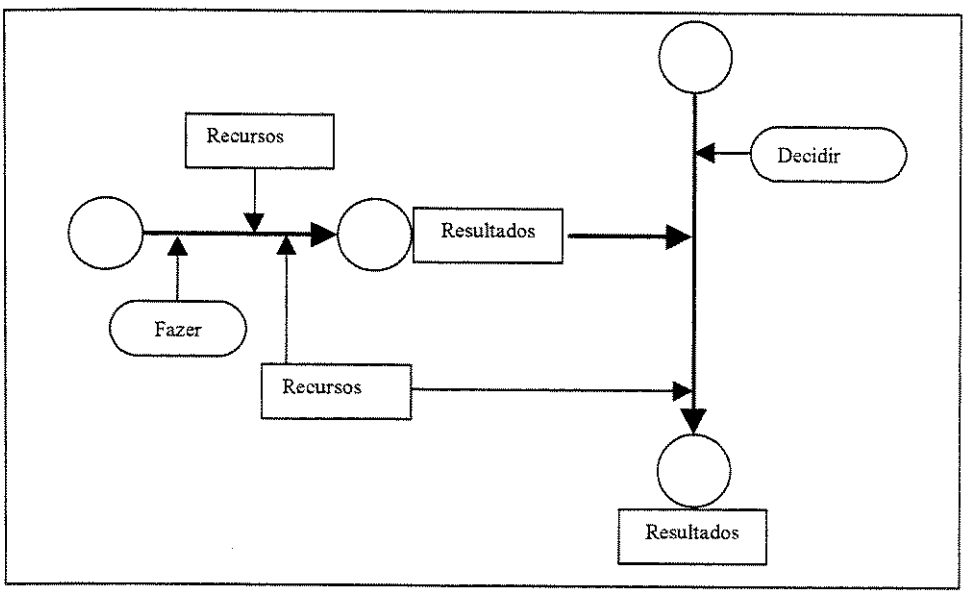


Figura 2.5: GRAI *Net* **Fonte: [Doumeingts 87]**

Quanto à infra-estrutura de integração pode-se dizer que, no trabalho original descrito no método GRAI [Doumeingts 84], ela não era considerada, sendo que o conceito de Sistema Operante ainda não foi detalhado e não encontrou-se referência mesmo nos trabalhos recentes [Doumeingts 95], [ISO 97b], [Vernadat 96].

2.2.1.2 PERA (*Purdue Enterprise Reference Architecture*)

A arquitetura PERA foi desenvolvida pelo professor Theodore Williams, da Universidade de Purdue, Estados Unidos [Williams 92].

A Figura 2.6 apresenta a Arquitetura PERA . Do lado esquerdo vê-se sua proposta para o ciclo de vida de um projeto de integração, contendo seis fases: conceitual, definição, especificação, projeto, implementação, operação/manutenção.

No lado direito vê-se a arquitetura propriamente, que segundo Williams [Williams 92] é: "apenas um modelo ou quadro mostrando as relações dos vários passos envolvidos na realização

de um programa de integração de empresas...". A estrutura (ou quadro) contém três arquiteturas: arquitetura do sistema de informação (ASI), arquitetura do sistema de manufatura (ASM) e arquitetura organizacional e humana (AOH).

Segundo Bernus *et al.* [Bernus 96], a PERA representa de forma original o envolvimento humano no sistema de Integração da Empresa e seu relacionamento com os demais elementos e funções do sistema de manufatura. Observa-se também que o ciclo de vida proposto é bastante amplo, indo do posicionamento da empresa face aos seus ambientes interno e externo, passando pela definição da sua missão, visão e valores, até o momento da desativação da mesma. Esta amplitude de descrição fez com que este ciclo de vida tivesse influência na Arquitetura GERAM que será apresentada posteriormente.

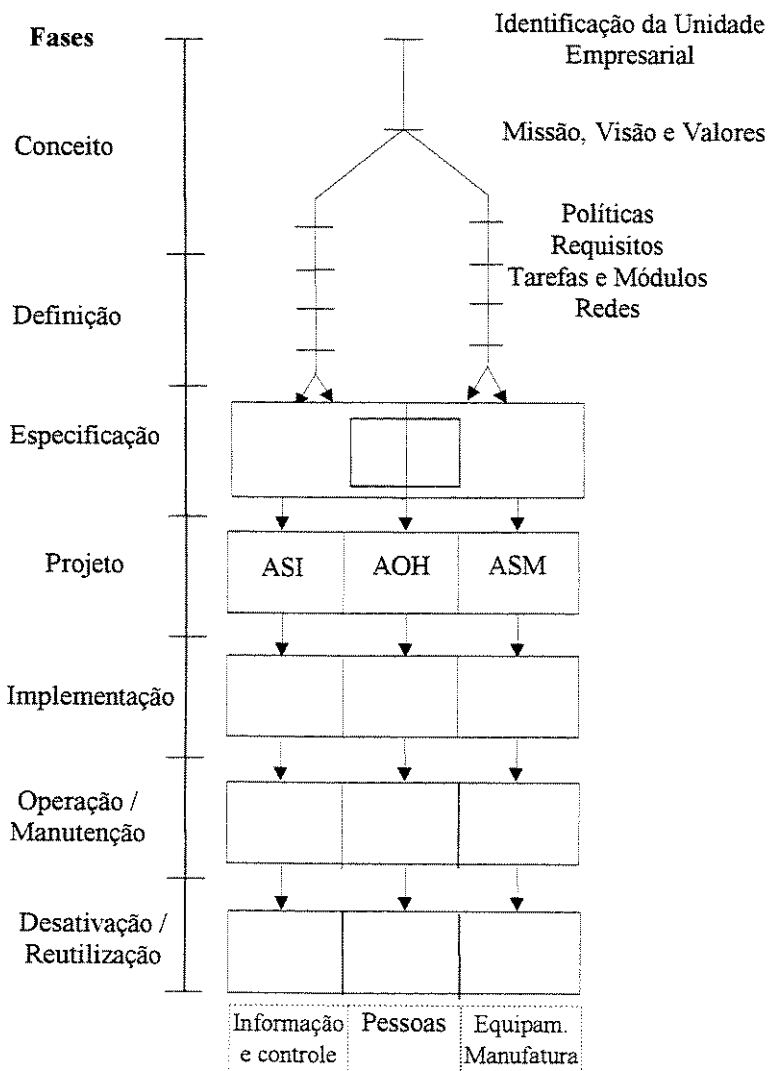


Figura 2.6: Arquitetura PERA **Fonte: [Williams 96]**

Para permitir a implementação da arquitetura foi desenvolvida uma metodologia descrita no documento "*PERA Master Planning Handbook*" [Williams 1996]. PERA enfatiza o planejamento do projeto de integração, via um *Plano Diretor*. A elaboração deste resulta da execução das três primeiras fases do ciclo de vida mostrado na Figura 2.6.

Esse plano essencialmente apresenta o estado atual e o estado desejado da unidade empresarial escolhida para ser integrada e descreve como evoluir para esse último.

A metodologia também propõe detalhadamente como proceder nas etapas de projeto e implementação (agrupadas sob o nome de "Programa de Integração da Empresa"). Neste sentido as especificações da PERA são as mais detalhadas atualmente disponíveis publicamente para um projeto de integração de empresas.

No tocante à infra-estrutura de integração não há na PERA uma elaboração desse tópico. O enfoque de PERA é em mapear o sistema físico existente e propor sua transformação no novo sistema, apesar de muito prático não apresenta uma evolução conceitual neste aspecto de infra-estrutura de integração.

2.2.1.3 ARIS (*Architecture of Integrated Information Systems*)

Esta arquitetura destina-se a suportar a análise, projeto e implementação de sistemas de informação. Foi elaborada pelo professor August-Wilhelm Scheer [Scheer 92], da Universidade de Saarbrücken, Alemanha, a partir do conceito de Modelo de Cadeia de Processo.

São utilizados três níveis para o ciclo de vida: definição de requisitos, especificação de projeto e descrição de implementação (Figura 2.7). São consideradas quatro vistas de empresa e cada uma delas é descrita de acordo com os níveis citados.

A Arquitetura ARIS utiliza um conjunto de métodos advindos da Engenharia de Software, isto facilita sua compreensão e uso pelos usuários. Os critérios para seleção destes métodos são [Scheer 94]:

- simplicidade dos meios de representação
- adequação aos conteúdos que devem ser expressos

- capacidade de usar métodos consistentes para todas as aplicações a serem representadas
- existência de um grau de familiaridade com os métodos
- independência dos métodos, num certo grau, dos desenvolvimentos em tecnologia de comunicação e informação

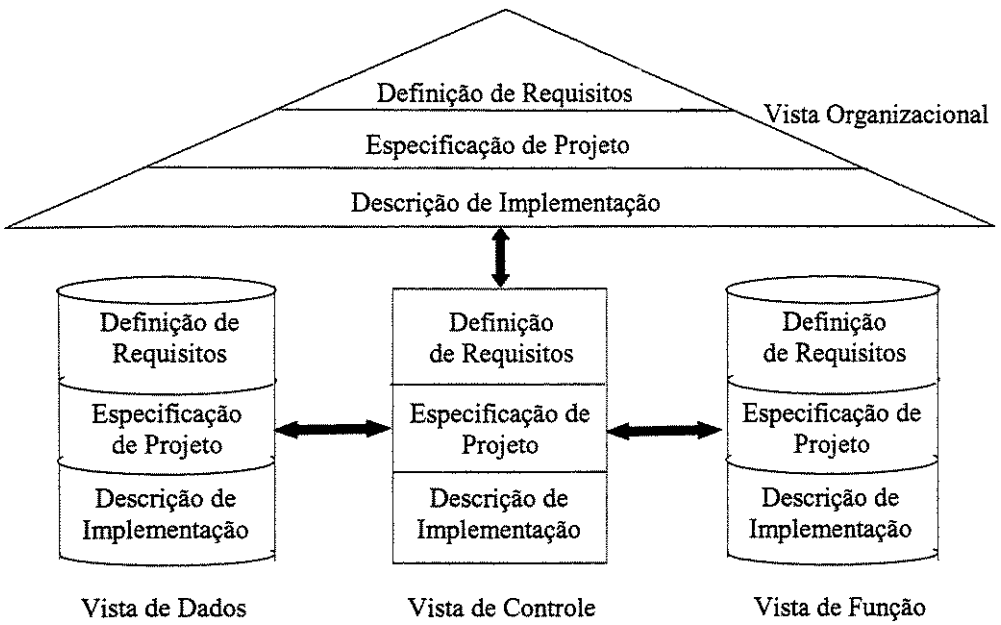


Figura 2.7: Arquitetura ARIS **Fonte: [Scheer 92]**

A Figura 2.8 apresenta um exemplo de cadeia de processo que permite ilustrar as vistas usadas em ARIS.

Esta cadeia consiste numa seqüência de processos individuais que transformam informações, sendo que cada processo é iniciado e concluído sob à ação de eventos.

A modelagem de empresa segundo ARIS envolve quatro vistas: função, informação, controle e organização.

A Vista de Função descreve uma hierarquia de funções especificadas como estruturas e módulos de programas que tornar-se-ão código de programa.

A Vista de Informação contém modelos semânticos de dados definidos na forma de diagramas entidade-relacionamento.

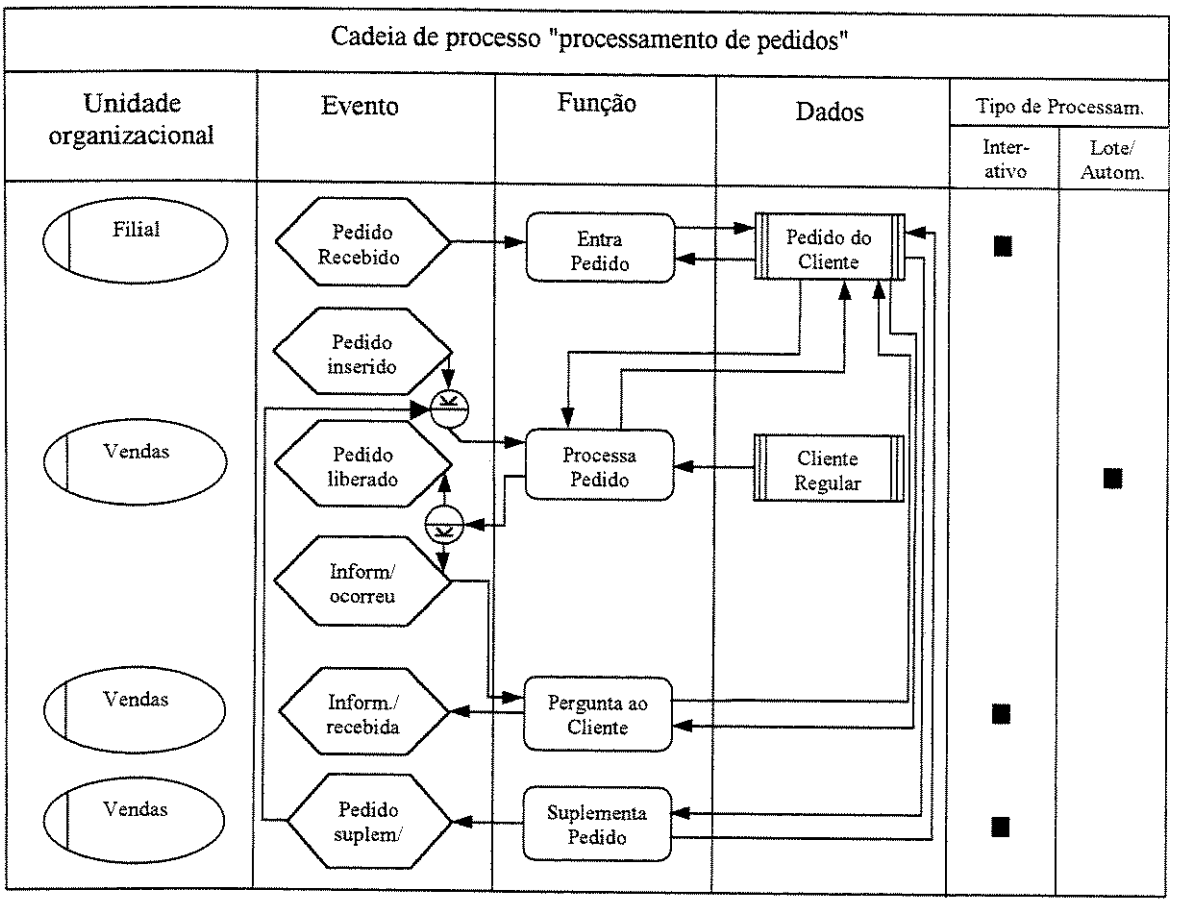


Figura 2.8: Cadeia de processos ARIS **Fonte: [Scheer 94]**

A Vista de Organização estabelece departamentos e pessoas relacionadas a cada processo, topologia de redes de comunicação etc.

A Vista de Controle relaciona-se às três outras vistas e contém as cadeias de atividades ou processos de negócios implementadas como seqüências lógicas de programas computacionais [Scheer 92], [Vernadat 97a].

Essa arquitetura conta com o suporte de ferramentas computacionais comercializadas pela empresa *IDS Scheer* e permitem projetar, avaliar e aplicar processos de negócio à empresa. ARIS também é utilizada em conjunto com o software R3 da SAP [Scheer 94], fatores que a tornam a arquitetura com o melhor suporte computacional.

No entanto este suporte não se reflete na descrição clara de uma infra-estrutura de integração.

2.2.1.3 CIMOSA (*Open System Architecture for CIM*)

A arquitetura CIMOSA foi desenvolvida na Europa, através do Consórcio AMICE que contava com a participação de diversas empresas (Digital, IBM, HP, etc.) e centros de pesquisa (GRAI, WZL,INRIA ...) [CIMOSA 93]. Seu objetivo é "permitir a modelagem de empresa para a exploração eficiente do conhecimento da mesma".

CIMOSA utiliza três conceitos para permitir a modelagem de empresas:

- Quadro de referência (contendo Arquitetura de Referência, Arquitetura Particular, Modelo da Empresa).

- Ciclo de Vida do Sistema e Ambientes de Engenharia e Operação.

- Infra-estrutura de Integração.

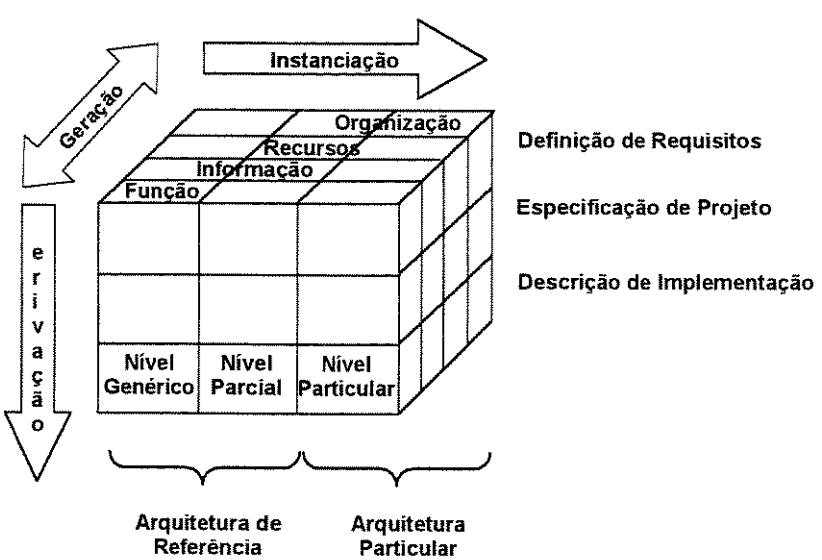


Figura 2.9: Quadro de Modelagem CIMOSA **Fonte: [Vernadat 96]**

Na Figura 2.9 é apresentado o Quadro de Modelagem CIMOSA, no qual pode-se ver a descrição da empresa segundo três dimensões: Geração de Vistas, Instanciação de Blocos

Funcionais e Derivação de Modelos. Estas dimensões definem as faces do chamado Cubo CIMOSA.

Na face superior do cubo vêem-se as vistas atualmente existentes, que têm o propósito de modelar a estrutura e comportamento do sistema sob análise. No momento estas vistas são de: Função, Informação, Recursos e Organização.

A Vista de Função permite descrever a funcionalidade do sistema e seu comportamento. A Vista de Informação descreve as estruturas de informação necessárias para gerir o sistema; a Vista de Recursos permite determinar quais são as capacidades necessárias para realizar os processos de negócio e a Vista de Organização permite modelar os mecanismos de tomada de decisão dentro da empresa.

Vernadat [Vernadat 96] argumenta que estas vistas podem ser expandidas se necessário.

Na face frontal a dimensão de Derivação de Modelos apresenta parte do ciclo de vida CIMOSA, referente àquelas fases em que há metodologias já definidas. As fases são:

- Definição de Requisitos, onde são descritos os domínios a serem modelados, os processos de negócio destes domínios e os objetos associados.

- Especificação de Projeto, onde encontram-se especificações detalhadas das atividades a serem realizadas, bem como os recursos e informações necessários para sua execução e a unidade organizacional responsável.

- Descrição da Implementação, descreve os meios computacionais e humanos necessários para a execução dos processos de negócio e atividades associadas, considerando as restrições de tempo e recursos dos subsistemas de informação e de manufatura.

Estas fases não são puramente seqüenciais, podendo haver interações entre elas, facilitando a depuração e operacionalização dos modelos [CIMOSA 93].

A dimensão de Instanciação determina o grau de generalidade dos modelos gerados. Parte-se de modelos altamente genéricos, passíveis de utilização por uma ampla classe de empresas (princípio da reutilização) ; passa-se por modelos parciais, os quais são adequados para classes

particulares de empresas (por exemplo: Automobilísticas, Eletrônicas, Aeroespacial, etc.) e chegam-se nos modelos particulares que são derivados para uma empresa específica.

Vê-se também na Figura 2.10 a distinção feita entre Arquitetura de Referência e Arquitetura Particular. A Arquitetura de Referência pretende ser um repositório de modelos, contendo os blocos construtivos básicos CIMOSA e modelos reutilizáveis adequados para amplos segmentos da indústria (por exemplo: MRP, Compras, Logística, etc..). A parte desta arquitetura que contém modelos parciais deve ser desenvolvidas por consórcios de empresas de um certo ramo de especialização, que se organizarão com este objetivo.

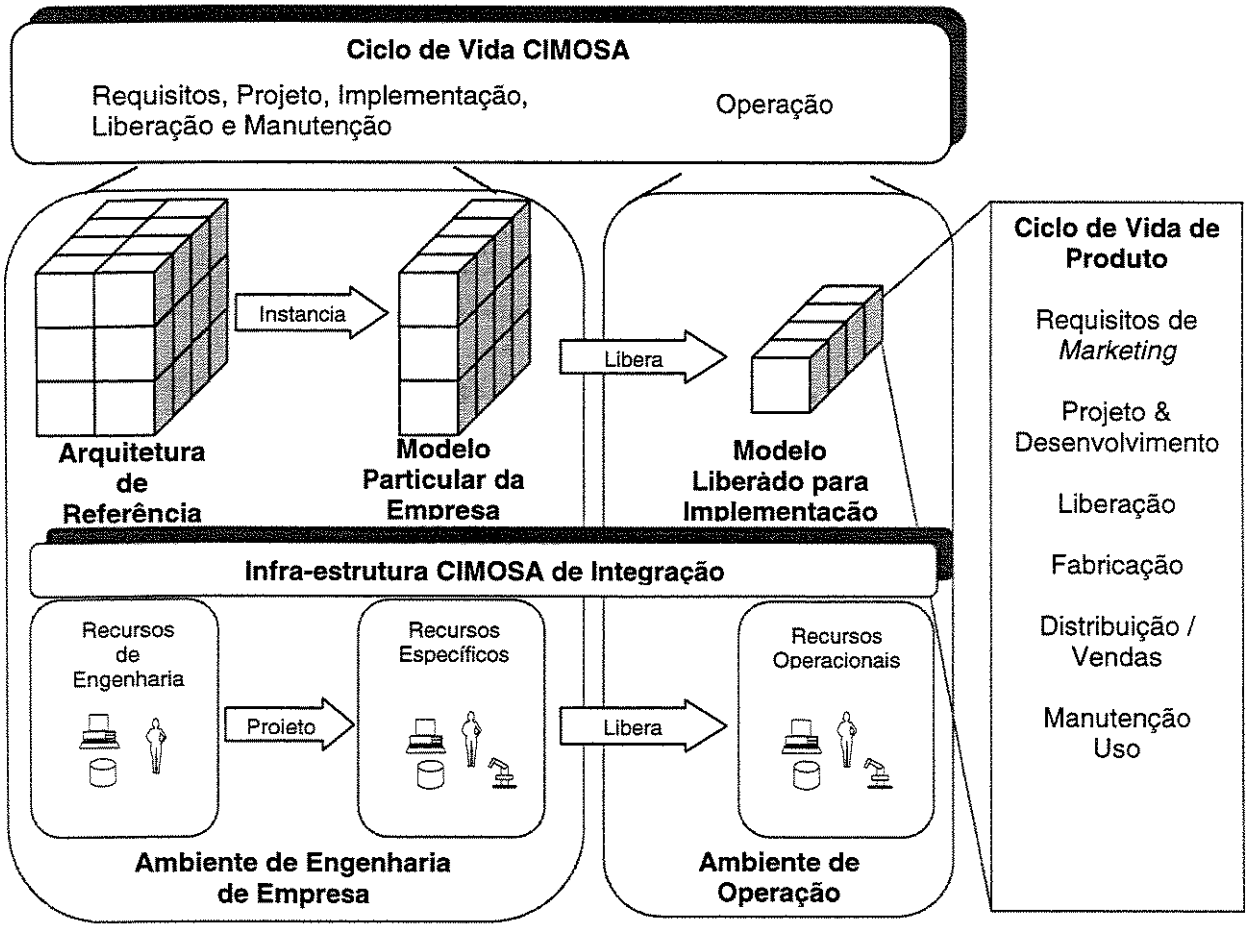


Figura 2.10: Visão Geral dos Conceitos CIMOSA **Fonte [CIMOSA 93]**

A Arquitetura Particular conterá os elementos extraídos da Arquitetura de Referência e particularizados para a empresa específica que esteja sendo modelada. Esta estratégia deve-se ao

UNICAMP
BIBLIOTECA CENTRAL
SEÇÃO CIRCULANTE

fato de que mesmo dentro de um área de negócios/ indústria especifica, existem imensas variações conjunturais e estruturais, tornando necessária a especialização das soluções.

A Figura 2.10 apresenta uma visão geral dos conceitos CIMOSA, observa-se que o ciclo de vida compreende além das fases já citadas outras para as quais não há ainda metodologias desenvolvidas. Aparece um Ciclo de Vida de Produto, que tem existência ortogonal à empresa, pois os produtos podem ser criados e descontinuados durante toda a duração da mesma.

Um dos aspectos originais de CIMOSA, de relevância para este trabalho é a definição de uma Infra-estrutura de Integração (IEI), conceituada da seguinte maneira: "é a tecnologia habilitadora que torna possível executar os modelos CIMOSA, ... através do controle e monitoração das atividades descritas nesses modelos" [CIMOSA 93].

Esta Infra-estrutura de Integração serve de suporte aos dois ambientes apresentados na Figura 2.10, o de Engenharia de Empresa e o de Ambiente de Operação. O Ambiente de Engenharia de Empresa está relacionado com a elaboração dos modelos CIMOSA e o Ambiente de Operação com a colocação em marcha desses modelos.

Esta discussão e análise da IEI CIMOSA será enfocada no item Disciplinas Relacionadas.

2.2.1.4 GERAM (*Generalized Enterprise Reference Architecture and Methodology*)

GERAM está sendo desenvolvida pela "Força Tarefa IFAC/IFIP(*International Federation of Automatic Control/ International Federation for Information Processing*) em Arquiteturas para Integração de Empresas". Este trabalho está sendo considerado para fins de padronização pela ISO, através do seu grupo de trabalho ISO/TC 184/SC 5/WG1 que cuida da preparação do "Rascunho de Norma Internacional" (*Draft International Standard*) "ISO/DIS 15704 - *Industrial automation systems - Requirements for enterprise-reference architectures and methodologies*" [ISO 98].

Como ponto de partida para a elaboração de GERAM, a Força Tarefa analisou em detalhes as arquiteturas CIMOSA, GRAI-GIM e PERA e propôs uma definição de arquitetura generalizada. Segundo seus autores "GERAM não é mais uma proposta para uma arquitetura de

referência de empresas, mas pretende *organizar o conhecimento em integração de empresas existente*" [ISO 98].

A Figura 2.11 apresenta o Quadro de referência GERAM. Verifica-se que há uma preocupação em estruturar um "Ambiente de Modelagem" onde pode-se partir de conceitos de integração de empresas e chegar até a implementação de tais conceitos, através da especialização de módulos operacionais.

A estrutura proposta é bastante complexa e ainda está em definição, far-se-á portanto uma descrição resumida e adaptada aos objetivos deste trabalho. Considerando a Figura 2.11, a descrição será pautada pelo critério de generalidade, iniciando pelos blocos GERA, GEMCs, PEMs e EMOs e descrevendo seu relacionamento com os demais.

GERA (*Generalized Enterprise Reference Architecture*), "define os conceitos genéricos recomendados para uso em engenharia de integração de empresas e projetos de integração" [ISO 98]. Este bloco tem influência da Arquitetura PERA no tocante à separação de atividades entre seres humanos e máquinas e também de CIMOSA, ao assimilar uma descrição baseada nas vistas de Função, Informação, Organização e Recursos.

Além dessa dimensão de Vistas também existem outras duas dimensões: Ciclo de Vida e Generalização. O Ciclo de Vida foi influenciado pelo da PERA (que é bastante abrangente) e a Generalização é semelhante a CIMOSA.

EEM (*Enterprise Engineering Methodology*) "descreve os processos de engenharia de empresa e integração". Estes métodos deverão permitir a progressão através do ciclo de vida do projeto de integração, da Identificação de Objetivos até a Operação e Descontinuação.

EML (*Enterprise-modelling languages*) "define os construtores de modelagem genéricos para modelagem de empresa adaptados às necessidades das pessoas que criam e usam modelos de empresa" [ISO 98]. Aspectos tais como participação humana, processos operacionais, informação e produção deverão utilizar linguagens de modelagem adequadas.

GEMCs (*Generic Enteprise Modelling Concepts*) capturam conceitos que são comuns a todas as empresas. Utilizam três formas de definição de conceitos, listadas na ordem crescente de formalismo [ISO 98].

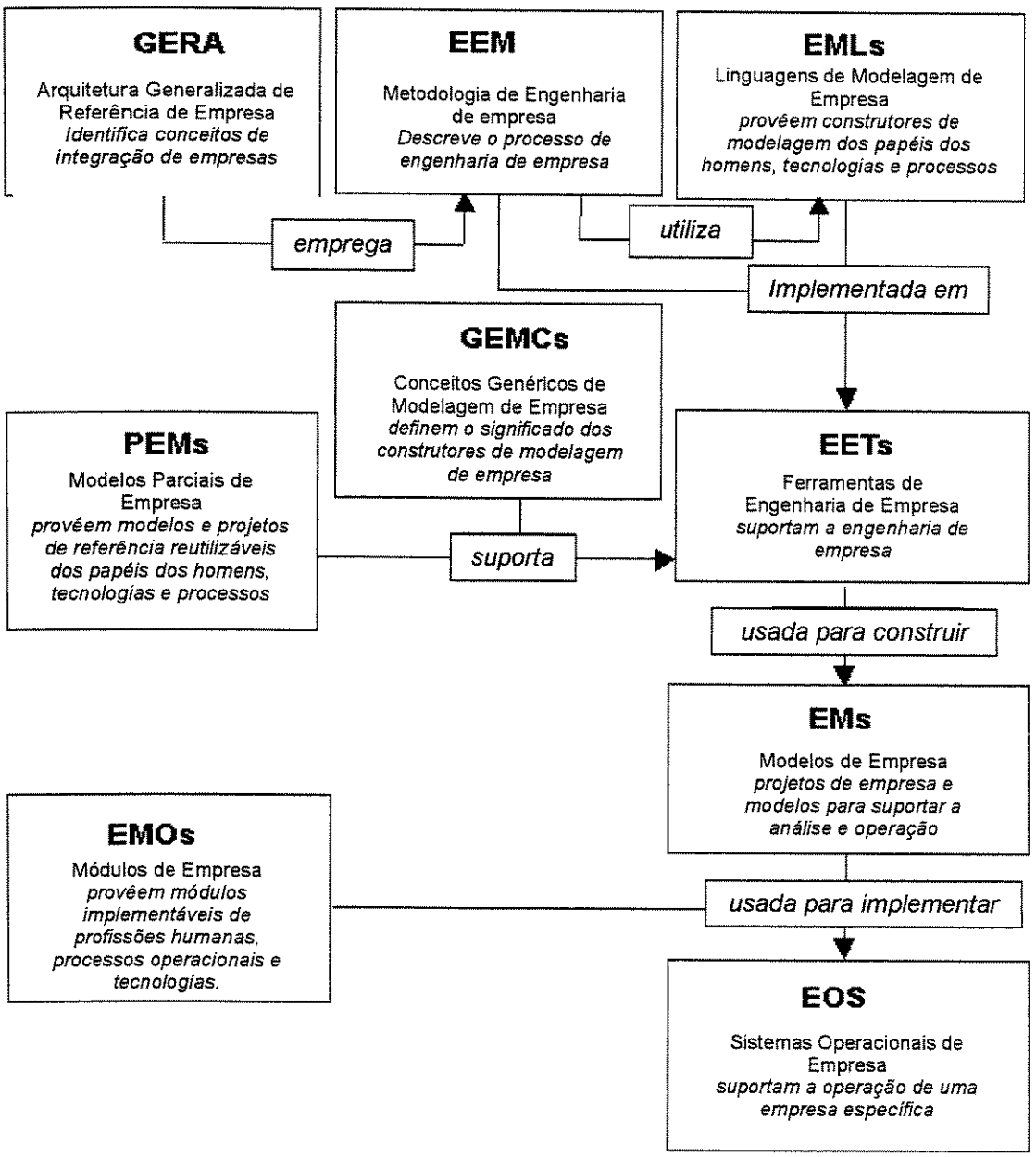


Figura 2.11: Componentes de GERAM **Fonte: [ISO 98]**

- Glossários - "terminologia usada em engenharia de empresa definida em linguagem natural".

- Meta-modelos - "modelos conceituais do componente de terminologia das linguagens de modelagem". Descreve os conceitos e seus relacionamentos.

- Teorias ontológicas - "modelos formais de conceitos usados na representação de empresa". Descrevem a semântica das linguagens de modelagem utilizadas.

PEMs (*Partial-enterprise models*) "capturam características comuns a muitas empresas dentro ou através de um ou mais setores industriais" [ISO 98]. Estes são modelos de referência reutilizáveis, os processos de engenharia de empresa podem basear-se nestes componentes testados para construir um modelo específico de empresa [Vernadat 96].

EETs (*Enterprise Engineering Tools*) "implementam uma metodologia de engenharia de empresa e suportam as linguagens de modelagem" permitindo a geração dos **EMs** (*Enterprise Models*), modelos particulares de empresa.

EMOs (*Enterprise modules*) "são blocos construtores implementados ou sistemas que podem ser usados como recursos comuns na engenharia e integração de empresa". A Infra-estrutura de Integração é composta por um conjunto destes módulos.

EOSs (*Enterprise-operational systems*) "suportam a operação de uma empresa específica. Contém todo o *hardware* e *software* necessários para cumprir os objetivos e metas da empresa" [ISO 98].

Na descrição de GERAM há um item intitulado "Conceitos orientados à Tecnologia", onde se discute o suporte de tecnologia de informação para a engenharia e integração de empresas. Neste item é apresentada, em caráter ilustrativo a infra-estrutura EMEIS (*Enterprise Model Execution and Integration Services*) sobre a qual discorrer-se-á em mais detalhe no item Infra-estruturas de Integração".

2.3 Disciplinas Relacionadas

As Arquiteturas para Supervisão e Controle de Chão de Fábrica e as Arquiteturas de Referência para Integração de Empresas são o resultado de um longo processo de amadurecimento de conhecimentos, de métodos e de tecnologias. Há um caráter multidisciplinar nessa evolução, pode-se visualizar diferentes contribuições, vindas de áreas como Gestão de Empresas, Sistemas de Informação, Sistemas de Produção e outras.

Interessa em especial investigar dois assuntos relacionados com estas arquiteturas e que serão relevantes para este trabalho: Suporte para Integração e Sistemas Inteligentes Distribuídos.

O primeiro já foi referenciado quando foram distinguidos quais eram os suportes para integração oferecidos pelas Arquiteturas de Referência, este assunto será detalhado agora. O segundo aparece claramente como alternativa para a implementação de Infra-estruturas de Integração, em especial os Sistemas Multi-agentes, que serão objeto deste estudo.

2.3.1 Suporte para a Integração

O termo Infra-estrutura de Integração, no sentido em que é proposto por CIMOSA, tornou-se corrente. Existe uma norma de comunicação industrial, chamada MMS *(Manufacturing Message Specification* - Especificação de Mensagens para a Manufatura) [MMS 90], a qual tem uma grande influência quando o assunto é comunicação em sistemas produtivos. Devido à essa importância far-se-á um breve relato do que é MMS e posteriormente mostrar-se-ão seus reflexos nas infra-estruturas de integração, quando forem apresentadas, e na proposta desta tese.

2.3.1.1 MMS - Especificação de Mensagens para a Manufatura

O Protocolo de Especificação de Mensagens para a Manufatura (MMS *Manufacturing Message Specification*) é uma norma internacional padronizada pela ISO (ISO 9506) [MMS 90]. Seu propósito é fornecer um conjunto de mensagens que permitam aos dispositivos de manufatura comunicar-se de maneira cooperativa.

A arquitetura MAP [MAP 88] utiliza na camada de aplicação o MMS , o qual foi considerado adequado às aplicações de automação industrial.

Os serviços MMS são extremamente adequados à manufatura, de tal forma que a despeito do projeto MAP não ter conseguido aceitação irrestrita, o MMS hoje é utilizado em diversos países. Os esforços de padronização para o Barramento de Campo (*FieldBus*), na forma de suas propostas regionais **FIP** [AFN 90], **PROFIBUS** [Göddertz 90] e **ISA SP50** utilizam subconjuntos do MMS para definir a camada de aplicação [Pleineveaux 94].

O MMS define com seus serviços um conjunto geral de mensagens para a manufatura; dada a diversidade de dispositivos de chão de fábrica existentes a ISO criou um conjunto complementar de protocolos, chamados de **padrões associados** (*Companion Standards*) os quais especializam os serviços do MMS para dispositivos específicos como Robôs ou Controladores Programáveis.

Alguns autores são fontes alternativas de consulta, descrevendo o MMS de maneira menos formal e mais condensada que na norma. Entre estes pode-se citar: Brill [Brill 91] e Hoekstra [Hoekstra 90].

Na descrição da estrutura do MMS a ISO utilizou duas técnicas que atualmente estão bastante disseminadas , uso da arquitetura **Cliente / Servidor** e **Modelagem por Objetos**.

Na Arquitetura Cliente / Servidor um dos lados em comunicação faz o papel de Servidor, ou seja o elemento que atende as requisições de serviços, o outro lado é o Cliente, aquele que solicita a execução dos serviços. Serviços são funções que o MMS disponibiliza para o usuário. O Cliente representa o dispositivo de manufatura que detém o controle, o Servidor é o dispositivo sob controle [Hoekstra 90].

A Modelagem por Objetos consiste em agrupar os recursos ou funcionalidades do dispositivo de manufatura em entidades, chamadas objetos, pelas quais estes recursos ou funcionalidades serão acessados por outros dispositivos ou programas de aplicação.

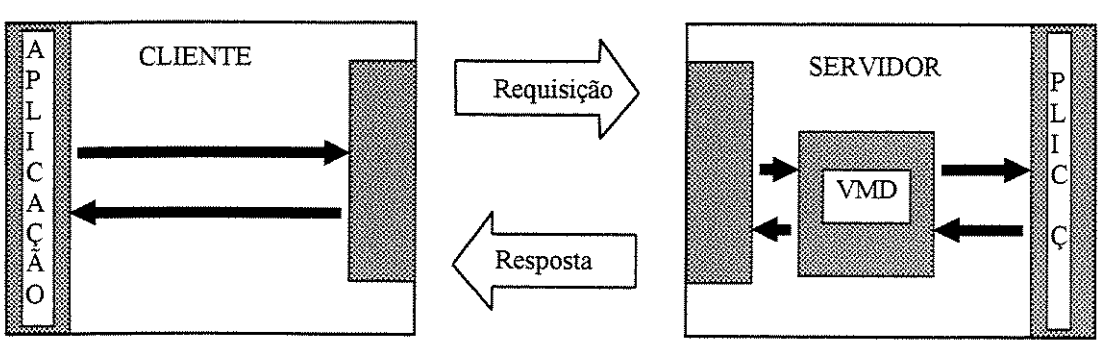


Figura 2.12: Relação Cliente/Servidor no MMS

Os serviços MMS são modelados para ler e modificar os atributos dos objetos [Hoekstra 90], eles são codificados em mensagens MMS que viajam pela rede de dispositivos.

Os objetos MMS podem ser manipulados pelos serviços, as operações normalmente possíveis são: criar, apagar, modificar ou obter os atributos de um objeto; existem algumas poucas exceções, como por exemplo o objeto **VMD** (*Virtual Manufacturing Device* - vide item seguinte), que não pode ser criado ou apagado. No modelo do MMS o VMD está colocado como um Servidor, de acordo com a Figura 2.12.

O Servidor do MMS tem seu comportamento externo modelado pelo objeto VMD (*Virtual Manufacturing Device*). O VMD torna disponível para controle e monitoração os recursos e funcionalidade associados ao dispositivo real de manufatura.

Os dispositivos reais são do tipo Controladores Programáveis (CP), Máquinas de Comando Numérico (CNC), Robôs e outros; para estes três inicialmente citados existem normas complementares ao MMS, chamados de padrões associados [CPM 89], [ROB 90], [CNC 91] as quais detalham o MMS para o dispositivo específico relativo à norma.

Serviços do MMS

Serão descritos os serviços do MMS, agrupados de acordo com sua funcionalidade em: suporte do VMD, gerenciamento de domínios, execução remota de programas, gerenciamento de contexto/conexão, acesso a variáveis, comunicação com o operador, gerenciamento de eventos e alarmes, gerenciamento de semáforos, gerenciamento de periódicos e gerenciamento de arquivos.

Quando for relevante serão listados ou explicados os serviços, caso contrário descrever-se-á apenas o grupo.

Suporte do VMD

Estes serviços permitem a um cliente MMS a determinação das características e condições gerais de um servidor MMS.

Status - Informa a um cliente MMS as condições gerais do servidor MMS, estas condições englobam os aspectos lógicos (serviços disponíveis) e físicos (estado do hardware). A resposta deste serviço inclui detalhes de padrões associados.

UnsolicitedStatus - É gerado espontaneamente pelo servidor MMS para informar as suas condições; no mais é idêntico ao serviço de Status.

GetNameList - Através deste serviço um cliente MMS pode solicitar, ao servidor MMS, a devolução de uma lista de nomes de objetos que cumpram os requisitos especificados no serviço. Estes requisitos devem incluir, por exemplo, a classe do objeto (*Named Variable*, *Semaphore*, etc.).

Identify - Fornece a informação de identificação do servidor MMS, a qual contem: o nome do fabricante, nome do modelo e o nível de revisão de sistema em utilização.

Rename - Usado por um cliente MMS, para pedir a modificação do identificador de um objeto, para um novo identificador especificado.

Gerenciamento de domínios

Um domínio representa um subconjunto de recursos de um VMD que é usado para um propósito específico. Exemplos de domínios são programas, variáveis e dados.

Execução remota de programas

Entre os vários objetos definidos no MMS, para o modelo do VMD, existe o que se chama *Invocação de programa*, que é o agrupamento de um ou mais domínios na forma de um programa executável. Os serviços que são oferecidos para o cliente gerenciar estes programas são listados a seguir:

CreateProgramInvocation, DeleteProgramInvocation,

Start, Stop, Resume, Reset, Kill,

GetProgramInvocationAttributes

Acesso às variáveis

O objetivo dos serviços de acesso às variáveis do MMS é permitir a leitura e escrita de variáveis contidas em um VMD. Os serviços básicos são:

Read, Write, InformationReport,

Existem serviços complementares para definir características das variáveis.

Gerenciamento de contexto/conexão

Estes serviços permitem que os pares de Processos de Aplicação(o iniciador e o receptor da comunicação) negociem, mantenham, liberem e abortem uma conexão lógica.

Comunicação com o operador

Estes serviços permitem a comunicação com uma estação de operador, que aceite entrada de dados, apresentação de dados ou ambas as coisas.

Gerenciamento de eventos e alarmes

Permitem gerenciar eventos e alarmes.

Serviço de gerência de semáforos

Estes serviços permitem a sincronização, controle e coordenação de recursos compartilhados entre usuários do MMS.

Gerenciamento de periódicos

Existem para prover facilidade no armazenamento e recuperação de informação cronologicamente ordenada que diz respeito a eventos, variáveis associadas a eventos ou ambos e texto, que é utilizado como anotação explicativa ou comentário.

Gerenciamento de arquivos

São utilizados na leitura de arquivos, para gerenciar armazenadores de arquivos, renomear e

apagar arquivos. Não são utilizados para acesso aleatório ou modificação nos arquivos. Os serviços existentes no MMS para gerenciar arquivos são um subconjunto da norma ISO FTAM *(File Transfer, Access and Management)*.

Os mecanismos para a criação dos padrões associados podem ser usados também para a criação de um VMD de um dispositivo qualquer, assim Rondeau [Rondeau 95], criou VMDs que foram usados pela Companhia de Metrô de Paris, através de um aprimoramento da análise dos objetos sob a ótica da metodologia entidade-relacionamento.

2.3.2 Infra-estruturas de Integração (IEI)

O objetivo das IEI é dar suporte para a integração de sistemas, permitindo a integração de recursos heterogêneos e provendo meios para controlar os processos envolvidos [CIMOSA 93]. Em [Vernadat 96] encontra-se uma apresentação das IEIs: CIMOSA, CIM-BIOSYS, CCE-CNMA, AMBAS, PACT e SHADE. Nesta tese o enfoque é nas IEIs mais formalizadas e disponibilizadas na literatura, neste sentido descrever-se-ão a IEI CIMOSA e a EMEIS. O leitor mais interessado pode consultar a referência indicada para as outras IEIs.

2.3.2.1 IEI CIMOSA

A IEI CIMOSA é baseada no fornecimento de um conjunto de serviços [Lapalus 95], [Querenet 92], [CIMOSA 93], conforme representado na Figura 2.13. Estes serviços são estruturados de acordo com o modelo Cliente-Servidor já apresentado na Figura 2.12. A descrição seguinte é extraída de [CIMOSA 93], serão adicionados comentários e outras referências quando necessário para esclarecer o texto.

a) **Serviços de Gerência de Sistema**: o propósito destes serviços é prover funções genéricas para ativar, manter e monitorar os componentes da IEI. A funcionalidade proposta suporta:

   - distribuir e instalar novas versões da IEI
   - instalar e configurar estes componentes para operação.
   - iniciar e operar estes componentes

- reconfigurar e relatar sobre aspectos de segurança
- coletar erros e medir a performance dos componentes

Nenhum serviço está especificado para a Gerência de Sistema [Vernadat 96].

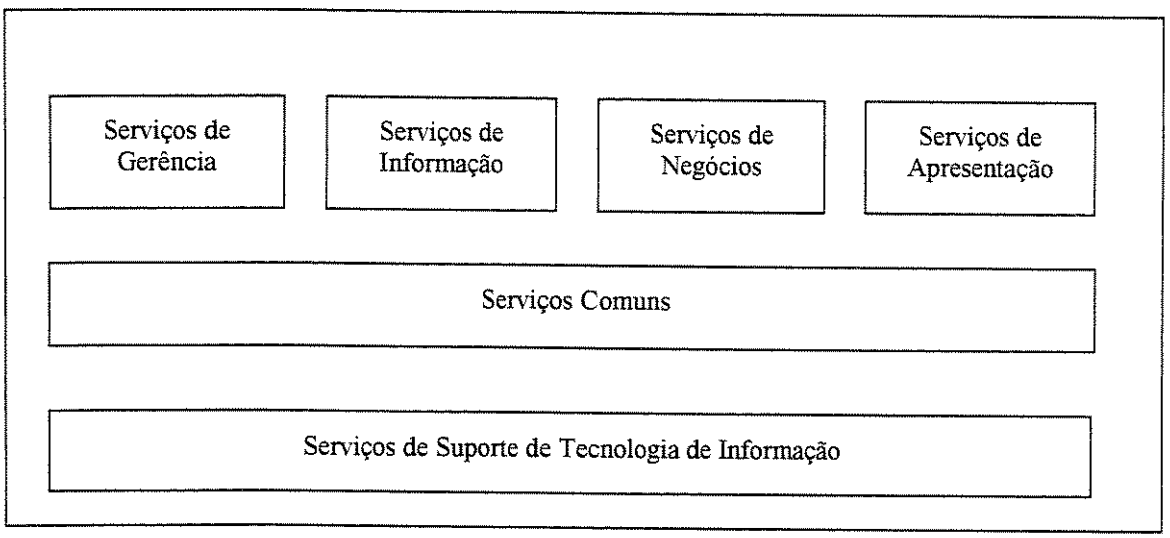


Figura 2.13: Infra-Estrutura de Integração CIMOSA Fonte:[CIMOSA 93]

b) **Serviços de Negócio**: o propósito é prover funções genéricas para iniciar, monitorar e controlar as operações da empresa, de acordo com o modelo particular da mesma, através do processamento do "modelo executável" correspondente. Eles suportam:

- operações da empresa ativadas por eventos
- coordenação, seqüenciamento e sincronização de operações da empresa
- gerência de recursos da empresa
- mudanças flexíveis das operações da empresa
- permitir a intervenção humana para lidar com eventos excepcionais (com o auxílio dos "Serviços de Apresentação")

Antes de continuar a descrição desses serviços é necessário apresentar algumas definições que serão utilizadas no texto:

Eventos são acontecimentos não solicitados que acontecem no mundo real e têm influência sobre a empresa.

Processo do Domínio são processos isolados ativados tão somente por Eventos e produzindo resultados finais bem definidos. Encapsulam um conjunto de funcionalidade e comportamento da empresa, com o fim de realizar objetivos de negócios.

Processo do Negócio são processos definidos pelo usuário que contém parte do comportamento da empresa. Podem ser decompostos recursivamente em Processos do Negócio ou em Atividades de Empresa.

Atividades de Empresa definem a funcionalidade da empresa na forma de tarefas elementares definidas por suas entradas e saídas, função e as habilidades necessárias.

Regras de Comportamento definem o fluxo de ações que descreve o comportamento de um Processo de Negócio ou de um Processo de Domínio.

Os Serviços de Negócio, para permitir a execução do modelo de empresa, necessitam gerenciar a execução dos Processos de Negócio, das Atividades da Empresa e a reserva e alocação de recursos; eles incluem:

Controle de Processos do Negócio - responde à ocorrência de Eventos através da interpretação das Regras de Comportamento definidas nos Processos do Negócio [Vernadat 96], demanda à "Gerência de Recursos" o escalonamento, reserva e alocação dos recursos necessários [Didic 95].

Controle de Atividades - controla a execução das Atividades da Empresa despachando as Operações Funcionais para serem executadas pelos recursos. É o servidor do "Controle de Processos do Negócio", dialoga com os Serviços de Apresentação (ver a seguir) os quais acessam os recursos físicos [Didic 95].

Gerência de Recursos - reserva e escalona dinamicamente os recursos usados pelas Atividades da Empresa para realizar as suas Funções Operacionais. Realiza estas funções sob demanda do "Controle de Processos de Negócio" e do "Controle de Atividades" [Didic 95].

c) **Serviços de Informação** - provêem funções genéricas para acesso, integração e manipulação de dados. A proposta CIMOSA é atuar nos dados de acordo com as normas SQL (*Structured Query Language*) e RDA (*Remote DataBase Access*).

d) **Serviços de Apresentação** - provêem as funções necessárias para controlar a execução da Funções Operacionais pelos recursos da empresa, os quais são do tipo máquinas, pessoas e aplicações. A funcionalidade destes serviços permite:

- prover informações de estado das Funções Operacionais e das Entidades Funcionais alocadas às Funções Operacionais.
- alocar e liberar Recursos de e para as Funções Operacionais
- preparar os Recursos para permitir a execução das Funções Operacionais

O Serviços de Apresentação utilizam funções semelhantes às da norma **MMS**, elas são divididas em três grupos:

- Preparação de Operações - funções Criar Objeto de Serviço de Operação Funcional, Apagar Objeto de Serviço de Operação Funcional, Iniciar Objeto de Serviço de Operação Funcional
- Gerência de Recursos - Atribuir Recurso para Operação Funcional e Liberar Recurso de Operação Funcional
- Controle de Operações - Executar Operação Funcional, Parar Operação Funcional, Reativar Operação Funcional, Abortar Operação Funcional e Terminar Operação Funcional.

e) **Serviços Comuns** - provêem soluções comuns para assuntos relacionados com os outros serviços. Estas questões são entre outras a da distribuição de componentes e a da comunicação entre estes componentes. Segundo [Lapalus 95], a funcionalidade destes serviços permite:

- ativação de uma aplicação
- estabelecimento de uma associação entre duas aplicações
- envio e recebimento de requisições e respostas entre as aplicações
- gerência de comunicação.

Do ponto de vista de implementação de sistemas CIMOSA, é relatado o uso de Gerência de Comunicação baseado em protocolos ISO [Lapalus 95].

Implementação da IEI CIMOSA

O Projeto McCIM foi realizado durante a fase de validação de CIMOSA, através dos projetos Europeus VOICE (EP 5510) e VOICE II (EP 6682) e tratou do desenvolvimento de uma IEI CIMOSA [Didic 93], [Didic 95]. Ele foi aplicado experimentalmente em uma Planta de Produção de Alumínio (ELVAL) [Arabatzis 95], as conclusões apontadas indicam que apesar dos conceitos CIMOSA serem considerados muito interessantes a sua aplicação prática ainda era difícil. Foi feita uma recomendação para a atualização dos serviços, buscando compatibilizá-los com as normas dos Sistemas Abertos.

2.3.2.2 EMEIS *Enterprise Model Execution and Integration Services*

A descrição da EMEIS é baseada na pré-norma européia prENV 13550 de 1999 [prEnv 99]. A Figura 2.14 apresenta a estrutura da EMEIS.

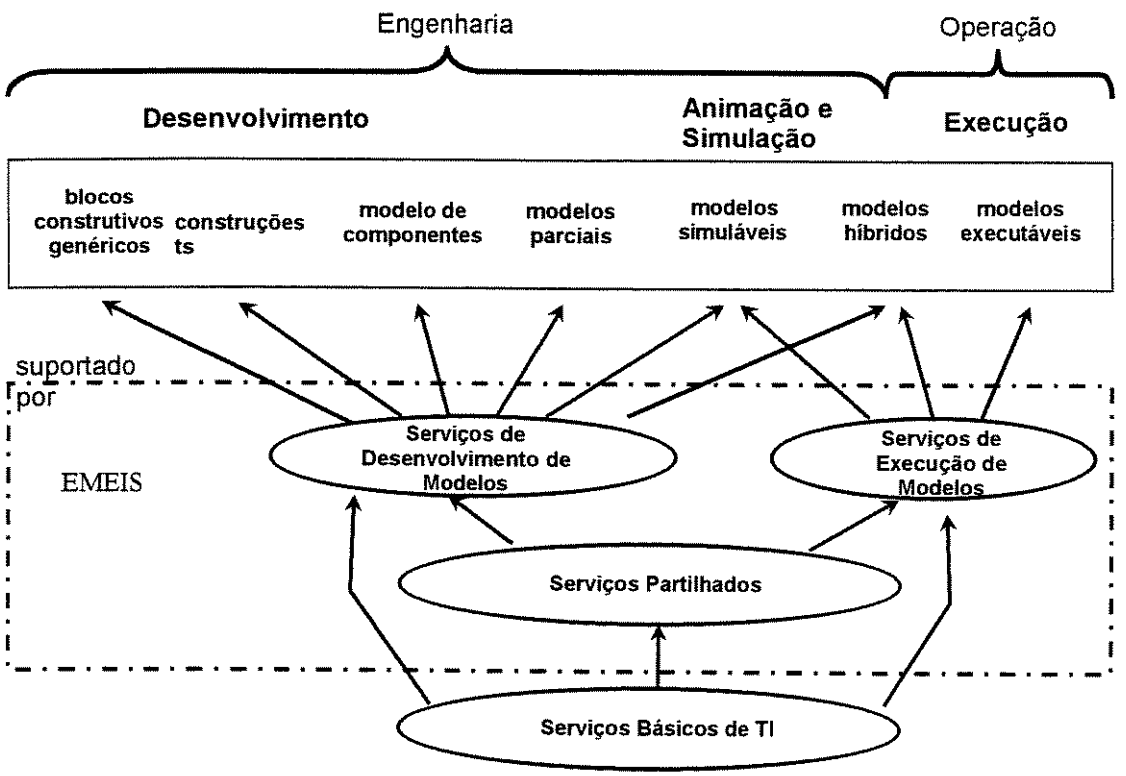


Figura 2.14: EMEIS **Fonte: [prENV 99]**

O propósito da EMEIS é dar suporte para permitir o desenvolvimento de modelos, sua animação e simulação e finalmente propiciar um ambiente para sua execução. Ela é organizada em três partes:

- Serviços de Desenvolvimento de Modelos - permitem o desenvolvimento cooperativo, análise , gerência e liberação de componentes de modelos, modelos parciais e modelos executáveis particulares de entidades de negócio de uma empresa.

- Serviços de Execução de Modelos - suportam o uso e operacionalização de modelos

- Serviços Partilhados - são aplicáveis às duas classes de serviços anteriores e tratam dos aspectos de Confiabilidade, Segurança, Negociação, Encapsulamento e Distribuição.

a) Serviços de Desenvolvimento de Modelos

Para o desenvolvimento de modelos são supostas quatro fases:

A fase 1 , "Definição de Domínios" está no nível global da empresa, considerando os domínios da empresa e as relações entre eles e com a Unidade Organizacional responsável.

A fase 2, "Definição de Processos" identifica o contexto dos processos, seu comportamento e demais detalhes. Os processos têm objetivos que se relacionam com suas entradas/saídas e com a Unidade Organizacional responsável.

A fase 3, Definição de Atividades, relaciona-se com as necessidades dos processos, em como eles adquirem suas entradas e como eles fornecem suas saídas e também relaciona-se com a Unidade Organizacional responsável.

A fase 4, "Definição da Vista de Modelo", preocupa-se com a construção de estruturas não redundantes (modelos e sub-modelos). Pode incluir modelos parciais e blocos construtivos já existentes. Os aspectos de Funcionalidade, Controle, Recursos e Entradas e Saídas são contemplados através do uso de Vistas de Objeto (sobre estes elementos).

São previstos quatro serviços:

Criação de Modelos, Avaliação de Modelos, Gerência de Modelos e Repositório de Modelos

b) Serviços de Execução de Modelos

Suportam a execução de um modelo ou de seus componentes. A execução de um modelo simulável é vista como um caso particular de execução no qual as saídas do modelo não afetam as operações da empresa diretamente. O serviços são compatíveis com as restrições para execução de serviços colocadas pela norma ENV 12204 [prENV 95].

Os serviços são:

Gerência de Processos - permitem registrar e tratar Eventos; escalonar Processos de Negócios, interpretar a regras destes processos e monitorar as condições necessárias para o serviço de interpretação de regras. Permitem também gerenciar Recursos executar Atividades de Empresa sob demanda do serviço de interpretação de regras.

Gerência de Apresentação - são responsáveis por fazer o mapeamento entre os comandos internos do modelo executável e os comandos e respostas disponíveis no mundo real (Recursos). Possui um serviço geral para interação com o monitoramento de condições e a gerência de atividades de Empresa e três serviços específicos para: Diálogo com Humanos, Diálogo com Máquinas, Diálogo com Aplicações.

Gerência de Informação - provê acesso ao sistema global de informações e aos serviços eletrônicos de trocas de dados (por ex. : EDI, STEP e SGML).

Repositório "*Run-Time*" de Modelos - mantém um registro (no momento da execução - *Run-Time)* das entidades envolvidas na execução de modelos.

c) Serviços Compartilhados

São compostos pelos serviços que são relevantes para os Serviços de Desenvolvimento de Modelos e Serviços de Execução. Eles são os serviços de:

Confiabilidade, Segurança, Contrato e Intermediação, Encapsulamento, Distribuição de Modelos.

2.3.3 Sistemas Multi-agentes

Sistemas Multi-agentes (SMAs) são um dos ramos da Inteligência Artificial, sendo esta uma área do conhecimento onde ainda persistem indefinições de nomenclatura, talvez por coexistirem enfoques conceituais oriundos de áreas diversas. O interesse é na aplicação de sistemas multi-agentes nos sistemas de manufatura, os conceitos, definições e exemplos relacionados a esse domínio serão objeto de atenção.

Tentar-se-á delinear um quadro conceitual que permita posicionar os sistemas multi-agentes de maneira a apontar os aspectos mais relevantes para o seu uso em um sistema de supervisão e controle de produção.

Jennings *et al.* [Jennings 98], posicionam os SMAs dentro da Inteligência Artificial Distribuída, em conjunto com os sistemas para Resolução Distribuída de Problemas. Ao mesmo tempo apontam as áreas de Programação Orientada a Objeto e Projeto de Interface Homem Computador como tendo influência no desenvolvimento dos SMAs.

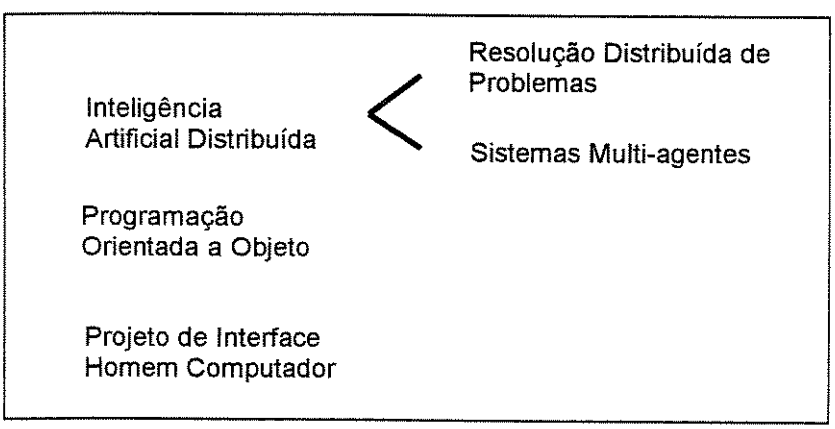


Figura 2.15: Influências sobre os sistemas multi-agentes. Adaptado de [Jennings 98]

Esta opinião também é compartilhada por Gasser [Briot 98], que adiciona a influência da área de Trabalho Cooperativo Suportado por Computador (CSCW). A Figura 2.15 ilustra estas

influências.

2.3.3.1 O que é um Agente?

Esta é uma questão tão polêmica quanto aquela da definição do termo "Inteligência" na área de Inteligência Artificial. Segundo Jennings, Sycara e Wooldridge [Jennings 98] este não deve ser um grande obstáculo ao progresso na área já que a Inteligência Artificial desenvolveu-se mesmo sem uma definição precisa do que é Inteligência.

Franklin e Graesser [Franklin 96] descrevem várias definições para qualificar Agentes e Agência e propõem uma taxinomia para os Agentes Autônomos. É desnecessário polemizar sobre o tema, partir-se-á de uma definição geral de agente para realizar este trabalho.

Segundo o "Dicionário da Língua Portuguesa" [Ferreira 1985], agente é "aquele que trata de negócio por conta alheia", na quarta acepção apresentada para o termo. É interessante observar que na língua inglesa esta acepção á a mais corrente, conforme vê-se em: "*A person who acts for, or manages the affairs of, other people in business, politics, etc.*" apresentada pelo "*Oxford Advanced Learner's Dictionary*" (Oxford University Press 1989).

Katia Sycara *et al.* [Sycara 96], dizem que "Agentes Inteligentes (de *Software*) são programas que podem agir por conta de seus usuários...", Charles Petrie [Petrie 96], argumenta que alguns consideram Agentes como sinônimo de "Agentes Autônomos" e que a característica de autonomia é relevante para definir Agentes.

Para Jennnings, Sycara e Wooldridge [Jennings 98], autonomia significa que "o sistema deve ser capaz de agir sem a intervenção direta humana (ou outros agentes) e devem ter controle sobre suas próprias ações e estados internos".

2.3.3.2 Vantagens dos Sistemas Multi-agentes

Bradshaw [Bradshaw 97], apresenta uma introdução aos Agentes de *Software*, onde destaca algumas vantagens dos SMAs, combinando-se com a discussão de Parunak [Parunak 98] pode-se apresentar a lista da Figura 2.16

modulares descentralizados / distribuídos modificáveis suportam sistemas complexos suportam especificações incompletas realizam tarefas delegadas melhoram as interfaces com usuários

Figura 2.16: Vantagens dos Sistemas Multi-agentes

Os SMAs são naturalmente distribuídos e essa característica facilita a modularização. A sua autonomia em conjunto com a distribuição permite que lidem com sistemas complexos e eles podem ser atualizados para atender aos novos requisitos de um sistema mutável.

Eles podem operar mesmo com especificações incompletas e apresentam uma capacidade de escalabilidade e evolução. Eles melhoram as interfaces com os humanos, realizando automaticamente tarefas que lhes sejam delegadas.

2.3.3.3 Aspectos relevantes da arquitetura de SMAs

Após uma revisão da literatura que incluiu entre outros [Parunak 98], [Bradshaw 97], [Jennings 98], [Genereseth 94], [Doran 96], [Nwana 96], [Wooldridge 94] buscou-se definir um subconjunto de aspectos dos SMAs, que fossem relevantes para o seu projeto e construção, a tônica desta classificação é dada por Parunak [Parunak 98]. Na Figura 2.17 apresentam-se os cinco aspectos básicos sobre os quais ocorrerá a discussão.

A **Configuração** de um SMA define uma topologia [Parunak 98], indicando como os agentes estão arranjados uns em relação aos outros. Esta Configuração pode ser estática ou dinâmica, neste último caso agentes podem ser adicionados ou retirados do sistema juntamente com suas conexões com os outros agentes.

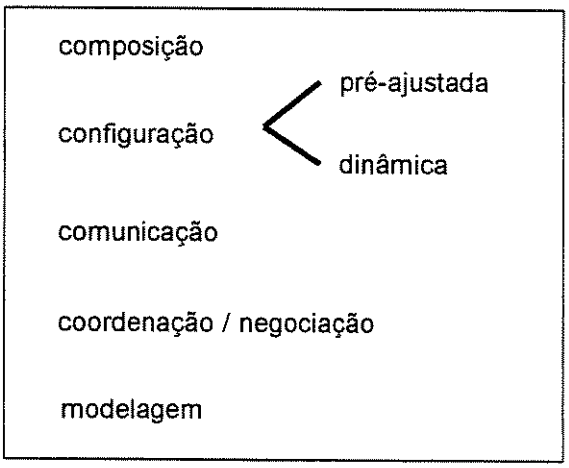


Figura 2.17: Aspectos da Arquitetura dos SMAs

A **Composição** (Figura 2.18) indica se os agentes são uniformes ou não e qual é o seu escopo de atuação. Os agentes podem ser de um mesmo tipo, o que caracteriza um sistema homogêneo ou de tipos diferentes, caracterizando um sistema heterogêneo.

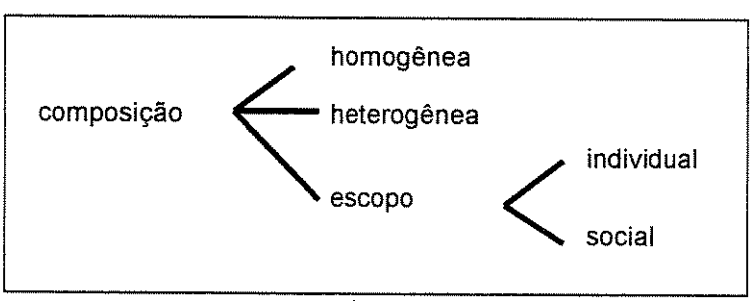


Figura 2.18: Composição dos SMAs

A **Comunicação** (Figura 2.19) pode ser classificada pelo seu conteúdo e pelo suporte que a permite (ou forma de comunicação). Quanto ao conteúdo ela pode ser baseada em diretivas (ou ordens) que determinam o que deve ser feito; por alguma forma de votação ou pela emissão de "atos discursivos" (*Speech Acts*) [Cohen 95] em decorrência dos quais uma atividade desejada pode ou não ser realizada, o que os diferencia das diretivas. Michael Wooldridge, em sua tese de doutorado [Wooldridge 92], fornece uma descrição formal para os "atos discursivos" e analisa suas diversas variantes.

Em relação ao suporte que permite a comunicação pode-se distinguir a comunicação que usa o próprio ambiente, sendo que os agentes sentem o que acontece e agem sobre este ambiente (através de sensores e atuadores) não se comunicando diretamente; a comunicação através de partilha de dados, os chamados "Sistemas de Quadro Negro", onde a comunicação se dá pela leitura e escrita em uma área de dados comum e finalmente pode-se apontar a comunicação baseada em mensagens.

Um sistema que utilize troca de mensagens necessariamente necessitará especificar um protocolo e uma linguagem de comunicação, os quais definem as regras que regem a comunicação e o conjunto de mensagens. Para isso é necessário considerar os aspectos de estrutura sintática das mensagens e também os aspectos semânticos de interpretação das mesmas. Genereseth e Ketchpel [Genereseth 94] levantam três questões importantes sobre o desenvolvimento de SMAs:

"Qual é uma linguagem apropriada de comunicação para agentes?

Como construir agentes capazes de se comunicar com essa linguagem?

Qual arquitetura de comunicação conduz à cooperação?"

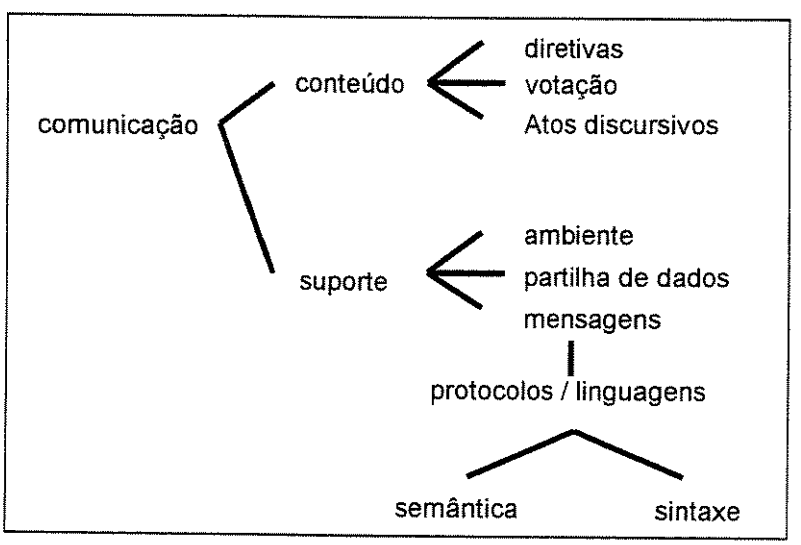


Figura 2.19: Comunicação entre Agentes

A resposta sugerida por eles passa por uma linguagem de estruturação do conhecimento chamada KIF (*Knowledge Interchange Format*) que serve para definir os elementos de

comunicação (vocabulário) os quais são trabalhados por uma linguagem chamada KQML (*Knowledge Query and Manipulation Language*) [Finim 93]. O significado das mensagens sendo trocadas depende do domínio do conhecimento, os mesmos termos podem ter significados diferentes quando são usados em áreas / contextos diferentes [Peng 98].

Para formalizar os conhecimentos dentro de uma certa área do conhecimento podem ser definidas Ontologias [Gruber 93], [Vernadat 96]. No domínio de integração de empresas foram realizados alguns trabalhos [Fox 92], [Uschold 98] mas ainda resta muito por fazer, especialmente quanto à definição das estruturas que representem o chão de fábrica.

A **Coordenação** , Figura 2.20, está relacionada com o comportamento social dos agentes, isto é com as relações que são desenvolvidas entre eles para atingir os objetivos do sistema. É possível caracterizá-la segundo duas visões, de acordo com o tipo de coordenação, que pode ser baseada na cooperação ou competição entre os agentes, ou de acordo com a estrutura do sistema, que pode ser hierárquica ou heterárquica. As definições de hierarquia e heterarquia apresentadas no item 2.1.2 Tipos de Arquitetura, são extensíveis aos SMAs.

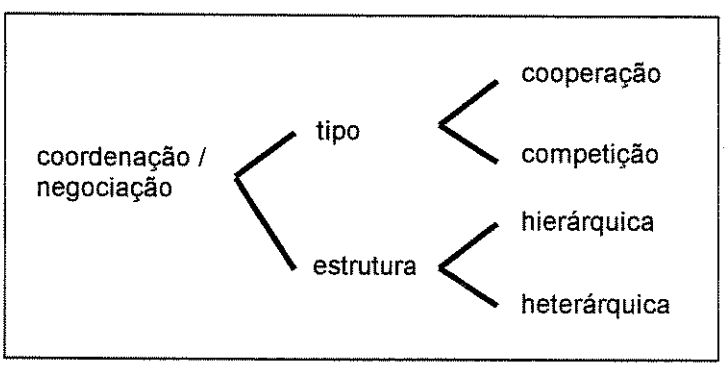


Figura 2.20: Coordenação/ Negociação entre Agentes

A questão da cooperação e coordenação requer que os agentes sejam capazes de comunicar-se, através de umas das formas descritas anteriormente. O relacionamento entre eles pode ser do tipo cooperativo ou competitivo, em ambos os casos será necessário o uso de um protocolo de comunicação, que terá também a função de coordenação/ negociação. O "Protocolo de Rede de Contratos" *(Contract-Net Protocol)* [Smith 80] , em suas variantes , é um dois mais largamente utilizados para coordenação dos SMAs. Parunak [Parunak 98] argumenta que protocolos de negociação (do tipo Rede de Contratos) possuem um conjunto fixo de opções e que

protocolos baseados em "Atos Discursivos" (*Speech Acts*) permitem maior generalidade.

A **Modelagem** dos SMAs está relacionada ao paradigma de construção dos agentes utilizados. Wooldridge e Jennings [Wooldridge 94], quando falam de "arquiteturas de agentes", classificam-nas em três classes: Deliberativas, Reativas e Híbridas.

As **Arquiteturas Deliberativas** são oriundas do enfoque clássico de Inteligência Artificial, que busca uma representação lógica do problema em mãos. Descobriu-se que a aplicação diretas das teorias lógicas não era factível para o caso geral, foram então desenvolvidas variantes como por exemplo os Sistemas Baseados em Planejamento e os Sistemas BDI *(Belief, Desire, Intention)*. Os Sistemas Baseados em Planejamento procuram propor alguma forma de decomposição dos objetivos gerais dos sistemas em objetivos locais dos agentes os quais devem seguir determinados planos de ação.

Os sistemas BDI procuram atribuir "atitudes mentais" [Rao 95] aos agentes. BDI ("Crença, Desejo e Intenção") para seus autores, Rao e Georgeff [Rao 95] estão associados aos aspectos informacionais, motivacionais e deliberativos do sistema. Para Parunak [Parunak 98], "crenças" são modelos do mundo em seu estado atual; "desejos e intenções" têm a ver com o estado futuro para o qual se pretende conduzir o sistema.

As **Arquiteturas Reativas** são aquelas que não incorporam nenhum tipo de modelo simbólico de mundo e não usam nenhum raciocínio simbólico complexo [Wooldridge 94]. Os primeiros trabalhos nessa área foram realizados por Rodney Brooks do MIT , o qual desenvolveu o conceito de "*subsumption architecture*" [Brooks 86], [Brooks 89]. Brooks desenvolveu um modelo em camadas, em que cada camada é capaz de realizar uma certa funcionalidade e pode beneficiar-se das outras camadas para atingir seus propósitos. Ele demonstrou sua arquitetura utilizando robôs que realizam ações autônomas.

As **Arquiteturas Híbridas** contém características de deliberação e reação, buscando obter um melhor desempenho do sistema. Usualmente os componentes reativos estão mais próximos do sistema físico.

Vários exemplos das arquiteturas Deliberativa, Reativa e Híbrida podem ser encontrados em [Wooldridge 94].

2.3.3.4 Aplicações de SMAs

A aplicação das técnicas de Inteligência Artificial na Manufatura inicialmente centrou-se no uso de Sistemas Peritos (*Expert Systems*). Heragu e Kusiak [Heragu 87] descrevem a estrutura dos sistemas peritos e abordam três aspectos relevantes para os mesmos: Representação de Conhecimento, Aquisição de Conhecimento e Estratégia de Inferência. Eles resenham uma série de sistemas que foram aplicados aos seguintes problemas: Projeto de Componentes, Planejamento de Processos, Seleção de Equipamentos, Estruturação Física (*Layout*) de Fábrica. Ainda Kusiak [Kusiak 88] apresenta mais quatro classes de problemas de manufatura que também se beneficiaram desses sistemas: Seleção de trajeto de corte de ferramenta, Seqüenciamento de trajetos de corte, Tecnologia de Grupo e Escalonamento de Sistemas Flexíveis de Manufatura.

Um histórico da evolução dos Sistemas Peritos é apresentado em [Gaines 88]. Muitos dos conceitos aplicados ao atuais SMAs começaram a ser pesquisados nessa época (década de 80), por exemplo [Yang 85] apresenta um estudo que se centra nos aspectos de comunicação e controle dos Sistemas de Inteligência Artificial Distribuída.

Nesta mesma época começam a emergir os SMAs com aplicações de controle. Dois exemplos, o DVMT (*Distributed Vehicle Monitoring Testbed*) e o YAMS (*Yet Another Manufacturing System*) são apresentados em [Parunak 88].

YAMS parte de uma estrutura hierárquica de controle, baseada no modelo NBS, e define seus agentes de acordo com os níveis da mesma. Utiliza um esquema de negociação entre os agentes baseado na Rede de Contratos [Smith 80], cada agente que compõe a arquitetura pode pertencer a três classes: Gerente, que identifica a tarefa a ser realizada e a atribui a outro agente para execução; Ofertadores, que ofertam-se para realizar a tarefa e Contratadores, os quais são Ofertadores bem sucedidos, isto é aqueles cuja oferta foi aceita pelo Gerente.

O Projeto PACT [Tenenbaum 92] é uma federação de agentes que permite a disposição de aplicativos (ferramentas) na Internet. PACT utiliza o conceito de "Facilitador", este é um agente especial, através do qual a comunicação entre os demais agentes é realizada (os outros agentes não comunicam-se diretamente). PACT utiliza a linguagem de representação de conhecimento KIF, o protocolo de comunicação KQML e ontologias desenvolvidas no projeto associado

SHADE (*Shared Dependency Engineering*) e foi aplicado a um sistema demonstrativo de projeto de manipulador robótico. De acordo com Vernadat o Projeto PACT é "inovador no sentido em que ele coloca menos ênfase na troca de mensagens e mais na partilha e troca de conhecimentos..." [Vernadat 96].

AARIA (*Autonomous Agents for Rock Island Arsenal*) [Baker 97], [Parunak 97], é um SMA para a manufatura discreta de componentes. AARIA combina agentes do tipo Recursos (pessoas, máquinas, facilidades), Tipos de Componentes e Processos Unitários (conceituado como sendo o conhecimento de como combinar recursos e componentes para fazer outros componentes). Um aspecto interessante de AARIA é que ele está sendo projetado para facilitar a simulação de fábricas, outro aspecto é que um estudo sobre algoritmos de escalonamento adequados às novas estruturas de produção está sendo desenvolvido conjuntamente com a arquitetura.

Diversos outros sistemas, de interesse histórico e/ ou com aplicações industriais, foram resenhados em [Parunak 98] e [Shen 99]. Outros trabalhos apresentam os SMAs em geral, por exemplo [Nwana 96] e [Wooldridge 94].

Universidades e Centros de Pesquisa em todo o mundo vêm desenvolvendo pesquisas na área de SMAs e assuntos relacionados.

Pode-se citar o Projeto Aglets da IBM Japão, relacionado aos agentes móveis [Oshima 98] baseados em Java. Também são baseados em Java o Projeto JATLite, desenvolvido na Universidade de Stanford [Jeon 00], [Petrie 96] e o Projeto JAFMAS [Chauhan 97] da Universidade de Cincinnati ambos nos Estados Unidos.

Da Universidade de Toronto, Canadá, destacam-se os trabalhos sobre Modelagem de Empresa (projeto TOVE) [Fox 92], linguagem COOL para coordenação de sistemas Multi-agentes [Barbuceanu 96a] e ambiente para construção de agentes ABS [Barbuceanu 96b].

Da Universidade de Massachusetts vem a arquitetura JAF [Horling 98] baseada em componentes Java com aplicação no projeto de casa inteligente IHOME [Lesser 99].

Especificamente considerando a problemática dos sistemas de produção pode-se apontar:

- Projeto CIM-BIOSYS da Universidade de Loughborough, Inglaterra [Aguiar 94] e o Projeto HOLOS da Universidade Nova de Lisboa, Portugal [Rabelo 95], cuja originalidade é apresentar uma infra-estrutura baseada em agentes parcialmente inspirada na Arquitetura CIMOSA [Rabelo 94a].

- Da Universidade de Calgary, Canadá, as contribuições são através do "Ambiente para Projeto Concorrente baseado em Agentes" ABCDE [Balasubramaniam 95] e através de uma resenha sobre o estado da arte dos SMAs aplicados à Manufatura [Shen 99].

- Da Universidade de Maryland veio o Projeto CIIMPLEX [Peng 98] que busca fazer a conexão entre o Planejamento da Produção e a execução deste planejamento, além da linguagem para comunicação de agentes KQML [Finin 93].

- A Universidade Católica de Louvain, Bélgica, tem grande contribuição em trabalhos sobre Sistemas Holônicos [Brussel 95], [Brussel 98] e aplicação dos seus conceitos no Projeto MASCADA no qual um dos propósitos é "desenvolver um sistema de controle da manufatura baseado em agentes e uma metodologia de projeto ..." [Brückner 98].

- A Universidade de Aix-Marselha, França, desenvolveu o Projeto D-CIM [Spinosa 97] e em conjunto com a Universidade Laval, Canadá, está desenvolvendo o Projeto NETMAN [Cloutier 99], [Lyonnais 99] para integração de empresas suportada por SMAs.

- A Universidade de Metz, França, desenvolveu o Projeto SIROCO [Anciaux 97], [Roy 98] que será objeto de um maior detalhamento no Capítulo 4 desta tese.

Listam-se apenas as contribuições mais relevantes para os objetivos deste trabalho, especialmente aquelas que têm demonstrado uma maior persistência na literatura e que se apresentaram de forma mais consistente.

Capítulo 3

Uma Arquitetura Experimental baseada em Componentes

No capítulo anterior apresentou-se a problemática das Arquiteturas para Supervisão e Controle do Chão de Fábrica, sua inserção dentro das Arquiteturas de Referência para Sistemas de Manufatura e as disciplinas relacionadas às questões de operacionalização destas arquiteturas, as infra-estruturas de integração (IEI), o protocolo de comunicação industrial MMS e os Sistemas Multi-Agentes (SMA).

Neste capítulo será mostrado como os conceitos utilizados nas Arquiteturas de Referência podem ser aproveitados na geração de uma Arquitetura Genérica para Supervisão e Controle, desenvolvida a partir do conceito de Componentes Genéricos.

Como foi dito, o problema de supervisão e controle da produção pode ser atacado de diversas formas, podem ser utilizadas por exemplo soluções hierárquicas ou heterárquicas; reativas ou reflexivas ou híbridas; baseadas em modelos ou "*ad hoc*". Devido à essa gama de possibilidades julgou-se mais prudente envidar esforços no sentido de desenvolver componentes básicos que pudessem ser usados independentemente da estrutura funcional (hierárquica ou heterárquica) e do nível de inteligência / independência delegado à arquitetura.

Para articular estes componentes e formar uma Arquitetura Experimental a partir da qual podem ser derivadas diversas outras, mais específicas e adaptadas aos problemas singulares que sejam colocados ao projetista, foi desenvolvida também uma Metodologia de Projeto, calcada nas técnicas mais recentes de desenvolvimento de programas.

3.1 Noção de Componentes Genéricos

As Arquiteturas de Referência CIMOSA e GERAM explicitam em seus quadros de modelagem (Figura 2.9 Quadro de Modelagem CIMOSA) a necessidade de reutilizar resultados e modelos. O eixo de Instanciação de Blocos Funcionais ilustra como isto é feito: parte-se de construtores genéricos, adequados à cada uma das Vistas (Função, Informação, Recursos e Organização) e a partir deles pode-se construir modelos parciais (adequados à uma classe de empresas) e destes modelos parciais pode ser instanciado um modelo particular adequado à uma empresa específica.

Os Blocos Funcionais CIMOSA e GERAM existem para a construção de modelos. Como foi visto pretende-se que estes modelos possam vir a ser interpretados ou compilados, de maneira a permitir a execução do código equivalente, através de uma infra-estrutura de integração apropriada. A questão que colocada foi se seria possível desenvolver Componentes Genéricos que pudessem desempenhar um papel análogo aos Blocos Funcionais, mas dentro do contexto de infra-estrutura de integração, e com a redução do escopo do nível de Empresa para o nível de Chão de Fábrica.

Estes Componentes Genéricos **não buscam compatibilidade** com as Arquiteturas CIMOSA ou GERAM, apesar de daí colherem vastos subsídios para sua elaboração. Isto se deve ao fato de se buscar um enfoque pragmático, tentando propor elementos para facilitar a um projetista o desenvolvimento de uma arquitetura de supervisão e controle.

Neste enfoque buscou-se unir o projeto de baixo para cima (*bottom-up*), considerando os elementos reais existentes no chão de fábrica, com o projeto de cima para baixo (*top-down*) que considera uma vista mais ampla do mesmo setor. Como não há ainda uma validação e aceitação ampla, pela comunidade de usuários industriais e acadêmicos, de qualquer IEI (incluindo-se aí CIMOSA e GERAM), e considerando ainda, a emergência na última década, de importantes evoluções nos sistemas distribuídos e nos modelos de desenvolvimento de programas, surgiu a oportunidade de propor uma contribuição conceitual e experimental ao problema.

Os Componentes Genéricos estão, ao nível da IEI , dando suporte para a implementação de Arquiteturas para Supervisão e Controle do Chão de Fábrica. Eles poderão servir de "tijolos" que

permitirão construir a "Arquitetura Particular" que permita resolver um problema específico de uma empresa.

3. 2 Arquitetura Experimental

No capítulo anterior posicionou-se o Chão de Fábrica no contexto de gerenciamento e controle da produção, através da Figura 2.1 (Planejamento da Produção Tradicional). Verificou-se então que o Chão de Fábrica tem uma interface com os módulos superiores de planejamento da produção, os quais serão denominados daqui por diante simplesmente de Planejamento da Produção.

O Chão de Fábrica recebe do Planejamento da Produção os objetivos ou ordens de produção e deve, no momento adequado, executar estas ordens, mantendo o Planejamento da Produção atualizado sobre o estado atual da produção (quantidades, qualidade, problemas etc.). A parcela de inteligência que o Chão de Fábrica possui em relação ao sistema produtivo pode ser avaliada pelo grau de detalhamento das instruções (objetivos ou ordens de produção) recebidas do Planejamento da Produção. Assim, se forem recebidas ordens detalhadas com todos os passos e tempos, pode-se supor que este Chão de Fábrica é pouco inteligente (aqui equivalente a pouco autônomo); por outro lado se forem recebidos objetivos de produção contendo por exemplo datas limite e quantidades a produzir, sem no entanto haver a fixação dos tempos e dos roteiros, pode-se supor que o Chão de Fábrica é capaz de completar o trabalho de escalonar a produção, sendo portanto mais inteligente (autônomo).

Um dos propósitos gerais deste trabalho é permitir uma maior autonomia do Chão de Fábrica, considera-se que não é prudente tentar obter esta autonomia diretamente a partir da definição de uma arquitetura, mas sim a partir dos componentes associados por meio dela. Decidiu-se assim utilizar o conceito de Sistema Multi-Agentes (SMA) para a construção dos componentes e da arquitetura. Como visto no capítulo anterior o conjunto de vantagens apresentados pelos SMAs, quanto ao grau de autonomia permitido a um sistema específico, depende de decisões do projetista. Tais decisões não podem ser antecipadas, mas podem ser suportadas através de uma definição adequada dos Componentes Genéricos.

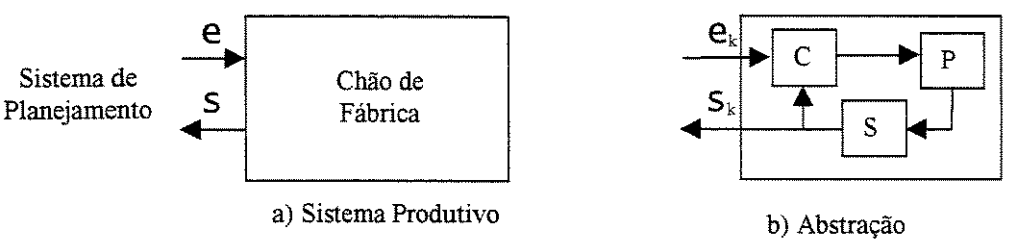
3.2.1 Derivação da Arquitetura Experimental

Vê-se na Figura 3.1 o relacionamento do Planejamento da Produção com o Chão de Fábrica (Sistema Produtivo).

e_k : um conjunto de entradas que podem possuir diversos níveis de refinamento, desde objetivos gerais de produção até ordens extremamente detalhadas

s_k : um conjunto de saídas contendo informações sobre o estado da produção e do Chão de Fábrica.

k: um índice seqüencial que permite relacionar as entradas e saídas.



a) Sistema Produtivo b) Abstração

Figura 3.1: Relacionamento do Planejamento da Produção e Chão de Fábrica

Na mesma Figura pode-se mostrar uma abstração (b) do que poderia ser o Sistema de Supervisão e Controle da Produção, contendo um módulo de controle (C), um módulo de produção (P) e um módulo para seguir (S) a produção, formando uma malha de controle.

Imagine-se agora que possa-se decompor o módulo de produção P em uma cadeia de transformações (O...O), sendo o elemento final da cadeia um módulo responsável por executar fisicamente a produção (M). Diversos níveis de realimentação sobre o estado da produção, via módulo S, podem existir, conforme representado pelas linhas tracejadas (para C e para os diversos Os) na Figura 3.2. Desmembra-se agora a parte física do Chão de Fábrica, este subsistema se comunicará com o outro através de outro conjunto de variáveis:

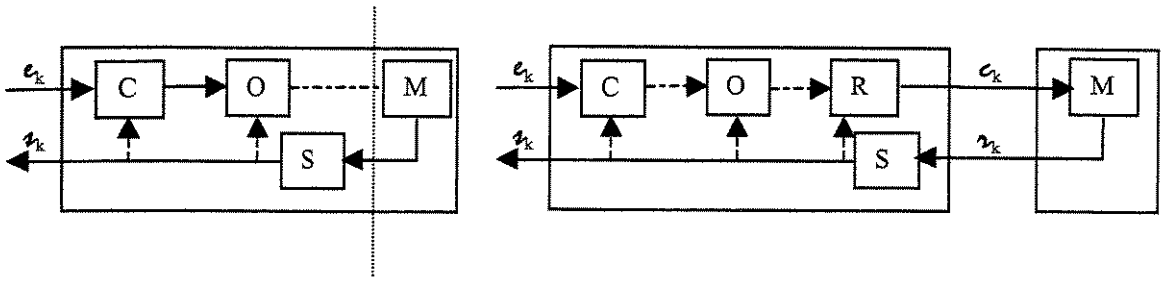


Figura 3.2: Partes Lógica e Física do Chão de Fábrica

c_{ki} : um conjunto de comandos para os recursos de produção (Homens, Máquinas e Aplicações).

r_{ki} : um conjunto de resultados em resposta aos comandos recebidos.

k: um índice seqüencial que permite relacionar as entradas e saídas.

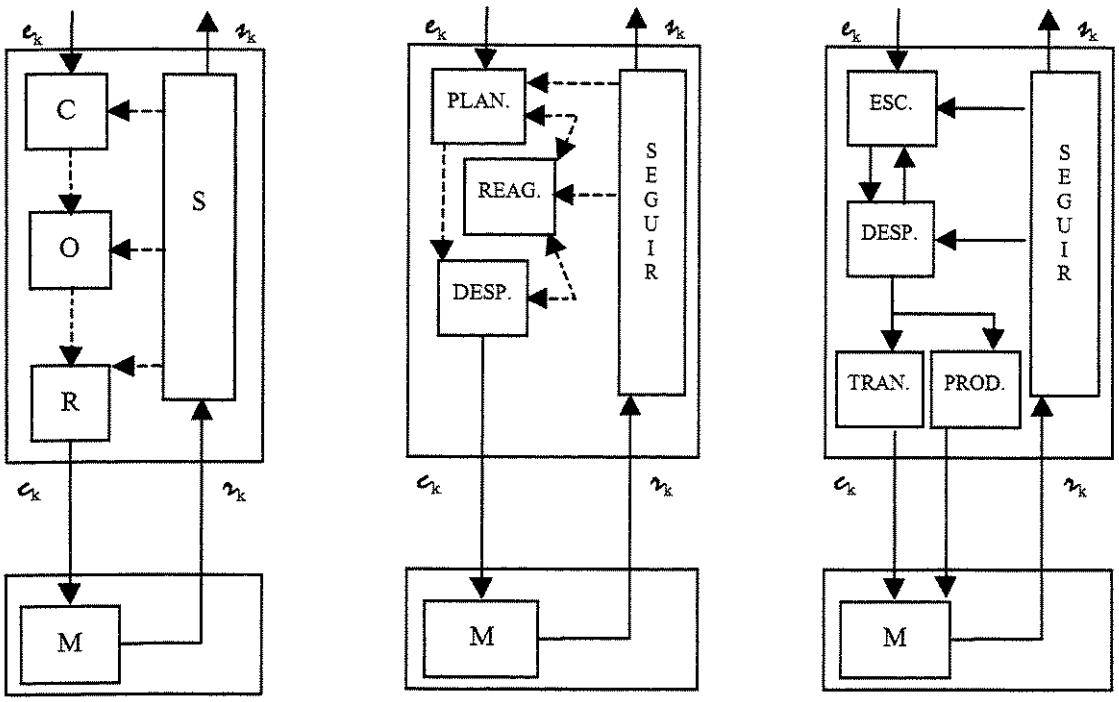
i: um índice seqüencial que permite ordenar os comandos e respostas referentes ao tratamento das variáveis referenciadas pelo índice k.

O módulo dentro da parte lógica que faz interface com a parte física é chamado de recursos (R).

Agora gira-se o conjunto no sentido horário de 90 graus e rearranjam-se os módulos para manter a seqüência entrada / saída, conforme letra a) da Figura 3.2.3.

Chamando a letra a) da Figura 3.3 de Arquitetura Lógica, pode-se verificar que, se fluxos bidirecionais forem permitidos, pode-se emular as arquiteturas de "Controle de Atividade de Produção"(PAC - *Production Activity Control*) representadas pelas letra b) adaptadas de [Huguet 95] e c) [Bauer 94].

Esta Arquitetura Lógica parece portanto adequada para ponto de partida na geração de uma Arquitetura Experimental.



a) b) c)

Figura 3.3: Arquitetura Lógica X PAC [Huguet 95] X PAC [Bauer 94]

3.2.2 Agentificação

Um novo paradigma de programação está surgindo, fundado no desenvolvimento dos SMAs, trata-se do "Projeto Orientado a Agentes" (POA). Há uma tendência a utilizar-se de técnicas do "Projeto Orientado a Objetos" (POO) mas ao mesmo tempo existem diferenças importantes entre agentes e objetos, como ilustrado por [Jennings 98], de tal forma a demandar a estruturação deste novo paradigma. Os primeiros esforços em POA remontam ao trabalho de Shoham ("*Agent-oriented Programming*") em 1993, desde então estão em cursos outros trabalhos que tentam explorar, definir e formalizar o que seria POA e aspectos relacionados [Brazier 97], [Jennings 98], [Wooldridge 99], [DeLoach 99] e [Gasser 00].

Existem diversos critérios para a criação de agentes de software, neste trabalho considerou-se que os agentes devem realizar uma parte do trabalho que lhes seja delegada, devem fazer sentido ao usuário (não parecerem construções artificiais) e devem ser capazes de cooperar através de comunicação.

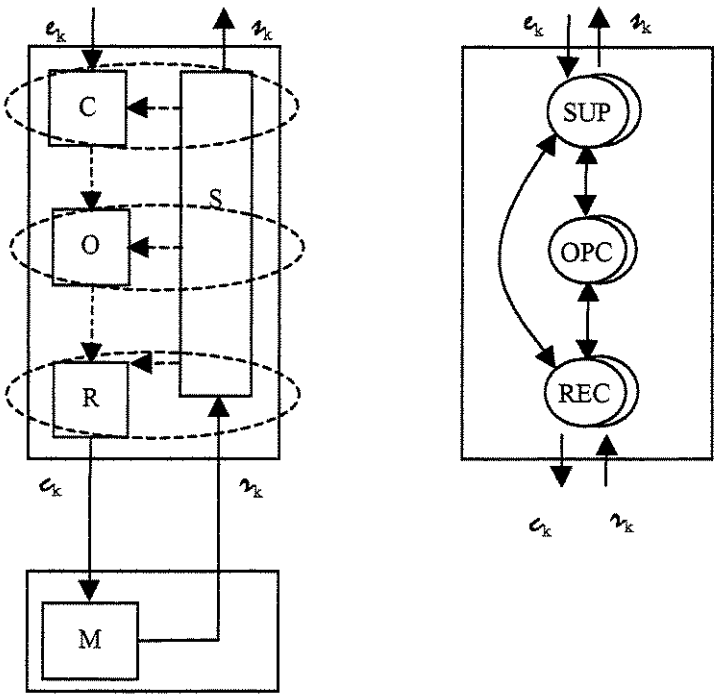


Figura 3.4: Agentificação e Arquitetura Experimental

Na Figura 3.4 mostra-se como gerar os agentes a partir da Arquitetura Lógica, através do agrupamento de funcionalidades, buscando criar elementos coesos e capazes de trabalhar autonomamente e de forma distribuída, gerando a Arquitetura Experimental (do lado direito da figura). Em especial enfatiza-se que os agentes são autônomos e capazes de diálogo, isso faz com que o fluxo de comunicação seja controlado por eles, justificando a incorporação da malha de realimentação dentro dos agentes.

Os agentes gozam de uma série de propriedades semelhantes aos objetos de *software*: eles podem ser especializados através da herança de propriedades de uma superclasse, eles podem possuir diversas instâncias (ou ocorrências) de um mesmo tipo de agente (por exemplo vários agentes recursos associados aos recursos físicos) e eles permitem um encapsulamento de funcionalidade e de informação, devido ao fato destas características somente serem acessíveis através de mensagens.

Fazendo a interface com o Planejamento da Produção, encontram-se os **Agentes Supervisor** (SUP), normalmente só será necessária uma instância desse agente, contudo é possível especializar e modificar a superclasse para realizar outras funções de interface, como

mostrar-se-á no exemplo do capítulo seguinte. Este agente recebe os objetivos ou ordens de produção e é responsável por detalhar os passos necessários (inclusive realizando escalonamento local se necessário) e pelo acompanhamento da evolução da execução, mantendo o Planejamento da Produção informado.

Fazendo a interface com os dispositivos físicos de produção (ou Homens ou Aplicações) estão os **Agentes Recurso**, normalmente pode-se usar um Agente Recurso para encapsular a funcionalidade de um dispositivo de produção (ou Homem ou Aplicação) existente, mas não é necessário definir-se isto como restrição.

Entre os Agentes Supervisor e Recurso(s) podem existir zero ou mais **Agentes Opcionais** (OPC), os quais podem ser nomeados de acordo com sua funcionalidade. Estes agentes podem estar organizados de qualquer maneira, não existe limitação original para sua topologia, podem formar uma hierarquia ou uma heterarquia, podendo ainda existirem tantos níveis ou camadas quanto o projetista deseje. Naturalmente é preferível optar-se pelas formas mais simples que sejam suficientes para os problemas em mãos.

A Arquitetura Experimental , baseada em agentes foi definida e mostrou-se como ela se relaciona com os elementos da empresa. Mostrar-se-ão a seguir os Componentes Genéricos que podem ser implementados na forma de agentes e permitirão a geração e implementação da Arquitetura Experimental e de suas especializações.

3.3 Componentes Básicos da Arquitetura

Na arquitetura vêem-se três tipos de agentes, eles partilham características comuns, das quais a mais evidente é a capacidade de comunicação, eles são já um nível de especialização de um agente básico, que é o Componente Genérico estruturado na forma de agente de *software*.

Viu-se no capítulo anterior uma classificação dos aspectos de SMAs, pode-se considerar que estes aspectos têm dois lados, o primeiro mostra o relacionamento entre os agentes (um lado social) e o segundo mostra o funcionamento interno do agente (um lado individual). Também encontra-se na literatura referências às visões Macro e Micro do agente.

Os aspectos que foram levantados foram: composição, configuração, comunicação, coordenação e modelagem. Todos tem seu lado social e individual mas a modelagem do agente é

UNICAMP
BIBLIOTECA CENTRAL

um aspecto que impacta os outros, amalgamando de certa forma os outros aspectos. Devido a esse fato há o interesse em inicialmente desenvolver um modelo do agente, este modelo não pretende ser universal ou seja, não pretende ser aplicável a outros sistemas que não os de supervisão e controle de chão de fábrica, as decisões de projeto serão feitas considerando as restrições destes últimos.

Após a apresentação do modelo do agentes analisar-se-ão os outros aspectos, iniciando pelo de comunicação, considerada fundamental para os SMAs.

3.3.1 Modelo do Agente

Neste momento, das questões colocadas por Genereseth e Ketchpel [Genereseth 94] e já citadas no capítulo anterior, têm relevância as duas primeiras, sobre a linguagem de comunicação e sobre a interpretação desta linguagem. Os agentes são autônomos e considera-se que as mensagens são equivalentes à eventos que irão ativar a reação dos agentes. Esta reação pode ser puramente automática, numa relação de causa e efeito, ou pode ser mais elaborada, baseada em conhecimentos presentes e passados e em reflexão sobre estes conhecimentos e sobre os objetivos do agente.

O agente deve conter então duas parte no mínimo, um sistema de comunicação e um sistema de interpretação e ação. A ação do agente será sempre decorrência de eventos, recebidos/ gerados através da mensagens ou gerados internamente (como saídas de temporizadores ou módulos de decisão).

A idéia é replicar no interior do agente o ambiente de comunicação existente no exterior, neste sentido, para facilitar a exposição do modelo, foi escolhido um modelo cliente-servidor para mostrar a interação na troca de mensagens / eventos. É importante observar que esta escolha não restringe a comunicação entre agentes ao paradigma Cliente-Servidor no sentido de relação mestre-escravo. Há uma tendência a utilizar-se a comunicação entre pares (*peer to peer*) nos SMAs, ela pode ser desmembrada logicamente em dois fluxos, em que cada par desempenha alternadamente o papel de Cliente ou Servidor. O protocolo MMS [MMS 90] é um exemplo deste desmembramento; dois robôs que se comunicam em MMS alternam os papéis de Cliente e

Servidor segundo o protocolo, por exemplo um pode pedir o estado do outro (serviço status), não há qualquer relação mestre-escravo entre eles devido ao protocolo.

A Figura 3.5 apresenta o modelo básico do agente:

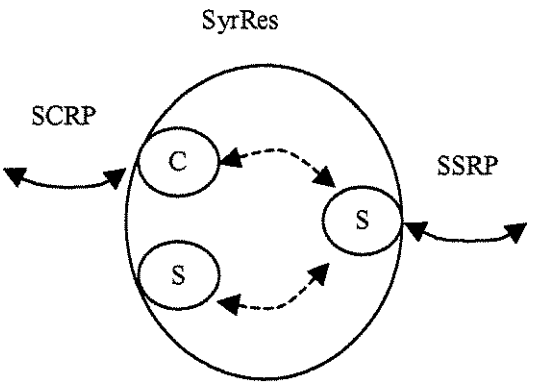


Figura 3.5: Modelo do Agente

Cada agente, identificado por seu nome (no caso SyrRes) pode ter zero, um ou mais Clientes (C) e um ou mais Servidores (S). Sempre existirá ao menos um Servidor, sempre apto a receber mensagens. Clientes e Servidores podem se comunicar internamente ao agente. A chegada de uma mensagem inicia uma transação (ou conversação), novas transações são conduzidas em paralelo através da criação de servidores de mesmo tipo.

As mensagens podem chegar de fora ou internamente ao agente. Os protocolos de comunicação dependem deste par Cliente / Servidor, uma parte do protocolo é realizada no lado Cliente e outra no lado Servidor; para fins de descrição do agente pode-se chamar de protocolos locais as partes que são realizadas pelo agente.

Assim na Figura 3.5 há dois protocolos locais, SCRP e SSRP, eles fazem par com outros dois protocolos locais definidos nos agentes com os quais se comunicam. Este protocolos locais podem ser descritos através de máquinas de estado, cada parte da transação (Cliente ou Servidor) terá usa própria máquina.

3.3.1.1 Sistema de Comunicação

A comunicação entre agentes pressupõe que eles possuam uma linguagem de comunicação em comum (não precisa ser comum a todos os agentes mas ao par que está em conversação). O desenvolvimento correto da comunicação depende não apenas dos mecanismos físicos de entrega das mensagens mas também dos mecanismos lógicos de interpretação das mesmas.

A interpretação das mensagens está associado à semântica (ou significação) das mesmas. Como já foi visto no capítulo anterior (Aspectos relevantes da arquitetura dos SMAs - Comunicação) ainda está por ser definida a questão de padronização da terminologia ou de ontologias para o gerenciamento da produção; devido a este fato deixa-se ao projetista do Sistema de Supervisão e Controle a especificação consistente da interpretação de sua linguagem de comunicação. Observa-se no entanto que para a comunicação com o chão de fábrica já existe o protocolo MMS [MMS 90] o qual serve como referência, num sentido amplo, para a definição das mensagens para este ambiente.

No âmbito dos SMAs, existem trabalhos no sentido de desenvolver uma Linguagem de Comunicação entre Agentes (ACL *Agent Communication Language*), estes trabalhos são na linha de utilização das linguagens KIF e KQML [Finim 93], existe uma tentativa de padronização destes esforços por meio da FIPA (*Foundation of Intelligent Physical Agents* http://www.fipa.org) .

Acredita-se que este enfoque (KIF + KQML), apesar de ter sido utilizado em alguns projetos na área de manufatura, por exemplo no PACT [Tenenbaum 92], não é ainda apropriado para os sistemas de supervisão e controle de chão de fábrica por :

- ser demasiado genérico, ainda restam por desenvolver as ontologias que permitam um diálogo padronizado relativo ao chão de fábrica e planejamento da produção.
- não considerar a interface com os padrões industriais existentes, MMS e Fieldbuses.
- não ser formalmente bem definido, segundo Lesperance *et al.* [Lesperance 96]

Neste sentido a proposta deste trabalho é desenvolver um sistema de mensagens "*ad hoc*", que considere os protocolos industriais de comunicação como referência geral e que ao mesmo

tempo deixe abertura para adequar-se aos novos protocolos que deverão ser desenvolvidos, talvez na linha (KIF + KQML).

O sistema de comunicação engloba duas partes, a primeira está relacionada com as regras para a comunicação, descrita através de um protocolo de comunicação, a segunda está relacionada com a estrutura sintática das mensagens, que permite que elas sejam transmitidas e reconhecidas por sistemas heterogêneos.

3.3.1.2 Modelo de Protocolo

Os agentes se comunicam através de mensagens, a chegada de uma mensagem em um agente que estava inativo dá início a uma nova transação (ou conversação), o decorrer desta transação é subordinado ao protocolo utilizado. O protocolo é dividido em duas partes, uma Cliente e a outra Servidor, cada uma delas é descrita por uma máquina de estados. Cada máquina de estados é descrita usando técnicas de desenvolvimento de sistemas de tempo real [Ward 85], [Hatley 88], considerando:

- Pré Condições
- Ações
- Pós Condições

As pré e pós condições são ativadas baseadas em eventos internos ou externos (por exemplo chegada de mensagens de um certo tipo). As ações contém o tratamento esperado para o evento recebido.

3.3.1.3 Sistema de mensagens

Existem duas visões necessárias para desenvolver o sistema de mensagens, a visão semântica e a visão sintática.

A visão semântica explicita o significado dos termos utilizados neste sistema de mensagens, assim esta visão está relacionada às ontologias para os diversos domínios do

conhecimento. Conforme explicado na introdução (item 3.3.1) o estado atual do desenvolvimento na área não permite aderir a padrões.

A visão sintática relaciona-se com a definição "gramatical" dos termos usados na comunicação e com o seu encadeamento dentro de uma mensagem. A Infra-estrutura de Integração CIMOSA e o protocolo MMS aderem aos formatos e métodos de descrição da ISO, julgou-se adequado este procedimento e ele será utilizado quando possível.

A estrutura sintática proposta para as mensagens é mostrada na Figura 3.6:

destino	M
fonte	M
idMensagem	M
serviço	M
ConteudoMens.	O

Figura 3.6: Estrutura das Mensagens

M significa que o campo é obrigatório (*mandatory*)
O significa que o campo é opcional

destino indica o objeto que recebe a mensagem.
fonte indica o objeto que envia a mensagem.
idMensagem identificador da mensagem
serviço indica a ação que deve ser executada no destino.
conteudoMens contém os parâmetros complementares.

O campo conteudoMens pode ser descrito usando o mesmo formalismo para mostrar quais são os seus componentes, se eles são obrigatórios, opcionais ou uma seleção entre parâmetros, e seus tipos de dados.

É feita ainda uma distinção entre os parâmetros do serviço para o caso de uma requisição ou de uma resposta. A requisição é feita por um Cliente ao Servidor, a resposta é gerada pelo Servidor com destino ao Cliente. Esta estruturação é a utilizada na norma MMS.

Não pretende-se neste instante propor um conjunto de mensagens, pois o mesmo seria especulativo e incompleto, no capítulo seguinte mostrar-se-á um exemplo de aplicação dos Componentes Genéricos e da Arquitetura Experimental e para este caso desenvolver-se-á um conjunto de mensagens seguindo a estruturação proposta. Observa-se porém que, para o caso da interface com os elementos físicos do chão de fábrica, um conjunto de mensagens derivado do MMS, com os serviços como: *Status*, *Identify*, *Read*, *Write*, *Start*, *Stop*, *Resume*, *Reset*, *Kill*, seria evidentemente adequado (vide 2.2.1.2 MMS - Especificação de Mensagens para a Manufatura).

3.3.1.4 Composição, Configuração e Coordenação

Quanto à Composição a Arquitetura Experimental proposta é um sistema heterogêneo; existem duas classes de agentes bem definidos, a classe Supervisor e a classe Recursos. a classe Opcional deve ser criada de acordo com o problema específico sendo resolvido, utilizando elementos do Componente Genérico e possivelmente das outras classes.

Quanto à configuração ela pode ser dinâmica, os agentes podem ser criados e terminados em tempo de execução.

Quanto à Coordenação dos agentes ela é feita através da troca de mensagens. Em um primeiro momento parece adequado o uso de protocolos imperativos que conterão ordens. Essa escolha permite tornar compatível a arquitetura proposta com as estruturas já existentes no chão de fábrica e com os protocolos utilizados, inclusive o MMS. Não há qualquer impedimento ao uso de protocolos mais sofisticados, que permitam por exemplo a negociação entre os agentes, sua implementação depende do grau de complexidade que se queira dar ao modelo do agente e terá impacto na dimensão e complexidade das máquinas de estado que ele utilizará.

No item Metodologia de construção de aplicações, vê-se que é possível descrever a coordenação entre os agentes, no nível mais elevado, através do uso de Diagramas de Estado (*Statecharts*). Estes diagramas podem explicitar protocolos de negociação mais gerais e permitem

a decomposição do problema através de uma hierarquia de diagramas, até chegar ao nível de máquinas de estado mais simples.

3.3.2 Metodologia de construção de aplicações.

Uma aplicação, para fins desta discussão, é a adaptação e extensão da Arquitetura Experimental para atender aos requisitos de um sistema de supervisão e controle da produção específico.

Existe uma diversidade de métodos para a construção de aplicações, do ponto de vista da Engenharia de *Software*; Pressman no Capítulo I de seu livro [Pressman 92] mostra seu conceito do que é *Software* e do que é Engenharia de *Software*, ele apresenta como podem ser feitos sistemas a partir do enfoque de ciclo de vida, desde o clássico, passando por prototipagem de sistemas, desenvolvimento em espiral e uso de técnicas de quarta geração. Argumenta que é possível utilizar-se uma combinação destas técnicas para o desenvolvimento de sistemas.

Para a classe de problemas enfrentados (supervisão e controle da produção), pertencente ao domínio dos sistemas de tempo real, é realmente útil combinar técnicas de projeto, especialmente visando o desenvolvimento de protótipos que possam validar os requisitos das aplicações. Há ciência de que o desenvolvimento de SMAs necessita de novas técnicas, possivelmente assemelhadas com o desenvolvimento de sistemas baseados em objetos. A metodologia foi baseada em obras clássicas do projeto de sistemas de tempo real [Ward 85], [Hatley 88] e ao mesmo tempo propõe a utilização da Linguagem UML (*Unified Modeling Language*) [Douglass 98] que fornece construtores adequados para este problema.

Em [Douglass 98] o ciclo de vida de desenvolvimento de aplicações UML contém as fases de: Especificação de Requisitos, Definição de Estrutura e Comportamento dos Objetos e Projeto do sistema. Em relação à Figura 3.7, a qual mostra o ciclo de vida tradicional [Pressman 92] (estendido com a fase de Projeto Detalhado) pode-se fazer o seguinte mapeamento (apenas conceitual): Especificação de Requisitos idêntica à de Douglass, Projeto Funcional engloba a definição estrutural e comportamental dos objetos, Projeto Detalhado idêntica ao projeto de sistema de Douglass.

Será utilizado este ciclo de vida clássico, considerando a possibilidade de desenvolver protótipos funcionais, os quais permitem validar os requisitos dos usuários.

Serão descritas as etapas de Especificação de Requisitos e Projeto Funcional, as demais etapas são mais dependentes do problema atacado e dos recursos tecnológicos disponíveis e não serão descritas nesse capítulo. No capítulo seguinte será apresentada uma aplicação onde as outra etapas serão exemplificadas.

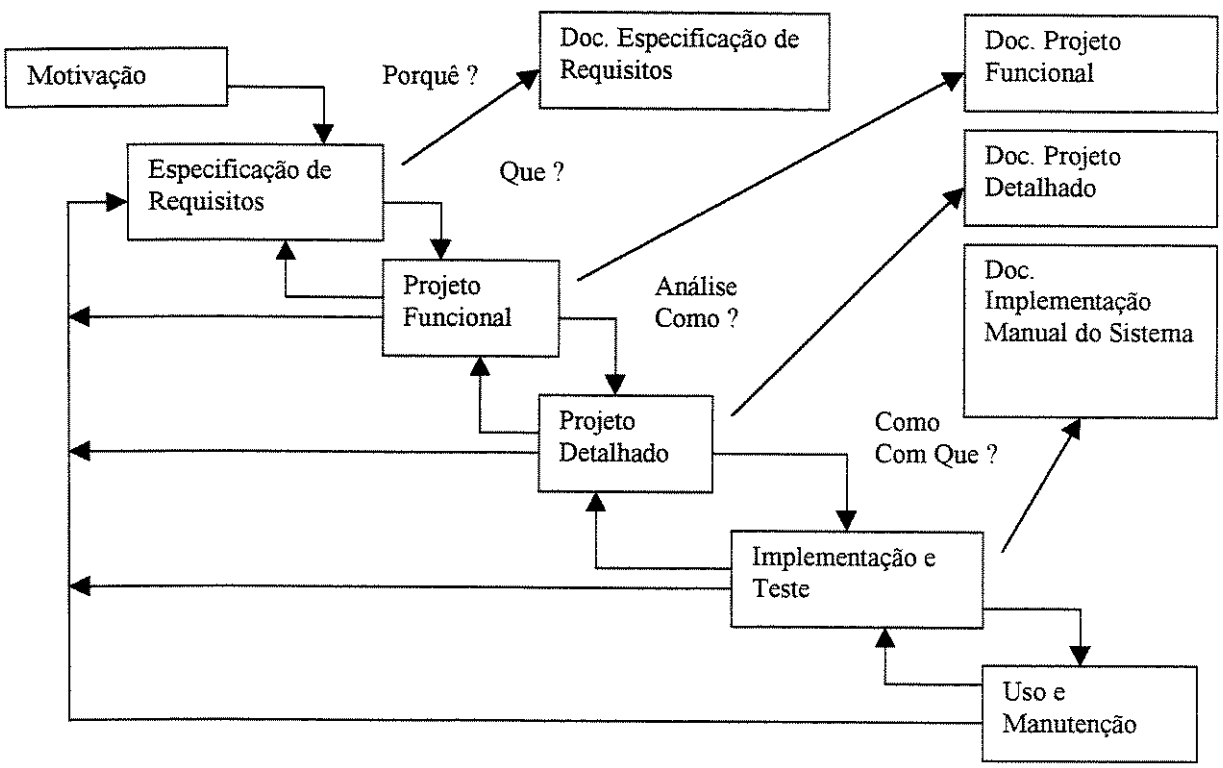


Figura 3.7: Ciclo de Vida de Projeto

3.3.2.1 Especificação de Requisitos

Nesta fase a questão básica é responder "o que" deve ser feito; a partir do problema analisado deve-se buscar definir as entradas e saídas do sistema. Em UML é proposto o uso de um diagrama de contexto em que são representados o sistema sendo modelado e as entidades externas com as quais ele se relaciona (através de fluxos físicos e lógicos). Esta fase tem uma

correspondência com a fase de mesmo nome utilizada no Projeto Estruturado de Sistemas de Tempo Real [Ward 85].

Como saída desta fase existirá uma Especificação de Requisitos, que conterá uma explicação geral do comportamento do sistema em relação às entidades externas e o conjunto de mensagens, eventos e fluxos físicos ou lógicos trocados com estas entidades.

Para documentar a composição dos dados (entradas e saídas) utilizar-se-á a estruturação proposta por Ward e Mellor [Ward 85] e Hatley e Phirbai [Hatley 88], a qual é mostrada na Figura 3.8:

Símbolo	Significado	Descrição
=	composto de	o elemento da esquerda é composto dos elementos da direita
+	em conjunto com	conecta os elementos de um grupo não ordenado
{ }	iterações de	a expressão contida em M{exp} N aparece um mínimo de M vezes e um máximo de N vezes. A ausência de M ou N indica o caso indefinido.
[|]	selecionar um de	seleção entre um ou mais elementos separados pela barra vertical
()	opcional	indica uma expressão opcional, eqüivale a {exp }1
" "	literal	os elementos devem ser considerados literalmente
* *	comentário	

Figura 3.8: Composição dos Dados

3.3.2.2 Projeto Funcional

Comporta as fases de definição estrutural dos objetos e seu comportamento de acordo com o proposto em [Douglass 98]. Aqui cabe observar que, como os SMAs não são Sistemas Orientados a Objetos, mesmo se possam ser construídos usando estes últimos, necessitam de uma maior ênfase na descrição da troca de mensagens entre seus componentes.

Neste sentido utiliza-se também uma descrição com Diagrama de Fluxo de Dados (DFD) , que permite uma explicitação mais realista da troca de mensagens e um refinamento em níveis se necessário.

Em UML são utilizados Diagramas de Uso (Use Cases) que mostram quais são as funções básicas executadas pelo sistema e como elas se relacionam. Os relacionamentos são do tipo herança, extensão, comunicação, etc. Eles permitem começar a estruturar as classes de objetos envolvidas no projeto.

A Figura 3.9 mostra um Diagrama de Uso, extraído de [Douglass 98].

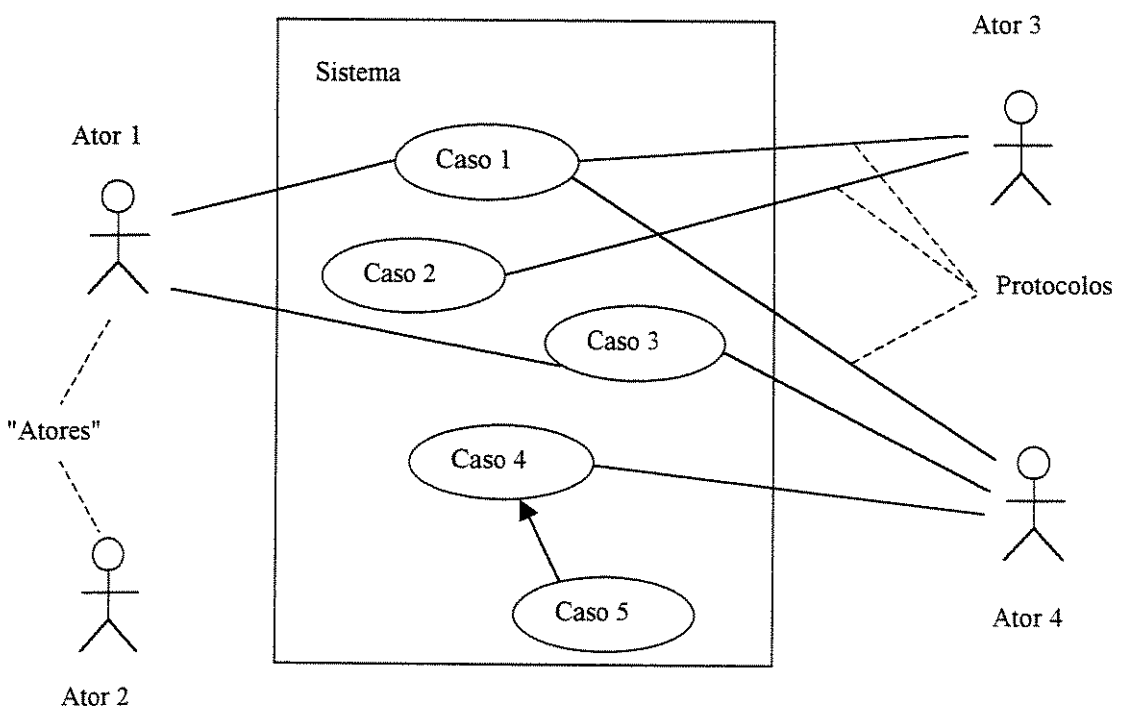


Figura 3.9: Diagrama de Uso **Fonte: [Douglass 98]**

A dinâmica (comportamento) do sistema é descrita usando-se Cenários e Diagramas de Seqüência (*Sequence Diagrams*). Cenários (instâncias específicas de Diagramas de Uso) são montados a partir de suposições sobre a evolução temporal do sistema em resposta aos eventos recebidos, eles são representados usando os diagramas de seqüência.

Neste ponto adicionam-se os DFDs, que têm uma utilidade adicional àquela já citada de explicitar as mensagens, eles permitem fazer uma engenharia reversa de sistemas já existentes de uma maneira mais amigável. A interação entre os processos (representados por círculos nos DFDs) é feita através de troca de mensagens. Uma determinada seqüência de troca é registrada em um diagrama de seqüência. Se os círculos dos DFDs forem atribuídos a agentes fica mais fácil acompanhar os diagramas de seqüência.

Na Figura 3.10 mostra-se um Diagrama de Seqüência fictício (lado direito) gerado supondo o sistema composto por agentes Supervisor, Opcional e Recurso mostrado do lado esquerdo. O agente Supervisor pede para o agente Opcional iniciar a produção, este envia pedidos ao agente Recurso para executar as fase de produção 1 e 2, recebendo uma informação de final de fase para cada uma delas, a seguir informa ao Supervisor que a produção está terminada. Diversos outros diagramas podem ser gerados supondo outros cenários, por exemplo falha no Recurso ou falha de comunicação.

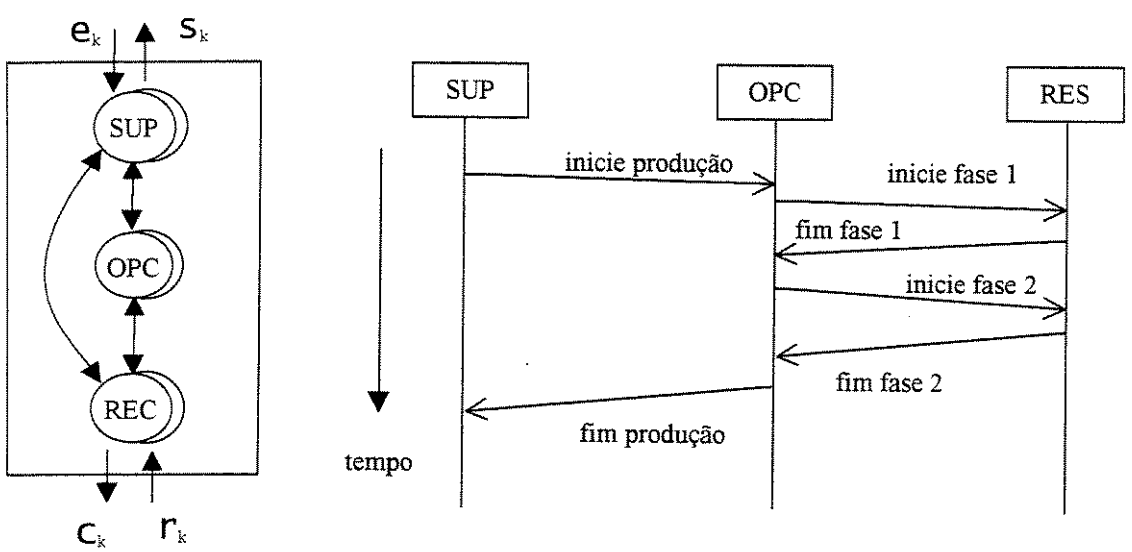


Figura 3.10: Diagrama de Seqüência

Os diagramas de seqüência podem ficar muito complicados quando há uma transação que envolve múltiplas entidades, um diagrama tipo DFD usado em conjunto permite lidar melhor com esta complexidade.

Estruturação de classes - quanto à descrição da estrutura dos objetos, UML utiliza as técnicas de Projeto Orientado a Objetos. São definidas cinco classes de relacionamento entre objetos: associação, agregação, composição, generalização e refinamento. Não entrar-se-á em mais detalhes sobre a representação gráfica e o detalhamento das classes, pois esses são compatíveis com as técnicas tradicionais de Projeto Orientado a Objetos, já que o grupo que desenvolve UML contém os nomes de Grady Booch, Ivar Jacobson e Jim Rumbaugh, que são as referências na área.

Comportamento - quanto ao comportamento dos objetos, UML prevê a utilização das "*Statecharts*" desenvolvidas por Harel [Harel 87], que também faz parte do grupo que desenvolve as especificações UML. Descrever um sistema complexo através de máquinas de estado simples pode ser uma tarefa dificílima, devido ao número de estados envolvidos. A idéia por trás das "*Statecharts*" é permitir uma estruturação hierárquica do sistema, com respeito `a evolução dinâmica de seus estados.

Problemas da especificação das "*Statecharts*", tais como uma certa indefinição semântica dos gráficos e símbolos ("*Statecharts*" é uma ferramenta visual), foram sendo resolvidos, pelo próprio Harel e por outros, que propuseram extensões à sua obra. Por exemplo pode-se citar Suraj *et al.* [Suraj 97] que propõem uma extensão para a modelagem e especificação dos sistemas de controle da manufatura. Em seu trabalho é de especial interesse a metodologia de prototipagem de um modelo, que praticamente coincide com esta proposta (quanto ao aspecto da dinâmica do sistema).

Em um certo nível de decomposição dos diagramas de estado, chega-se a diagramas simples, os quais podem ser representados e implementados pelas técnicas convencionais tais como aquelas indicadas em [Ward 85] e [Hatley 88].

3.4 Robustez e flexibilidade da arquitetura

Pode-se dizer que a robustez da arquitetura é obtida por projeto, já que os agentes são concebidos para trabalharem de maneira autônoma, isto é uma vez recebidos seus objetivos eles buscam realizá-los de maneira independente. Naturalmente o ponto crítico da arquitetura é a comunicação, os agentes podem necessitar trocar informações durante a consecução de seus objetivos e se o elo de comunicação estiver interrompido eles poderiam ficar bloqueados.

Para evitar o bloqueio dos agentes foi previsto um sistema de temporização (*Timeout*), que permite que os agentes passem para um modo de tratamento de falha. Não é possível antecipar os tipos de falha, já que elas dependem da aplicação, mas foi previsto um esquema conservador em que, após o disparo da temporização de comunicação (espera de resposta), o agente tenta recuperar a conexão por um certo número de vezes, temporizando cada tentativa. Se após estas tentativas o agente não conseguir continuar a transação (por falta de comunicação) ele faz uma "limpeza de casa" (termina transações pendentes) e passa para um estado inicial (*default*).

Se a falha de comunicação for intermitente, como é normalmente o caso de redes muito carregadas, o agente pode continuar a trabalhar após recuperar a comunicação. Se suas atividades não dependerem de comunicação (atividade de computação por exemplo) os agentes operam autonomamente, ativando a comunicação apenas no momento de responder ao serviço requisitado.

Quanto à flexibilidade da arquitetura, observa-se que a mesma foi mantida muito simples, com poucos componentes básicos. Esta simplicidade permite especializá-la para realizar diversas funcionalidades, permitindo grande flexibilidade em sua configuração.

O elemento que permite esta flexibilidade é o agente básico, o Componente Genérico que pode receber requisições de serviço a qualquer instante, alocando o tratamento deste serviço a um Servidor instanciado dinamicamente. Este Componente Genérico também permite o estabelecimento de conexões com um número ilimitado de outros agentes (através de seus Clientes), independente de qualquer hierarquia. A definição da configuração é responsabilidade do projetista, que deverá também prover as definições dos protocolos a serem utilizados. Quanto maior o número de agentes diferentes e de instâncias desses agentes tanto maior será a complexidade do sistema.

Para uma aplicação simples, a Arquitetura Experimental já é suficiente, por exemplo para o controle de uma célula de fabricação baseada em ordens já totalmente escalonadas, neste caso os elementos já implementados poderão ser facilmente reutilizados.

3.5 Conclusão

Apresentou-se neste capítulo uma Arquitetura Experimental para supervisão e controle de chão de fábrica. Esta arquitetura baseia-se em um Componente Genérico, capaz de aceitar a requisição de serviços e de executá-los de maneira autônoma. A inspiração para este Componente Genérico veio das Arquiteturas de Referência, em especial CIMOSA e GERAM; para manter a simplicidade do componente, visando sua aplicação imediata a problemas industriais, não foi buscada a compatibilidade direta com estas arquiteturas. A outra fonte de inspiração para este Componente Genérico, a qual aliás também inspirou a Infra-estrutura de Integração CIMOSA, é o protocolo MMS.

Este Componente Genérico pode ser refinado, inicialmente para fornecer os Agentes Supervisor, Recursos e Opcional da Arquitetura Experimental, e posteriormente para especializar estes agentes ou mesmo criar outros que se façam necessários.

A Arquitetura Experimental é construída a base de Agentes, sendo o Componente Genérico o Agente Básico que é utilizado para construir os outros. Este Agente Básico foi desenvolvido considerando aspectos de tempo real, em especial suporta a perda momentânea da comunicação e pode recuperar-se, permitindo manter a transação (conversação) em andamento.

O que distingue esta Arquitetura Experimental de outros trabalhos que foram desenvolvidos, classificados tanto como Sistemas Multi-Agentes quanto como Arquiteturas de Supervisão e Controle e já referenciados no Capítulo 2, é que esta Arquitetura Experimental:

- desde sua concepção busca uma compatibilidade com os problemas imediatos da indústria, através de sua simplicidade e capacidade de ação em tempo real
- considera a questão primordial da interface com os Sistemas de Planejamento, no nível superior e com o Dispositivos Físicos no nível inferior

- através de Componentes Genéricos simples, que podem ser validados e aperfeiçoados, busca permitir a reutilização de blocos funcionais, otimizando o projeto de novas arquiteturas particulares, sem que essas tenham de partir da estaca zero

- é robusta, flexível e adaptada ao problema de supervisão e controle de chão de fábrica

No capítulo seguinte será apresentado um exemplo de derivação de uma Arquitetura Particular, baseada na Arquitetura Experimental e utilizando a Metodologia de Desenvolvimento de Aplicações preconizada aqui. Serão mostrados os detalhes da metodologia, por exemplo explicitando as mensagens trocadas entre os agentes e sua formalização.

Capítulo 4

Aplicação da Arquitetura Experimental

Após a definição, no capítulo anterior, de uma Arquitetura Experimental baseada em Componentes Genéricos, será mostrado como pode-se derivar outras arquiteturas (particulares), adaptadas aos problemas encontrados em cada empresa.

A seqüência que será utilizada é baseada na metodologia de projeto apresentada no capítulo anterior. Utilizar-se-á como alvo para este protótipo a plataforma PIPEFA [Rosário 96], composta por uma célula de montagem e desmontagem e equipamentos e aplicativos associados (vide Figura 4.6).

A PIPEFA permite a montagem ou desmontagem de uma variedade de 26 produtos, que podem ser produzidos em forma unitária ou em pequenos lotes. A alimentação de matéria prima nos alimentadores e a retirada dos produtos acabados (ou rejeitados) dos depósitos é feita manualmente, porém todo o restante da produção e verificação de qualidade do produto é feita automaticamente sob controle computadorizado.

Esta plataforma é representativa dos problemas reais encontrados em Sistemas Flexíveis de Produção, pois contém dispositivos reais de sensoriamento e atuação, bem como aplicativos de supervisão e controle da produção disponíveis comercialmente. Em vista disso acredita-se que a derivação de uma Arquitetura de Supervisão e Controle para a PIPEFA será de grande valia para dimensionar a qualidade do trabalho desenvolvido com os Componentes Genéricos e a Arquitetura Experimental.

Inicialmente serão apresentadas duas arquiteturas referenciadas na literatura e que são aplicáveis à PIPEFA. Discorrer-se-á rapidamente sobre a questão de algoritmos de escalonamento adequados a essa aplicação e far-se-ão algumas observações sobre como considerar as restrições colocadas pelo mundo real de sistemas de produção.

Partir-se-á então para projetar uma arquitetura adequada à PIPEFA, seguindo a metodologia especificada e desenvolvendo os Componentes Genéricos e Arquitetura Experimental.

Mostrar-se-á o estado atual da implementação e far-se-á uma avaliação dos resultados disponíveis.

4.1 Seleção de uma Arquitetura

Dentre os exemplos descritos na literatura de Sistemas de Supervisão e Controle baseados em SMAs, foram selecionados dois, considerando os critérios:

- aplicabilidade a um Sistema Flexível de Produção de pequeno porte (Célula) **real**.
- disponibilidade de documentação adicional além dos artigos de referência.

Os sistemas selecionados foram: PHOCS / PARSIFAL [Bongaerts 98] e SYROCO [Anciaux 97], [Roy 98].

4.1.1 PHOCS / PARSIFAL

O grupo PMA (*Production Engineering, Machine Design and Automation*), do Departamento de Engenharia Mecânica da Universidade Católica de Leuven, Bélgica, têm um importante papel na pesquisa de Sistemas Holônicos e desenvolveu uma Arquitetura de Referência para Sistemas Holônicos de Manufatura, chamada PROSA [Brussel 98].

PROSA define um conjunto de três hólons básicos: recurso, produto e ordem. Opcionalmente um quarto hólon, chamado "*Staff*" pode ser utilizado na arquitetura.

O hólon Recurso relaciona-se com os recursos de produção do sistema de manufatura. Ele oferece a capacidade e funcionalidade destes recursos aos demais hólons.

O hólon Produto contém o modelo de produto, relacionando-se com os aspectos de projeto desse produto, tais como: planos de processo, composição e restrições de qualidade; ele não contém informações sobre o estado atual de produção de um produto sendo fabricado.

O hólon Ordem (Pedido) representa uma tarefa do sistema de produção. Ele contém informações e gerencia a produção do produto físico sendo fabricado. A funcionalidade do hólon Ordem pode ser comparada com os módulos de monitoramento e despacho de sistema de controle tradicionais.

O hólon *Staff* tem o propósito de assistir aos outros hólons na realização de seus trabalhos. Eles podem ser considerados como "conselheiros" para os hólons básicos, que são os que realizam as tarefas de produção. Os hólons *Staff* permitem a presença da funcionalidade de elementos centralizados na arquitetura, a qual sem eles seria puramente heterárquica.

A partir da Arquitetura PROSA, Luc Bongaerts [Bongaerts 98] desenvolveu sua arquitetura de controle de chão de fábrica. A idéia trabalhada em sua tese é a de que um Escalonador pode operar de maneira concorrente com o controle do chão de fábrica, sendo que este controle realizaria especificamente as funções do Despachador de acordo com o descrito em [Bauer 94].

Assim ao invés de reagir às perturbações através de um reescalonamento total da produção o Escalonador Concorrente poderia, considerando o estado atual do sistema, "aconselhar" sobre uma nova seqüência de operações. Este Escalonador necessitaria de novos algoritmos e alguns são propostos na tese.

Na Figura 4.1 apresenta-se a Arquitetura PHOCS /PARSIFAL.

PARSIFAL (*PMA Reactive Scheduler for Flexible Assembly Systems, Leuven*) é o Hólon de Escalonamento Reativo cujo propósito é escalonar de maneira dinâmica, repetitiva e reativa a célula de produção do Laboratório PMA. Ele foi projetado de maneira a facilitar o experimento de diferentes algoritmos de escalonamento.

As Células podem atuar em diversos Produtos coordenando sua produção. Os agentes Produto são uma espécie de memória da célula, contendo os dados do produto tais como roteiros e tempos de fabricação.

O Supervisor controla o conjunto de Células e responde por uma reorganização da produção em função de eventos aleatórios tais como quebra ou indisponibilidade de recursos ou mudanças nos objetivos de produção.

O Meta-Objeto gera dinamicamente os agentes Produtos, sob demanda do Supervisor, para atender os objetivos de produção, mesmo em caso de ocorrência dos eventos citados. Ele faz uso de ferramentas e de conhecimento externos para tal. Por exemplo uma das ferramentas é um programa de geração de caminhos para escalonar localmente a produção (módulo Escalonamento) e os conhecimentos podem estar contidos em base de dados de processos de produção (BDOO Base de dados orientada a objetos).

SYROCO foi testado com um exemplo acadêmico [Roy 98], que não utiliza recursos reais, e mostrou-se satisfatório, sendo capaz de reescalonar localmente a produção quando da ocorrência de eventos aleatórios.

Sua implementação foi feita com as linguagem C e C++, com a comunicação baseada em TCP/IP (*Transport Control Protocol / Internet Protocol*). O algoritmo de escalonamento utilizado é do tipo *Simulated Annealing*.

4.1.3 Arquitetura selecionada

Os dados obtidos das duas arquiteturas que foram estudadas em mais detalhe não permitem decidir da superioridade de uma sobre a outra, devido ao fato de que não se dispúnha do mesmo conjunto de dados de controle, mesmo conjunto de algoritmos de escalonamento e da mesma estrutura física de produção para compará-las.

Ambas as arquiteturas são baseadas no conceito de SMAs, com as vantagens associadas a este enfoque.

Elas também usam o conceito de agente Recurso para encapsular as funcionalidades do sistema físico de produção.

As duas usam uma forma de representação do conhecimento sobre o produto através de um agente Produto.

Um módulo externo de escalonamento é usado por SYROCO, sendo que a execução e controle do escalonamento é realizada pelo agente Supervisor, enquanto PHOCS / PARSIFAL usa um agente de escalonamento (PARSIFAL) e outro de controle (PHOCS).

SYROCO utiliza um agente Meta Objeto para gerar dinamicamente outros agentes, neste sentido tem uma complexidade estrutural maior do que PHOCS / PARSIFAL.

Decidiu-se pela utilização de uma estrutura baseada na arquitetura SYROCO, esta escolha deveu-se a:

- ela prevê o uso de ferramentas externas, sendo projetada de maneira modular e expansível (através destas ferramentas)
- desenvolvimento e experimentação de algoritmos pode ser feito sem modificar a arquitetura (eles são fornecidos pelas ferramentas externas).
- a estruturação de SYROCO lembra o modelo NBS, facilitando sua compreensão pelo pessoal do chão de fábrica
- a cooperação acadêmica existente entre a Universidade de Campinas e a Universidade de Metz facilitaria o uso da arquitetura.

Contudo, devido à maior complexidade de SYROCO, algumas modificações estruturais foram feitas, visando adaptá-la ao problema de controlar a célula de produção da PIPEFA.

A experiência da FEM-Unicamp em conjunto com o ITI, com sistemas reais e com a PIPEFA, aponta para a necessidade de arquiteturas enxutas, facilmente explicáveis ao usuário e compatíveis com os recursos reais existentes. Neste sentido foi proposta uma modificação que aproxima SYROCO das hierarquias de sistemas de controle de chão de fábrica existentes, sem no entanto desvirtuar sua filosofia de distribuição e reatividade.

Esta arquitetura conta com os agentes: Meta-Objeto, Supervisor, Célula e Recursos.

A modificação consiste em incorporar o agente Produto dentro da Célula, isto faz sentido pois células (físicas) são organizadas de acordo com as famílias de produtos que elas podem gerar. Tal modelo remete de imediato ao bem conhecido modelo NBS [McLean 83], facilitando o diálogo com o meio industrial.

Do ponto de vista de desempenho, a eliminação de um nível de distribuição e os caminhos de comunicação associados, deve levar a uma melhoria global do sistema, quando se consideram sistemas de produção organizáveis em células, como aqueles representáveis na PIPEFA.

A necessária reatividade do sistema foi então preservada, através da possibilidade de reescalonar localmente a produção.

4.1.4 Derivação da Arquitetura Particular

A partir da Arquitetura Experimental pode-se derivar a Arquitetura SYROCO-2 (O 2 indica uma modificação da arquitetura original) da seguinte forma:

- SYROCO-2 possui dois agentes que fazem interface com o Planejamento da Produção, o Supervisor e o Meta Objeto., eles podem ser derivados do tipo Supervisor da Arquitetura Experimental.

- agente Célula de SYROCO-2 pode ser derivado do agente Opcional da Arquitetura Experimental.

- agente Recurso de SYROCO-2 mapeia-se no agente Recurso da Arquitetura Experimental

A Figura 4.3 apresenta esquematicamente a derivação de SYROCO-2. No lado esquerdo está a Arquitetura Experimental e no lado direito a Arquitetura derivada SYROCO-2.

SYROCO-2 herdará as capacidades de comunicação e interfaceamento existentes na Arquitetura Experimental, os fluxos de mensagens entre os seus agentes componentes deverão ser

redefinidos a partir dos existentes na Arquitetura Experimental, de acordo com a metodologia proposta no capítulo anterior e que será ilustrada neste capítulo.

A cardinalidade dos agentes é: Um Supervisor, Um Meta Objeto, várias Células e vários Recursos. Na prática será adotada a simplificação de permitir apenas uma Célula ativa.

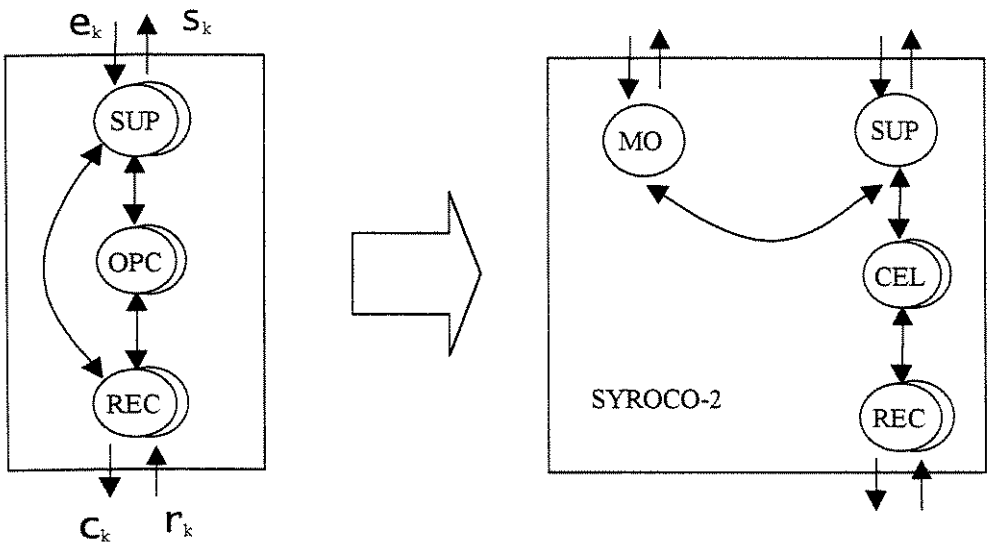


Figura 4.3: Derivação de SYROCO-2

4.2 Seleção de Algoritmo de Escalonamento

"Escalonamento é a alocação de recursos no tempo para realizar uma coleção de tarefas" [Baker 74]. Antes do escalonamento ser realizado, é preciso que a resposta para três questões de **planejamento** estejam respondidas:

- que produtos ou serviços deverão ser providos?
- em que escala eles serão providos?
- quais recursos estarão disponíveis?

A partir destas respostas o problema de escalonamento procura determinar as questões:

- qual recurso será alocado para realizar cada tarefa?
- quando cada tarefa será realizada?

Mostrou-se no Capítulo 2, Figura 2.1 "Planejamento da Produção Tradicional", como está organizada a relação entre as fases de planejamento execução da produção. Tradicionalmente pressupõe-se que as questões de planejamento citadas estejam resolvidas, um escalonador, ainda no nível de planejamento, determina então a alocação física e temporal das tarefas, cabendo ao chão de fábrica a execução deste escalonamento.

Neste esquema tradicional a obtenção de um escalonamento pode levar horas ou dias [Newman 88], o que o torna incompatível com as necessidades atuais de sistemas de produção diversificada em pequenos lotes (mesmo que eventualmente ainda possa ser usado em outras aplicações).

A tendência corrente é de "fortalecer" o chão de fábrica, transferindo-lhe parte das atribuições de escalonamento, as arquiteturas PHOCS/ PARSIFAL e SYROCO são representativas desta tendência.

As novas arquiteturas que estão sendo desenvolvidas, para melhorar a resposta do chão de fábrica às perturbações, necessitarão também de novos algoritmos de escalonamento. Na atualidade, nos poucos casos em que há um escalonamento de curto prazo, são utilizadas heurísticas. Para fazer um escalonamento dinâmico um "despachador" utiliza desde regras simples de prioridades, até heurísticas mais complexas e regras de decisão [Newman 88].

Lin e Solberg [Lin 91] argumentam que é possível melhorar o desempenho de um sistema de produção se for adicionada flexibilidade nos planos de processo (significando que a seqüência de operações não é fixa). Para permitir uma resposta rápida são usadas "regras de despacho", um total de nove foram testadas. Este tipo de enfoque não usual está sendo possível devido à evolução das arquiteturas e dos recursos computacionais.

Um extenso levantamento de algoritmos de escalonamento da produção, passíveis de serem aplicados à sistemas heterárquicos, foi realizado em [Baker 98]. Sem dúvida serão necessários adaptar os algoritmos existentes e inventar novos, como afirma Baker. Acredita-se que é possível inovar nas arquiteturas, trazendo poder de decisão ao chão de fábrica, através do uso de algoritmos simples e eficazes, já amplamente testados na prática.

As estruturas de decisão e controle estão deixando de ser puramente hierárquicas, passando a ser híbridas ou heterárquicas. Rolstadas [Rolstadas 94] afirma que "para realizar o sistema de planejamento e controle de produção da próxima geração, técnicas de modelagem serão importantes", este trabalho é um primeiro passo neste sentido.

Para cumprir o objetivo, os algoritmos de escalonamento são meios e não fins, neste sentido optou-se pela arquitetura SYROCO, na qual a funcionalidade do escalonamento está disponível em um módulo externo, permitindo então testar diversas possibilidades sem modificar a arquitetura.

Lin e Solberg [Lin 91] usaram regras de despacho, Bongaerts [Bongaerts 98] usou um algoritmo para minimizar o atraso, Roy [Roy 98] usou um algoritmo baseado no "*simulated annealing*". O módulo de escalonamento aparece então como mais um elemento de experimentação.

Neste sentido a primeira versão deste módulo deverá utilizar um algoritmo de "Busca Local" desenvolvido em [Claudio 00], baseado em algoritmos divulgados na literatura [Pirlot 96], [Fadil 97]. O algoritmo desenvolvido possui ordem quadrática e seu mecanismo de busca local procura evitar ótimos locais, através da análise de um maior número de vizinhanças.

4.3 Projeto e Implementação

Usualmente as fases de Projeto e Implementação são executadas separadamente, no entanto neste caso técnicas de prototipagem foram usadas, de forma que estas fases foram combinadas. As especificações foram transferidas para um Sistema Multi-agentes, desenvolvido usando a linguagem Java [Flanagan 97].

A intenção foi de procurar características genéricas que poderiam permitir a reutilização dos resultados do projeto em aplicações subsequentes. Este é um primeiro esforço com o propósito de desenvolver blocos genéricos que facilitem a integração da manufatura.

Descrever-se-á a Arquitetura Experimental e como ela pode ser configurada para gerar a Arquitetura Particular SYROCO-2.

4.3.1 Decisões Tecnológicas.

Das fases de Especificação e Projeto pode-se derivar alguns atributos os quais se mapeiam em tecnologias existentes,

- Sistema Multi-agente
- Usa a linguagem Java.
- Interfaceia com outras linguagens.
- Orientado à troca de mensagens.
- Cliente - Servidor
- Especialização através de módulos externos
- Reutiliza o conhecimento existente

Para contemplar estes requisitos foi proposto um sistema multi-agentes como solução de implementação. Juntamente com a linguagem Java ele permite satisfazer a esse conjunto de especificações.

Exemplos de sistemas aparentados, usando a linguagem Java para suporte de SMAs, são: JAFMAS [Chauahan 97], JATLITE [Jatlite 97] e UMASS [Horling 98].

4.3.1.1 Linguagem de programação

Java foi a linguagem escolhida, devido à suas características de legibilidade, portabilidade e ligação em rede. Ela tem também pontos negativos, como por exemplo uma performance inferior quando comparada às linguagens compiladas. Mesmo assim pensa-se que o balanço geral de Java é positivo o que justifica sua seleção.

Deve ser observado que Java não é projetada para aplicações de tempo real, mas há uma tendência de usá-la em projetos de controle de chão de fábrica. Considerou-se que os requisitos específicos de tempo real desta aplicação podem ser satisfeitos por programas dedicados de Supervisão e Controle e pelos Controladores Programáveis já existentes no ambiente PIPEFA.

4.3.1.2 Sistema Operacional

Windows foi escolhido como sistema operacional. Espera-se que as aplicações em

desenvolvimento rodem também em ambiente Unix devido ao código em Java.

4.3.2 Definição de Requisitos

Nesta fase a reatividade foi enfatizada. O sistema deve ser capas de responder a eventos imprevistos, vindos tanto do nível mais alto (planejamento da produção) quanto do nível inferior, na forma de notificações de quebra de máquinas.

Os seguintes requisitos gerais foram levantados:

- Orientado à pequena e média empresas
- Capaz de escalonar localmente a produção
- Totalmente distribuído
- Configurável e facilmente atualizável
- Portável e multi-plataforma
- Interface com Sistemas e Aplicações já existentes.
- Tolerante à falhas

4.3.3 Projeto Funcional

No Projeto funcional foram desenvolvidas as características gerais dos agentes, sendo estes: Meta-Objeto, Supervisor, Célula e Recursos. Tais agentes interagem com entidades externas (não pertencentes à arquitetura) do tipo Ferramentas (de reescalonamento da produção ou reestruturação do chão de fábrica), Base de Dados, Planejamento da Produção (Global) e Recursos Reais.

Buscou-se desenvolver uma especificação funcional que pudesse ser mais genérica, isto é capaz de ser reutilizada em problemas semelhantes.

Foi utilizada a metodologia proposta pela linguagem de modelagem UML (*Unified Modelling Language*) aplicada a problemas de tempo real [Douglass 98]. Tal metodologia prevê uma combinação entre diferentes técnicas de descrição, dentre as quais foram especialmente utilizadas as *State Charts* [Harel 87] e os diagramas de seqüência.

No entanto a falta de uma ferramenta automática baseada em UML não permitiu uma verificação extensiva do projeto.

A Figura 4.4 apresenta o diagrama geral do sistema, usando os Diagramas de Fluxo de Dados, de acordo com a metodologia de Hatley-Pirbhai [Hatley 88] e Ward-Mellor [Ward 85], que se julga facilitar o entendimento do sistema.

Os agentes são o Meta-Objeto, Supervisor, Célula e Recurso. Eles interagem com entidade externas (retângulos), as quais não pertencem à arquitetura, como Ferramentas (escalonamento), Banco de Dados de Produção (BDP), Planejamento da Produção e Recursos Físicos.

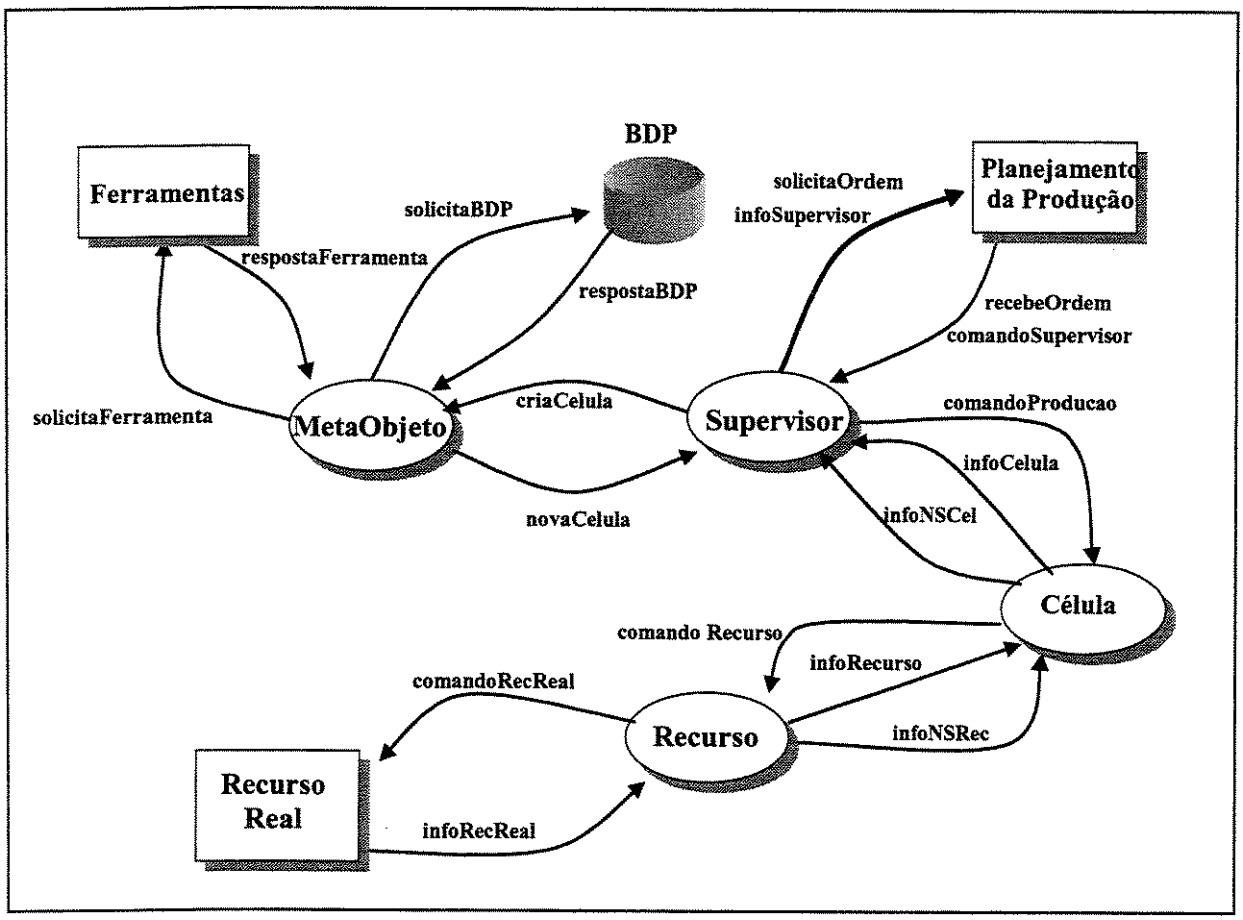


Figura 4.4: Diagrama Geral de Syroco-2

A interação entre os agentes, representadas como fluxo de mensagens na Figura 4.4, foi detalhada usando UML (*Unified modeling language*). Na falta de uma ferramenta de suporte automática este detalhamento foi feito manualmente.

4.3.3.1 Descrição das Mensagens

As mensagens definidas são:

1. solicitaFerramenta
2. respostaFerramenta
3. solicitaBDP
4. respostaBDP
5. criaCelula
6. novaCelula
7. solicitaOrdem
8. recebeOrdem
9. comandoSupervisor
10. infoSupervisor
11. comandoProducao
12. infoCelula
13. infoNSCel
14. comandoRecurso
15. infoRecurso
16. infoNSRec
17. comandoRecReal
18. infoRecReal

Cada mensagem é descrita utilizando a formalização apresentada no capítulo anterior. O funcionamento do sistema é baseado em requisições de serviços e respostas às requisições, nos agentes Célula e Recurso, que são puramente reativos, são adicionadas mensagens que eqüivalem a respostas à requisições não solicitadas (infoNSRec e InfoNSCel). Este tipo de mensagem foi inspirado na norma MMS, onde elas são usadas para avisar da ocorrência de eventos imprevistos e permitir uma reação rápida do sistema.

É mais fácil compreender as mensagens no contexto de funcionamento do sistema, assim no item seguinte será mostrado como os objetos se interrelacionam através da troca de mensagens, explicitando-as.

4.3.3.2 Descrição dos Agentes

O agente Supervisor é responsável pela execução das ordens de produção. Ele pede ao Planejamento da Produção, usando **solicitaOrdem**, um conjunto de ordens de produção para

serem executadas. Recebe as ordens via mensagem **recebeOrdem** e passa a gerenciar a execução das mesmas.

Ele pode gerenciar muitas células, apesar de no protótipo ter sido usada apenas uma, e é responsável pela sincronização entre elas. Ele pode trabalhar de duas formas, a primeira gerenciando a execução de ordens previamente otimizadas (recebidas do Planejamento da Produção) ou mapeando objetivos de produção em novas ordens geradas internamente (esse caso não é considerado na atual implementação).

O agente Meta-Objeto age sob demanda do Supervisor que lhe solicita a criação (lógica) de uma nova célula para a execução de uma certa ordem de produção (mensagem **criaCelula**). Ele cria agentes Célula (de acordo com a estrutura dos produtos), consulta planos de processo na Base de Dados de Produção através de **solicitaBDP** / **respostaBDP** e instancia os roteiros de fabricação (caminhos) a serem seguidos pela Célula criada. Este agente gera um escalonamento para cada produto, através do uso do módulo externo Ferramentas, usando as mensagens **solicitaFerramenta** / **respostaFerramenta**.

Como ele usa ferramentas externas de escalonamento, ele pode tratar diferentes tipos de sistemas produtivos, através da seleção de algoritmos de escalonamento adequados.

Os agentes Célula recebem do agente Supervisor sua configuração (que foi criada pelo Meta-Objeto) mensagens **comandoProdução**, sabendo então qual produto deverá ser fabricado e quais recursos serão utilizados. Os agentes Célula estão associados conceitualmente com um produto específico e são responsáveis por seu ciclo de fabricação. O conceito de célula aqui considerado é o de célula virtual e não de célula física, isto significa que qualquer recurso do chão de fábrica pode ser utilizado pelos agentes Célula.

No final da produção do produto requisitado o agente Célula informa o Supervisor através da mensagem **infoCélula**. Durante a produção também podem ser recebidas outras mensagens **comandoProdução**, que conterão uma ordem para fornecimento do estado da produção, neste caso a Célula responde com a mensagem **infoCélula** contendo o estado requisitado. No caso da ocorrência de um imprevisto na produção, a Célula pode informar o Supervisor através de uma mensagem **infoNSCel**, para que este tome as providências cabíveis enquanto a Célula continua a produzir (se possível).

Os agentes Recursos são ligados aos recursos físicos, tipicamente máquinas. Eles guiam a operação dos equipamentos e provêem todas as informações de estado relacionadas à célula virtual (mensagens **comandoRecurso**, **infoRecurso** e **infoNSRec**) . Sua ligação com os recursos físicos depende do grau de inteligência destes, eles podem ser embutidos nestes equipamentos físicos ou podem formar uma camada de empacotamento para os recursos físicos, provendo uma interface para o sistema de controle.

4.3.3.3 Operação Normal

Durante a operação normal do sistema os agentes Recurso executam as operações, controlando os recursos físicos disponíveis, e mantém o agente Célula informado através da troca de mensagens **comandoRecurso** / **infoRecurso** ou **infoNSRec**. A Célula informa o Supervisor sobre o fim de cada operação, significando que a quantidade demandada de produto foi produzida (mensagem infoCélula). O Supervisor mantém o controle da produção, verificando que o escalonamento está sendo realizado, e informa o estado da produção ao módulo Planejamento da Produção quando solicitado (mensagens **comandoSupervisor** / **infoSupervisor**).

4.3.3.4 Falha de Máquina

Quando uma falha ocorre em uma máquina o agente Recurso que supervisiona aquela máquina toma ciência do ocorrido. Ele informa ao agente Célula a ocorrência da falha, através da mensagem **infoNSRec**. O agente Célula, por sua vez, informa o agente Supervisor (mensagem **infoNSCel**)o qual irá ativar um processo de reescalonamento dinâmico da produção.

O Supervisor requisita ao Meta-Objeto para gerar um novo escalonamento, este baseando-se no estado atual da produção, requisita ao módulo externo Ferramentas a execução de um algoritmo adequado de reescalonamento. Após receber o novo escalonamento o Meta-Objeto gera a configuração de um novo agente Célula, que será responsável pela continuação da produção do produto que tiver sido afetado pela falha. O Supervisor recebe esta nova configuração (que contém também o código do novo agente Célula), via mensagem **novaCélula**.

Ele decide em qual computador disponível as novas Células serão ativadas, termina a operação da Célula que estava em falha, usando uma mensagem **comandoProdução** (parâmetro

"terminar", resposta **infoCélula**) e ativa a nova Célula usando **comandoProdução** com o parâmetro de "iniciar". A ordem de ativação das Células é baseada no novo escalonamento que foi gerado. Uma vez a nova Célula ativada o sistema retorna à operação normal.

4.3.3.5 Chegada de pedido prioritário

Uma outra possibilidade de perturbação pode ocorrer quando uma nova ordem (pedido) é inserido no sistema, com prioridade de produção.

Neste caso o Supervisor reconhece que a ordem (obtida via comandoSupervisor) é prioritária e determina ao Meta-Objeto para reescalonar a produção, considerando esta restrição. Neste caso ele não interrompe a fabricação do produto corrente, e espera até que ele termine.

Enquanto o produto está sendo terminado o Meta-Objeto realiza o novo escalonamento (como descrito para o caso anterior), que irá inserir a ordem recebida como a primeira a ser realizada, modificando as outras de acordo com as possibilidades.

Quando a célula informar que o produto está terminado (mensagem infoCélula com parâmetro "fim") o Supervisor irá ativar a Célula seguinte, que será responsável por realizar a ordem prioritária. A partir daí a operação segue normalmente, de acordo com o novo escalonamento.

Para qualquer um dos casos de operação pode ocorrer que o escalonamento não possa ser gerado (pelo módulo Ferramentas) devido à recursos indisponíveis (máquinas, matérias primas, etc.), neste caso o Supervisor avisará ao Planejamento da Produção (infoSupervisor em resposta ao comandoSupervisor que determinava iniciar a produção) para que este corrija os parâmetros necessários na BDP e tome outras medidas adequadas.

4.3.4 Implementação - Protótipos dos Agentes

A Figura 4.5 mostra a configuração de teste do sistema. Os agentes básicos são derivados diretamente da especificação funcional (vide Figura 4.4), eles são: **Supervisor**, **Meta-Objeto**, **Célula** e **Recurso**.

Da mesma especificação funcional, identificou-se as entidades externas **Planejamento da Produção**, **Ferramentas**, **Recursos Reais** e **Base de Dados de Produção - BDP**, as quais interagem com os agentes.

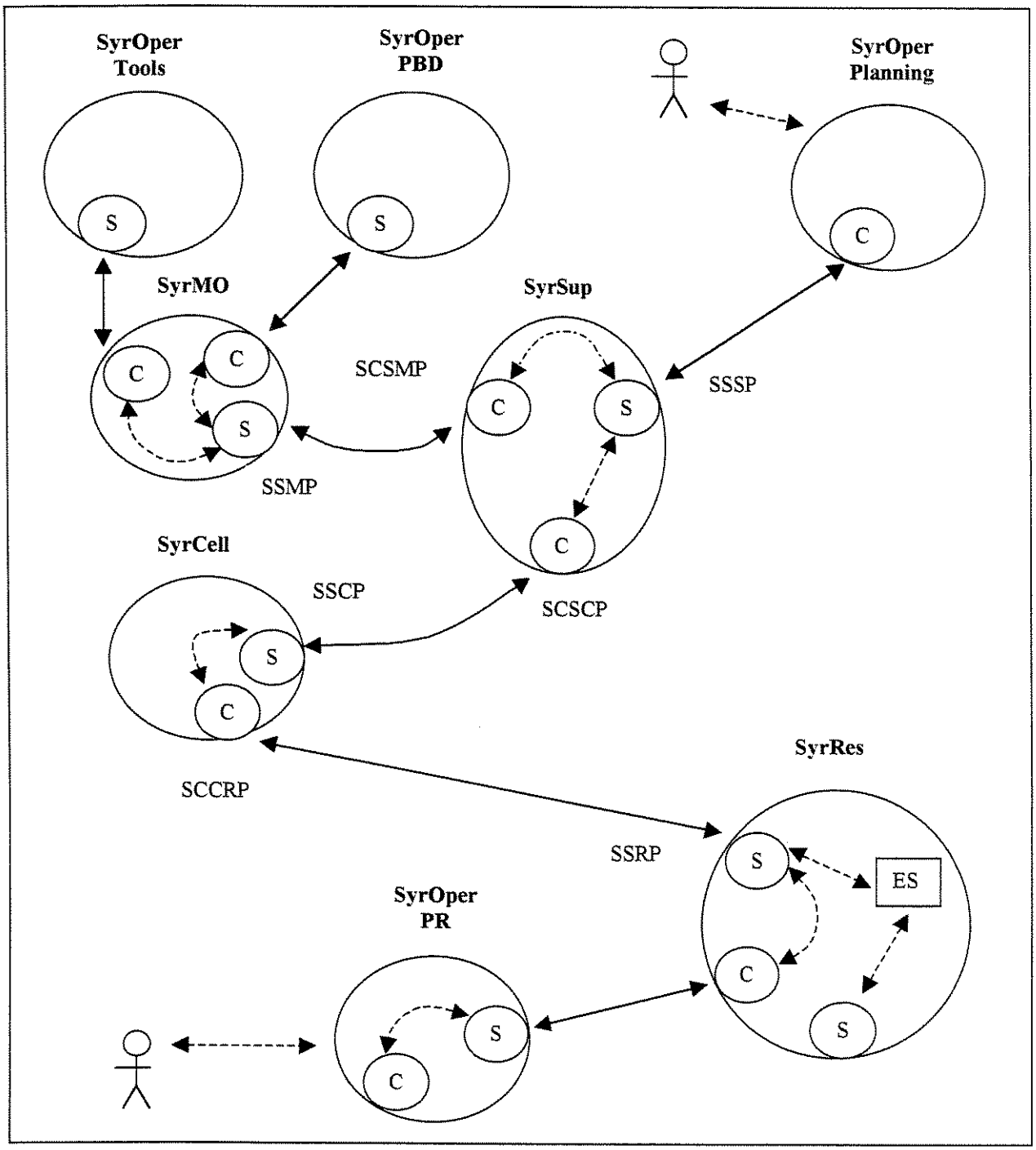


Figura 4.5: Protótipo dos Agentes

Para representar estas entidades e permitir a definição de um protótipo, um agente de interface, chamado **SyrOper** foi criado. Ele pode ser especializado de acordo com o comportamento das entidades externas sendo modeladas.

Os agentes podem estar em um mesmo computador ou espalhados na rede. A arquitetura não é rigidamente ligada a uma configuração física determinada, no entanto, para fins deste protótipo, os agentes devem ter conhecimento prévio da existência dos outros (seus endereços de rede IP).

Encontra-se no Anexo 1 uma descrição de alguns cenários de uso e diagramas mais detalhados que ilustram a metodologia de projeto.

4.3.4.1 O ambiente Industrial emulado pela PIPEFA

A plataforma PIPEFA foi o alvo para a implementação. Ela é usada para fazer experimentos em técnicas de integração e usa aplicativos e componentes comerciais, buscando compatibilidade com o nível de maturidade tecnológica das PMEs, sendo um de seus propósitos a transferência de tecnologia para estas empresas.

Ela é um cooperação entre quatro instituições:

- Faculdade de Engenharia Mecânica da Unicamp, encarregada do Chão de Fábrica (Automação e Mecatrônica).
- Instituto Nacional de Tecnologia de Informação - ITI, encarregado da Infra-estrutura de integração, Planejamento da Produção e Modelagem de Empresa.
- LIISI do ISMCM - CESTI, Toulon, França, encarregado da Modelagem de Empresa (Sistemas Reativos, Sistemas de trocas de Conhecimentos).
- LGIPM, Universidade de Metz, França, encarregado da Modelagem de Empresas e do desenvolvimento de novas arquiteturas para supervisão e controle de chão de fábrica, em conjunto com o ITI.

A Figura 4.6 apresenta a interação entre os parceiros do projeto PIPEFA. ITI, no lado esquerdo atua como o gerador dos Planos de Produção, eles são entregues para serem produzidos na célula de manufatura da Faculdade de Engenharia Mecânica da Unicamp, do lado direito.

Estas instituições estão separadas por uma distância de dez quilômetros e são conectadas

através de um rádio enlace Internet.

No ITI existe um aplicativo de Planejamento da Produção comercial (LOGIX) e um aplicativo experimental de otimização (RETRA) os quais trabalham em conjunto com um Sistema de Supervisão e Controle comercial (Wizcon) , este último também é usado na Unicamp e permite que a célula seja monitorada localmente na Unicamp ou remotamente no ITI.

O Chão de Fábrica (célula) é organizada em torno de um sistema circular de transporte e é composto por cinco postos de trabalho: Um para Carga e outro para Descarga de produtos (C/D, representados por um único posto na figura), dois postos de Montagem e Desmontagem (M/D) e um posto de Inspeção (I).

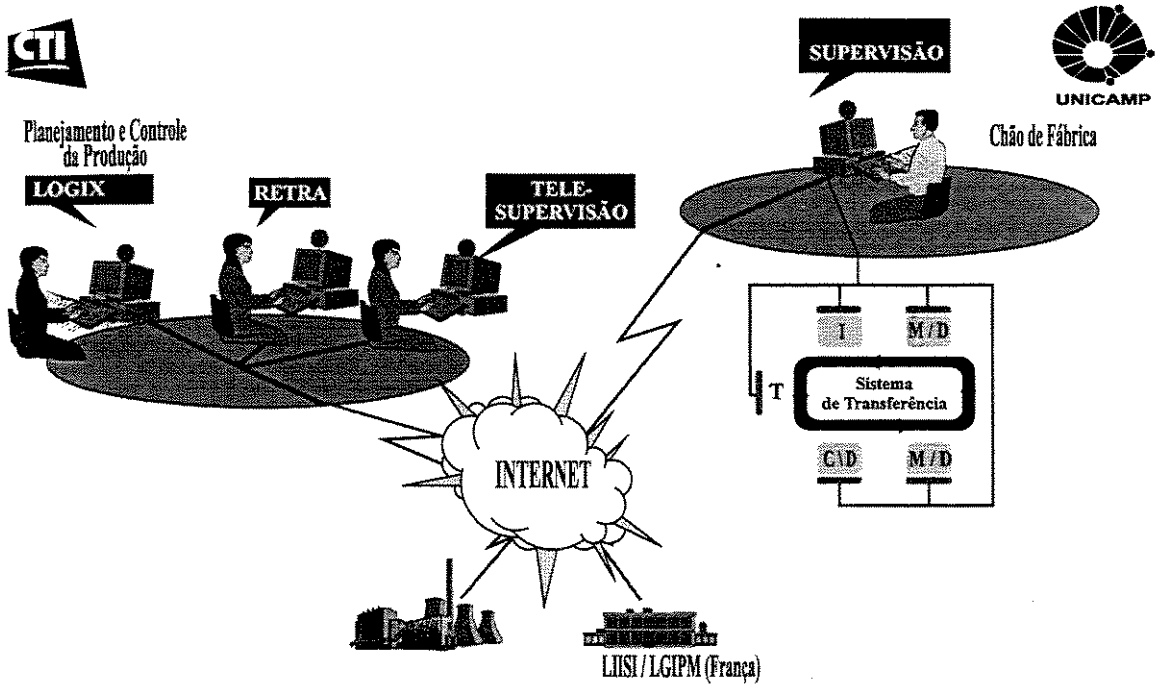


Figura 4.6: PIPEFA

Ao invés de produzir peças usinadas, a célula foi projetada para realizar montagens, de tal forma a não ter custos extras associados às máquinas ferramentas (materiais, manutenção e capital imobilizado).

O produto a ser montado é uma combinação de uma peça base de componentes coloridos (cubos). São usados blocos Lego. A montagem dos cubos na base pode ser feita em diferentes

posições e níveis, de tal forma que um total de 26 variações de produtos são possíveis.

Esta plataforma representa bem o problema de implementar uma arquitetura para supervisão e controle de chão de fábrica, com suas máquinas, controladores programáveis e aplicações já instaladas ("aplicações herdadas").

Considerou-se o sistema existente como sendo o "Sistema Herdado". A supervisão e controle é realizada através de uma aplicação comercial chamada Wizcon (também é possível utilizar outra na mesma plataforma), contendo programas para controlar os postos de trabalho e esteira, através do uso dos Controladores Programáveis (CP). CPs permitem a cada posto uma operação autônoma, o sistema de supervisão é usado para a apresentação das interfaces gráficas Homem-Máquina, carga dos programas dos CPs e para sincronizar a produção. Esta é uma configuração usual nos Chãos de Fábrica. Mudar as aplicações pode ter um alto custo devido ao fato de dúzias de sensores, atuadores e equipamentos diferentes de controle serem usados.

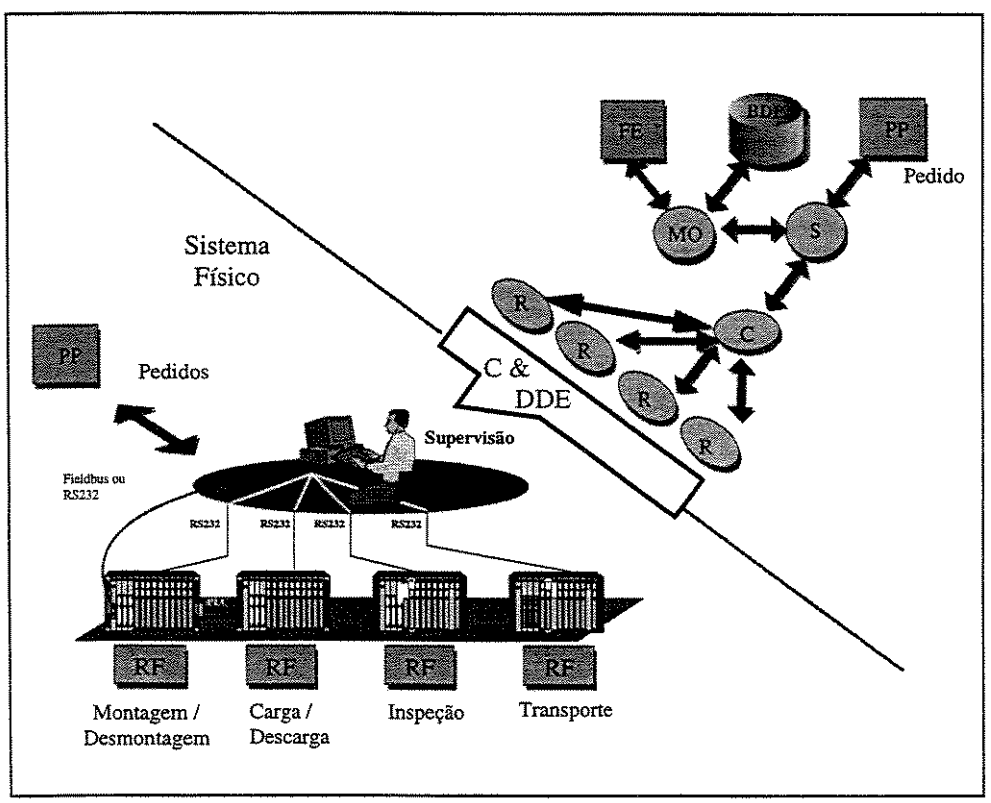


Figura 4.7: Estratégia de Migração

Será mostrada a estratégia para preservar estas aplicações e ao mesmo tempo melhorar a

reatividade do sistema através da incorporação da arquitetura SYROCO-2.

O Sistema de Supervisão existente é mantido com todas as suas aplicações já desenvolvidas (interface Homem-Máquina e programas de controle) e é criada uma interface entre ele os agentes Recurso (os quais neste caso poderão ser agrupados em um único agente), usando a capacidade de exportação de dados do sistema de supervisão. Este permite tanto exportar um arquivo de dados quanto exportar dinamicamente os dados através do protocolo DDE (*Dynamic Data Exchange*) da Microsoft. Está sendo desenvolvido um programa de interface, capaz de aceitar dados em DDE, usando a linguagem C.

No caso de falha de recursos físicos ou da chegada de novos pedidos, a arquitetura detecta os eventos e reconfigura a produção. Para fins de sincronização, é necessário que SYROCO-2 informe ao sistema de supervisão que a produção foi mudada, já que este não acessa o BDP, isto é feito com uma mudança nos arquivos de configuração do sistema de supervisão.

4.3.4.2 Avaliação da Implementação

Testes foram realizados usando dois computadores, sem usar recursos reais (postos de trabalho). O tempo de resposta foi da ordem de um segundo.

Muitos problemas relacionados à linguagem Java surgiram. A comunicação usando as portas TCP/IP, em Java, mostrou-se restritiva e insuficiente, mesmo sendo fácil de programar, É necessário haver suporte para comunicação assíncrona baseada em interrupções, não disponíveis em Java até o momento.

O esquema global de comunicação foi fortemente baseado em temporizações para verificar a chegada de mensagens. Tal esquema não é suficientemente confiável para esta classe de aplicação.

Outra dificuldade é relacionada com a geração dinâmica de agentes, necessária durante a reconfiguração do sistema. É necessário transferir o código do agente e ativá-lo. Java tem fortes restrições de segurança que dificultam este tipo de instanciação, mesmo não sendo proibido pela linguagem.

Uma solução baseada em carregamento dinâmico de classes foi encontrada. Mesmo sendo 100% compatível com Java ela pode ter problemas em versões futuras da linguagem, já que parece haver uma tendência dos projetistas da linguagem Java na direção de RMI (*Remote Method Invocation*) e CORBA (*Common Object Request Broker Architecture*) [Orfali 98].

A implementação continua sendo feita, faltando apenas:

- a conexão de SYROCO-2 com o Sistema de Supervisão e Controle já existente, através de uma interface escrita na linguagem C a qual utiliza o recurso DDE (*Dynamic Data Exchange*) do Windows

- a finalização do módulo de escalonamento, o qual permitirá a experimentação com o algoritmo de busca local selecionado e poderá ser expandido para incluir outros.

O término desses trabalhos necessita da colaboração com outros pesquisadores, os quais complementarão a arquitetura com as suas pesquisas e desenvolvimentos. O espírito da plataforma PIPEFA é justamente o de colocar em contato uma equipe multi-disciplinar, de tal forma que todos ganhem com o intercâmbio de informações.

4.4 Conclusão

Apresentou-se neste capítulo um exemplo de como a Arquitetura Experimental desenvolvida, baseada em Componentes Genéricos que utilizam a tecnologia de Sistemas Multi-Agentes, pode ser particularizada para atender à uma aplicação específica. No caso a aplicação foi o desenvolvimento de uma Arquitetura para Supervisão e Controle para a Plataforma PIPEFA.

PIPEFA é representativa da classe de sistemas de montagem e utiliza recursos mecânicos e computacionais tecnologicamente atualizados, podendo portanto servir como protótipo para uma implementação na indústria.

Foram mostradas duas arquiteturas existentes com possibilidade de servirem de modelo para o controle da PIPEFA, outras poderiam ser desenvolvidas, mas a seleção dessas duas tem o propósito de mostrar que a metodologia desenvolvida permite reaproveitar trabalhos anteriores. Foi selecionada a arquitetura SYROCO, esta foi analisada em detalhes, através de uma

cooperação estreita com a Universidade de Metz (estágio de doutorado sanduíche), e foi desenvolvida uma variante chamada SYROCO-2 a qual foi o ponto de partida para o processo de especialização da Arquitetura Experimental.

O desenvolvimento conceitual dos Agentes de SYROCO-2, baseados nos Componentes Genéricos, foi uma fase tranqüila, o maior problema foi a redefinição das mensagens de comunicação, que não estavam bem formalizadas. Acredita-se que a formalização sintática desenvolvida dentro da metodologia mostrou-se correta e útil.

Verificou-se também que a descrição dinâmica de SYROCO, ao nível macro, feita com os Diagramas de Estado (*StateCharts*) apesar de formalmente correta, ainda trazia problemas de compreensão ao implementador. No mesmo nível macro, a representação paralela com os Diagramas de Fluxo de Dados propostos por Ward-Mellor colaborou para melhorar a legibilidade das especificações.

No nível micro, de detalhamento das especificações, o uso de máquinas de estado com um conjunto de pré condições, ações e pós condições, mostrou ser uma boa opção para facilitar a implementação.

A linguagem UML foi utilizada de maneira referencial, buscou-se combinar as técnicas propostas para UML por [Douglass 98] com aquelas técnicas mais tradicionais de Ward-Mellor [Ward 85] e Hatley-Phirbay [Hatley 88] . Este enfoque mostrou-se bem sucedido e adequado para o problema de reinterpretar uma arquitetura existente e reprojetá-la dentro do paradigma proposto dos Componentes Genéricos.

A implementação ainda não foi totalmente completada e não é possível mensurar o desempenho do sistema, porém se considerarmos que, o ciclo de controle normal do sistema já existente na PIPEFA, é de um segundo, e que este tempo é apenas relativo à leitura e escrita de variáveis, pode-se afirmar que: se o tempo de geração de um novo escalonamento for de alguns segundos então a arquitetura SYROCO-2 terá trazido à PIPEFA uma capacidade de reação bastante rápida e mais do que suficiente para o tipo de produto sendo fabricado.

Considerando ainda que isto será feito sem interromper o funcionamento do sistema existente e sem descartar os investimentos já feitos no parque instalado de aplicativos e

máquinas, então PIPEFA poderá ganhar esta nova capacidade, inexistente até então, com custos relativamente baixos.

Do ponto de vista de desempenho do código gerado a avaliação é de que ele é bastante baixo , devido basicamente à codificação em Java, e pode ser melhorado. Trabalho futuros irão indicar se a linguagem Java é limitante para a melhoria do desempenho e se é necessário utilizar outra linguagem. No momento ainda não há uma conclusão definitiva sobre este assunto.

No capítulo seguinte será feita uma avaliação global do projeto, considerando as especificações dos Componentes Genéricos e da Arquitetura Experimental e sua aplicação na plataforma PIPEFA, através da Metodologia de Projeto proposta.

Capítulo 5

Análise dos Experimentos e dos Resultados

Neste capítulo o propósito é fazer uma avaliação crítica do processo de desenvolvimento dos Componentes Genéricos, da Arquitetura Experimental baseada nos mesmos e, finalmente, da prototipagem de uma Arquitetura Particular adequada à plataforma PIPEFA, que foi chamada SYROCO-2.

A discussão será iniciada avaliando-se a influência das Arquiteturas de Referência (para Integração de Empresa) e suas Infra-estruturas de Integração, a seguir serão discutidos os Sistemas Multi-agentes e a influência do protocolo para comunicação industrial MMS.

Continuar-se-á a discussão com uma análise da Arquitetura Experimental e da metodologia de desenvolvimento proposta.

A finalização será com a discussão da aplicação dessa metodologia para desenvolver a arquitetura SYROCO-2.

5.1 Influência das Arquiteturas de Controle e de Referência.

O problema em foco é controlar um chão de fábrica, assim uma necessidade óbvia era estudar as diversas possibilidades de controle de chão de fábrica descritas na literatura, sobre este tema versou o item 2.1 do Capítulo 2 (Arquiteturas para Supervisão e Controle).

Observou-se no entanto que o enfoque tradicional é centrado em aspectos de controle e escalonamento da produção. Como mostrou-se no item 2.2 do Capítulo 2 (Integração das Funções de Supervisão e Controle do Chão de Fábrica), o chão de fábrica é um dos componentes da

empresa, é necessário inseri-lo dentro deste contexto, para que novas arquiteturas de controle possam ser desenvolvidas, aproveitando-se de uma visão global do sistema e da possibilidade de descentralização de decisões.

As Arquiteturas de Referência, se por um lado oferecem um quadro abrangente da empresa, através das diversas vistas preconizadas, por outro lado deixam a desejar no aspecto de implementação de sistemas e detalhamento da operação, em especial do subsistema produtivo.

Verificou-se que todas as Arquiteturas de Referência apresentam uma evolução temporal do projeto de integração, o chamado "Ciclo de Vida de Projeto". O ciclo de vida mais completo é aquele que foi originalmente concebido na PERA, e que está sendo incorporado nos esforços mais recentes de padronização, representados por GERAM.

É interessante observar que, durante a existência de um empreendimento, diversos ciclos de vida, relativos a aspectos distintos, são perceptíveis. O primeiro é o próprio "Ciclo de Vida de Projeto de Integração", assunto primeiro das Arquiteturas de Referência. Um outro é o "Ciclo de Vida de Produto", representado na Figura 2.10 (Visão Geral dos Conceitos CIMOSA), o qual tem diversas ocorrências durante a fase de operação da empresa e além disso possui diversas instâncias (produtos diferentes são projetados / produzidos ao mesmo tempo).

Além disso, como a empresa é um ente dinâmico, podem existir outros ciclos durante a fase de operação. No chão de fábrica esta dinâmica é bem perceptível, e está de alguma maneira associada aos ciclos de vida de produtos, pois quando novos produtos são desenvolvidos novas arquiteturas físicas (*layouts*) e lógicas (de controle) podem ser necessárias. Na menor escala de tempo aparecem também os Ciclos de Produção, que são relacionados à produção dos produtos em lotes especificados nos pedidos enviados ao chão de Fábrica [Doumeingts 84].

Um outro aspecto comum a todas as Arquiteturas de Referência é a utilização de eventos para a sincronização de funções, eles estão mais evidentes na cadeia de processos de ARIS (Figura 2.8, Capítulo 2) e na modelagem de processos de negócios utilizada em CIMOSA (a qual não foi detalhada nesta tese, vide por exemplo [CIMOSA 93], [Vernadat 96]).

Quando se trata de trazer estes conceitos para o campo prático, para permitir inclusive a operação da empresa, notou-se uma lacuna. Somente CIMOSA e GERAM apresentam propostas

de Infra-estruturas de Integração e, como viu-se no Capítulo 5, item 2.3.2 "Infra-estruturas de Integração", os resultados experimentais ainda não são concludentes.

Dessa análise das Arquiteturas de Controle e de Referência verificou-se então que é necessário considerar, em um projeto de arquitetura de supervisão e controle de chão de fábrica, o contexto geral da empresa e os três ciclos seguintes:

- Ciclo de Vida do Projeto de Integração - para permitir inserção no contexto global
- Ciclo de Vida do Produto - para permitir flexibilidade na operação e reconfiguração
- Ciclo de Produção - para a produtividade, flexibilidade e reatividade

Concluiu-se que: um enfoque em modelagem da empresa, no nível específico de operação do chão de fábrica, seria útil para relacionar esses três ciclos. Tal modelagem deveria ser orientada a eventos, de maneira condizente com os esforços já desenvolvidos.

5.2 Arquiteturas de Supervisão e Controle e Sistemas Multi-agentes.

Conforme apresentado no Capítulo 2, os SMAs estão encontrando aceitação no campo de controle de chão de fábrica, devido às vantagens lá citadas.

O enfoque foi o de levantar quais aspectos seriam relevantes para o projeto de um SMA, dentro de uma linha de modelagem aderente aos padrões de desenvolvimento de sistemas informatizados.

Os aspectos levantados foram: composição, configuração, comunicação e coordenação / negociação. Buscou-se desenvolver um modelo de Componente Genérico considerando tais aspectos.

A comunicação apresentou-se como o aspecto polarizador neste projeto, os componentes interagem através da troca de mensagens, dentro do espírito de sistemas baseados em eventos (as mensagens carregam eventos e dados ou sua recepção / emissão ativam eventos de importância para o sistema). Nota-se a influência do MMS, do qual foram extraídos auxílios para a formalização das mensagens na Arquitetura Experimental e para a definição da funcionalidade das mensagens no protótipo.

A questão do desenvolvimento de um sistema de mensagens adequado à produção é campo em aberto. O MMS aparece como referência apenas para a comunicação com os dispositivos de manufatura, a interface com o Planejamento da Produção ainda está por ser feita, apesar dos esforços iniciais com Ontologias para Produção que foram apontados no Capítulo 2.

Outros pontos de projeto podem ser discutidos, por exemplo qual seria a composição e configuração mínima possível para a Arquitetura Experimental? A questão é parecida com aquela da definição das camadas de uma Rede Neural: a resposta depende da aplicação, mas certamente uma Rede Neural terá uma camada de interface com as entradas (estimulações) e outra com as saídas. Por exemplo, para decodificar os dígitos (0- 9) serão necessárias as entradas (por exemplo uma matriz binária 7 X 9) e as dez saídas, cada uma correspondendo a um dígito.

Da mesma maneira vê-se que o chão de fábrica possui duas interfaces claras, uma com o Planejamento da Produção e a outra com os dispositivos físicos. O paralelo com redes neurais não é completo aqui, pois têm-se entradas e saídas em ambas as interfaces.

Como não é possível antecipar as aplicações foi proposto o modelo com 2 + 1 camadas (Supervisão, Recursos e Opcionais).

A questão da coordenação / negociação, também depende da classe de sistema que se pretende desenvolver para uma certa aplicação. O modelo proposto de agente é bastante simples e esta simplicidade é intencional, pois pretende-se evoluir a especificação no sentido de gerar ferramentas automáticas de suporte ao desenvolvimento de aplicações.

No entanto a simplicidade não implica em demasiada perda de generalidade, de vez que, apesar de parecer uma solução baseada em reatividade (no contexto de inteligência artificial), ainda assim é possível construir sistemas inteligentes com esta estrutura. Basta lembrar que os sistemas peritos são baseados em regras e tais regras podem ser incorporadas de maneira quase imediata no modelo proposto.

Da mesma maneira a incorporação de protocolos de negociação mais refinados, baseados em contratos (*Contract-Nets*) [Smith 80] ou em atos discursivos (*Speech-Acts*) [Wooldridge 92], é possível dentro do modelo proposto.

5.3 Influência das Infra-estruturas de Integração (IEIs)

Tanto CIMOSA quanto GERAM (através da EMEIS) baseiam suas IEIs no conceito de serviços. É justamente este o conceito que apresenta interesse para esta tese. Os Componentes Genéricos possuem uma certa funcionalidade que é colocada à disposição na comunidade através da oferta de serviços.

O acesso aos serviços é feito através de requisições e respostas, de maneira semelhante ao protocolo MMS.

CIMOSA tem uma influência clara do MMS na especificação de seus serviços de apresentação. Na EMEIS existem os serviços de execução de modelos, que deverão ter uma funcionalidade semelhante.

Não foi o propósito fazer um mapeamento das IEIs CIMOSA ou GERAM, não buscou-se nesse momento compatibilidade com as mesmas. No entanto elas tiveram forte influência conceitual, mesmo porque elas são uma condensação das melhores práticas para o desenvolvimento e operação de sistemas abertos.

Os aspectos mais influentes das IEIs sobre este trabalho foram a especificação orientada a serviços e a visão dos componentes genéricos como sendo os atores da execução de modelos, desempenhando o papel de Entidades Funcionais que executam Funções Operacionais, numa analogia aos modelos CIMOSA.

5.4 Desenvolvimento do Protótipo e Avaliação da Metodologia

No Capítulo 3 apresentou-se o conceito de Componente Genérico e esquematizou-se uma metodologia para desenvolvimento de aplicações baseada nestes componentes. Desenvolveu-se também uma Arquitetura Experimental que utiliza estes componentes e os organiza de uma maneira estruturada e didática para ilustrar o problema geral de supervisão e controle de chão de fábrica.

Esta Arquitetura Experimental visa apenas servir como referência inicial a partir da qual o projetista pode tentar enquadrar o seu problema real, através da derivação de uma Arquitetura Particular adequada à esse problema.

Como o processo de derivação não é rígido, diversas soluções para um único problema podem ser encontradas. A determinação de qual dentre elas é a melhor não é o propósito deste trabalho, devendo o projetista estabelecer as medidas de comparação que lhe permitam escolher entre diversas alternativas, se houverem.

O propósito foi o de criar um quadro conceitual, suportado por um conjunto mínimo de componentes implementados, que permitisse facilitar o trabalho de projeto e implementação de uma arquitetura para controle do chão de fábrica.

Falou-se anteriormente de três diferentes ciclos de vida, o chão de fábrica será posicionado em relação a eles.

No tocante ao "Ciclo de Vida do Projeto de Integração" deve-se considerar o chão de fábrica em todas as suas fases. Este trabalho procura colocar à disposição elementos de modelagem para auxiliar no projeto de um sistema de operação e controle, estando então mais fortemente relacionado com as fases de Projeto, Implementação e Operação (ou Execução).

Durante a Operação da Empresa, podem estar em curso múltiplos "Ciclos de Vida de Produto", os quais podem ter fases como: Especificação de Requisitos de Mercado, Projeto, Liberação, Fabricação, Distribuição e Vendas. Dentro da fase de Fabricação encontra-se o "Ciclo de Produção" o qual é regido pela arquitetura de supervisão e controle, objeto deste trabalho.

5.4.1 Levantamento do Sistema Existente e Especificação dos Requisitos

Esta-se de acordo com Parunak quando este diz que SMAs para aplicações industriais devem ser capazes de enfrentar as fases do ciclo de vida de um projeto de engenharia [Parunak 98]. Devido à essa convicção enfatizou-se na metodologia o acompanhamento da evolução do projeto da arquitetura acompanhando um ciclo de desenvolvimento.

Na primeira fase deste ciclo consta a "Especificação dos Requisitos" da arquitetura, o que implica levantar o estado atual do sistema de produção, levando em consideração o chamado "Sistema Herdado" (*Legacy System*).

Os Sistemas Herdados são importantes para as empresas, pois significam investimentos e

treinamentos já feitos e com os quais elas têm comprometimentos. A tendência normal é de tentar incorporar estes sistemas, desde que eles não sejam o elemento de degradação da eficiência (o que ocorre se eles forem obsoletos).

Na aplicação foi bem ilustrada esta problemática, mostrando que o novo sistema, o qual deveria aportar uma capacidade de reagir aos imprevistos de maneira automática, deveria conviver com o sistema já em operação.

5.4.2 Fase de Projeto

Na fase seguinte, que é de Projeto (Funcional e Detalhado), são exercitados os Componentes Genéricos, na forma dos agentes desenvolvidos na Arquitetura Experimental.

Este exercício pode ser feito, num primeiro instante, através da criação de variações de arquiteturas. Por exemplo mostrou-se que, para o caso do protótipo, duas arquiteturas já haviam sido utilizadas em situações semelhantes. O projetista poderia fazer uma prototipagem rápidas das arquiteturas possíveis, comparando-as, antes de partir para o projeto detalhado final.

Nesta fase, a metodologia propõe a utilização de UML, em moldes semelhantes ao apregoado em [Douglass 98], porém julgou-se que, como apontado anteriormente, é usual a convivência com Sistemas Herdados, tal fato leva a incorporar uma capacidade de análise da situação atual da empresa, uma espécie de reengenharia ou engenharia reversa.

Neste sentido foram feitos experimentos com as técnicas clássicas de projeto de tempo real [Ward 85], [Hatley 88], especialmente com o uso de Diagrama de Fluxo de Dados, para estabelecer o contexto da aplicação e a interação entre os módulos (agentes) principais.

A avaliação é que tal combinação técnica é meritória nesse caso. Para o caso de um sistema totalmente novo, acredita-se que pode ser usada UML na sua forma pura sem maiores problemas.

Quanto ao modelo de agente desenvolvido avaliou-se que para o protótipo ele se mostrou satisfatório. Notou-se no entanto que o detalhamento necessário de todos os eventos (relacionados às mensagens ou internos) e das ações associadas, obtidos dos diagramas de seqüência e das especificações com diagramas de estado (*Statecharts*) resultou muito complexa.

Acredita-se que esta complexidade não é um defeito do modelo, que foi pensado para evoluir dentro de um sistema de geração semi-automática de aplicações, mas é um indicador da necessidade de usar sistemas de suporte ao desenvolvimentos de aplicativos (CASE - *Computer-Aided Software Engineering*). Ocorre que, como já foi observado no Capítulo 3, item 3.22 (Agentificação), ainda não há sistemas de suporte ao desenvolvimento de SMAs. Os sistemas UML existentes são genéricos e para projeto orientado à objetos.

O modelo e metodologia desenvolvidos ensaiam ser uma contribuição nesse sentido.

5.4.3 Fase de Implementação

A implementação dividiu-se em duas partes, a primeira tratou dos Componentes Genéricos, implementados em Java de acordo com a Arquitetura Experimental. Para a segunda foi feita uma reengenharia do Projeto SYROCO [Roy 98], e foi feita uma programação totalmente nova, a partir da especificação atualizada e adaptada à PIPEFA, que foi chamada SYROCO-2 (Figura 4.4 do Capítulo 4).

A estruturação dos dados proposta na metodologia, a qual já havia sido utilizada na PIPEFA, mostrou-se satisfatória. A coordenação dos agentes foi representada pelos DFDs e por diagramas de estados (*Statecharts*). Julga-se que é necessária uma ferramenta de suporte aos diagramas de estado, para permitir a avaliação dinâmica dos mesmos. As ferramentas mais avançadas disponíveis suportam a animação e simulação dos modelos, o que permite encontrar falhas na especificação de maneira antecipada. O orçamento não permitiu dispor dessas ferramentas durante o decorrer deste projeto.

Para a implementação foi decidido usar a versão de Java distribuída pela *Sun Microsystems* chamada JDK 1.22. Esta versão já suporta RMI (*Remote Method Invocation*) e interface com CORBA. Como já argumentado, optou-se por um código 100%, Java mas buscou-se deixar espaço para migrar para soluções híbridas.

Foram usadas técnicas de prototipagem rápida para o desenvolvimento dos agentes, as quais permitiram corrigir as especificações funcionais quando necessário.

Sabe-se da literatura que o desempenho de aplicações Java é inferior ao de aplicações

escritas em linguagens compiladas (como por exemplo c e C++). O código interpretado Java é entre 5 e 30 vezes mais lento que o código gerado por estas linguagens.

O protótipo desenvolvido apresentou um tempo de resposta da ordem de um segundo, usando duas máquinas de baixa performance (mas representativas do disponível nas indústrias), um Pentium 75 Mega-hertz (16 Mega-bytes de memória) e outro de 166 Mega-hertz de relógio e 32 Mega-bytes de memória.

A avaliação é de que este tempo, apesar de alto em relação à complexidade da aplicação, é bom em relação aos requisitos do sistema real representado pela PIPEFA, onde a operação de cada posto de trabalho é realizada na ordem de segundos. Assim, quando o protótipo estiver finalizado, a reconfiguração da PIPEFA será feita dentro do "Ciclo de Produção" e dentro do ciclo menor que é a execução de uma operação básica.

Observou-se que, para permitir prototipar a aplicação, muita energia foi investida em módulos externos, que não estavam ainda disponíveis (na forma necessária) no sistema PIPEFA. Estes módulos foram: Ferramentas de Reescalonamento, Banco de Dados de Produção, Planejamento da Produção e Recursos Físicos, como indicado na Figura 4.5 do Capítulo 4 (Protótipo dos Agentes). Assim o trabalho de desenvolvimento da arquitetura, que conta com os agentes Supervisor, Meta-objeto, Célula e Recursos, foi praticamente dobrado em extensão, causando uma aumento proporcional do tempo de implementação.

Por outro lado, após a finalização do protótipo, a qual deve ocorrer em breve, a PIPEFA contará com interfaces padronizadas (via agentes), o que lhe permitirá interagir mais eficientemente com seus parceiros atuais e futuros.

Cabe ressaltar que um problema encontrado durante a implementação é a existência de proteções de rede do tipo "paredes corta fogo" (*Firewalls*), as quais dificultam a conexão remota via Internet. Este problema deve ser superado quando o protótipo se estabilizar e puderem ser negociadas portas específicas para conexão, as quais seriam registradas junto aos organismos competentes.

Nesta fase inicial não foram considerados aspectos de segurança de acesso, pois eles impactariam no desempenho e complexidade do sistema. Como a operação da arquitetura está

circunscrita ao chão de fábrica, pode-se supor que é possível garantir a segurança de acesso por mecanismos convencionais, tais quais determinação de subredes. O ponto de acesso externo é a interface com o módulo de Planejamento da Produção, se for necessário poderão ser incorporados mecanismos de segurança para este módulo.

O acesso para o Banco de Dados de Produção, se feito remotamente, poderá utilizar-se dos mecanismos normais de conexão SQL (*Structured Query Language*) [Date 89], que baseiam-se em senhas de acesso. Tal possibilidade já foi anteriormente exercitada na PIPEFA, com um acesso remoto da FEM ao ITI, as restrições de parede corta fogo aplicam-se.

5.4.4 Conclusão

Neste capítulo foi feita uma revisão crítica das propostas de Componentes Genéricos e Metodologia de Projeto, à luz dos resultados já obtidos.

Em geral o enfoque proposto mostrou-se consistente e operacional, não observou-se nenhuma inconsistência conceitual no decorrer da aplicação do método ao desenvolvimento do sistema SYROCO-2.

Este desenvolvimento mostrou claramente a necessidade de utilizar-se ferramentas de suporte do tipo CASE, as quais infelizmente não estão ainda disponíveis para SMAs, sendo objeto de pesquisas e desenvolvimentos. No entanto existem ferramentas UML que podem ser utilizadas de acordo com a metodologia preconizada, em conjunto com ferramentas DFD. A continuação do projeto e finalização da implementação certamente será beneficiada com a aquisição de tais suportes.

Em [Santos 00] encontra-se uma motivação e enfoque parecido a este trabalho, quanto aos mecanismos de execução. Uma diferença é que este trabalho é mais orientados à adaptação ao mundo real, enquanto o trabalho citado, que versa sobre uma linguagem de modelagem de empresa, buscou uma maior aproximação com CIMOSA e GERAM.

O sucesso parcial obtido ainda não permite antecipar bons resultados em outras aplicações. Será necessário finalizar as interfaces existentes, estabilizar o código dos Componentes Genéricos e reaplicá-los, para poder juntar mais evidências favoráveis da eficácia das propostas

apresentadas. O trabalho será continuado neste sentido, para, em futuro próximo, confirmarem-se com mais dados as avaliações.

Capítulo 6

Conclusão

O tema que foi abordado nesta tese foi o Projeto e Construção de Arquiteturas para Supervisão e Controle de Chão de Fábrica.

A abordagem adotada buscou contemplar dois aspectos, freqüentemente negligenciados em trabalhos assemelhados: integração com os outros partícipes da empresa e aproveitamento de conhecimentos adquiridos.

Como foi descrito no Capítulo 1 - Evolução das Empresas, acredita-se que as novas empresas irão mais e mais necessitar de uma compreensão e domínio de seu próprio funcionamento.

Esta compreensão passa pelo "pensamento sistêmico" [Senge 94], pela busca de uma visão global e não departamental da empresa, e pelo processo de auto-conhecimento e gerência do processo de conhecer [Savage 96].

Uma maneira de atingir estes propósitos é através da Modelagem de Empresas e da Engenharia de Empresa [Vernadat 96].

A proposta de basear o desenvolvimento de arquiteturas em Componentes Genéricos visou permitir o desenvolvimento de modelos experimentais, que possam ser parcialmente reutilizados quando mostrarem-se adequados, de forma a compor no longo prazo uma biblioteca de soluções de engenharia.

Verificou-se que as Arquiteturas de Referência (para integração de empresas) existentes são

incompletas no referente ao quesito implementação. Dada a complexidade de tais arquiteturas não buscaou-se uma compatibilização restrita com nenhuma delas, procurando apenas extrair as características mais úteis ao objetivo deste trabalho.

Essa lacuna propiciou então apresentar a primeira contribuição do trabalho:

O desenvolvimento de Componentes Genéricos e de uma Arquitetura Experimental, os quais podem ser particularizados para aplicações específicas e podem ser reutilizados, diminuindo o custo de novos projetos e facilitando a Engenharia de Empresas.

Uma segunda contribuição foi no desenvolvimento de uma Metodologia de Projeto para sistemas de supervisão e controle. Tal metodologia engloba procedimentos clássicos e atuais, dentro de uma ótica de projeto de sistemas de tempo real. Ela contribui com a proposição de um balanço entre estes métodos, mostrando a utilidade de se usar diagramas de fluxo de dados em níveis mais altos de abstração, em conjunto com os diagramas de estado (*Statecharts*) para extrair a dinâmica do sistema, a qual então é detalhada de forma complementar em UML e em máquinas de estado. Verificou-se que esta metodologia se presta bem para fazer uma reengenharia de sistemas existentes, permitindo modelá-los mesmo que eles não sejam "orientados a agentes" ou "orientados a objetos".

A terceira contribuição, ainda incompleta, está na disposição de um protótipo de uma arquitetura particularizada de supervisão e controle, chamada SYROCO-2, que será aplicada à plataforma PIPEFA e permitirá a realização de diversas pesquisas, algumas das quais serão sugeridas ainda neste texto.

Esta arquitetura diferencia-se suficientemente de seu modelo original, o projeto SYROCO [Roy 98], tanto em estrutura lógica quanto em especificação funcional (baseada nos Componentes Genéricos) e desenvolvimento de interfaces e código, para que seja considerada como uma outra contribuição deste trabalho.

6.1 Trabalhos Futuros

Esta tese foi desenvolvida dentro de um ambiente maior que é aquele definido pela plataforma PIPEFA, a qual desde a sua concepção apresentou um caráter multidisciplinar, englobando automação industrial, tecnologia de informação e gerência da produção.

O desenvolvimento teórico realizado e a sua consolidação através do protótipo de SYROCO-2 visou criar uma infra-estrutura que desse suporte à colaboração já existente entre a Faculdade de Engenharia Mecânica da Unicamp, o Instituto Nacional de Tecnologia de Informação - ITI (Ministério de Ciência e Tecnologia) e os laboratórios parceiros LIISI (*Laboratoire d'Ingénierie Intégrée des Systèmes Industriels*) em Toulon, pertencente ao ISMCM - CESTI / (*Institut Supérieur des Matériaux et de la Construction Mécanique - Centre d'Etudes Supérieures des Techniques Industrielles*) e LGIPM (*Laboratoire de Génie Industriel et Production Mécanique*) da Universidade de Metz, ambos na França.

Pode-se apresentar as seguintes possibilidades de trabalhos futuros a serem desenvolvidos pelos parceiros utilizando a infra-estrutura da PIPEFA:

6.1.1 Criação de novas arquiteturas

Uma vez terminada a implementação poderão ser criadas arquiteturas alternativas, as quais poderão ser comparadas segundo diversos fatores tais como: desempenho, custo, complexidade etc. Existem arquiteturas que consideram o produto como um agente, outras consideram os pedidos como agentes; estas diversas possibilidades poderão ser exercitadas utilizando o Componente Genérico e a metodologia desenvolvidos.

6.1.2 Simulação distribuída de ambiente de manufatura

Um campo emergente de utilização de Sistemas Multi-agentes é na simulação distribuída aplicada no contexto de sistemas de manufatura. Dependendo do grau de abstração utilizado, os Componentes Genéricos podem representar todo um subconjunto produtivo (célula ou fábrica) e permitem exercitar simulações do relacionamento entre estes subconjuntos. Uma aplicação que já está sendo estudada é a utilização da estrutura básica de agentes (parte de comunicação e reação a eventos) para a simulação distribuída de cadeia de suprimentos.

Outra aplicação em estudo é a criação de Células Simuladas, semelhantes estruturalmente à

célula de produção da PIPEFA, para que em conjunto com essa última possam compor um problema de complexidade maior, o qual permitirá avaliar o desempenho das arquiteturas para casos mais complexos.

6.1.3 Modelagem de Infra-estrutura de Integração

Como foi dito no trabalho não procurou-se seguir estritamente os modelos preconizados nas IEI CIMOSA e EMEIS. Os trabalhos de padronização da Arquitetura de Referência GERAM continuam, assim como a necessidade de se dispor de uma IEI que permita executar os processos obtidos através da modelagem da empresa e / ou subconjuntos da empresa.

Pretende-se acompanhar a evolução do processo de padronização e contribuir com o desenvolvimento de componentes e blocos funcionais compatíveis com a especificações a serem desenvolvidas.

Este trabalho insere-se num contexto maior que é o de Modelagem de Empresa, que continua a ser desenvolvido no ITI, LGIPM e LIISI, e que potencialmente pode se beneficiar dos resultados a serem obtidos na pesquisa sobres IEI.

6.1.4 Desenvolvimento de algoritmos de escalonamento

Mostrou-se que os algoritmos de escalonamento são fornecidos por um módulo ou ferramenta externa. Foi desenvolvida uma maneira de "agentificar" este módulo, para que o mesmo possa comunicar-se com os componentes criados.

Assim será possível, para um mesmo caso industrial, real ou simulado, comparar-se o desempenho de diversos algoritmos, permitindo adquirir conhecimentos sobre quais são mais adequados ao problema enfrentado.

Estes algoritmos permitirão que uma mesma arquitetura (por exemplo SYROCO-2) possa

ser utilizada em uma variedade de casos, através da seleção adequada do algoritmo de escalonamento.

6.1.5 Cooperação e Gestão do Conhecimento no Chão de Fábrica

Esta linha de pesquisa possivelmente irá extrapolar os limites do chão de fábrica apesar de o tomar como ponto de partida.

Conforme apontou-se no Capítulo 3, ainda não há um consenso sobre ontologias adequadas à Gestão da Produção.

Para explorar a autonomia possível dos agentes, e ao mesmo tempo desenvolver soluções que sejam portáteis e repetitiveis, é necessário padronizar a representação do conhecimento no domínio dos sistemas de manufatura. Possivelmente esta padronização se dará utilizando as linguagens KIF e KQML já existentes, como suporte metodológico.

Em paralelo com este desenvolvimento, é preciso definir formas de relacionamento entre os componentes do sistema de manufatura. Dentro do espírito atual de tornar a empresa mais dinâmica, usando técnicas de Engenharia Concorrente (EC) e Trabalho Cooperativo em Grupo (CSCW - *Computer Supported Cooperative Work*), é preciso colocar o chão de fábrica em contato com os outros componentes da empresa.

A unificação semântica, possibilitada pelas ontologias, é um primeiro passo. O passo seguinte é definir a dinâmica do relacionamento e aí aplicam-se as técnicas já existentes ou em desenvolvimento para EC e CSCW. Uma tendência é a pesquisa usando "Atos discursivos" (*Speech Acts*) para permitir a interação entre os componentes.

Encontra-se em curso um outro projeto de doutorado [Araújo 00], que pretende investigar a cooperação entre os parceiros da PIPEFA. Os resultados desse trabalho certamente fornecerão subsídios para a questão do relacionamento entre o chão de fábrica e os demais componentes da empresa e servirá de suporte para a evolução nessa linha de pesquisa.

Referências Bibliográficas

[AFN 90] AFNOR, "FIP - *Flux d'Information Processus*", NF C 46_602, 1990

[Agostinho 91] Agostinho, O. L., "Manufatura Integrada por Computador", IPESI- Metal Mecânica, p. 71-90, nov. / dez. 1991

[Aguiar 94] Aguiar, M. W. C., Coutts, I., Weston, R., "Model Enactment as a Basis for Rapid Prototyping of Manufacturing Systems", European Workshop on Integrated Manufacturing Systems Engineering, IMSE'94, p. 86-96,Grenoble, France, 1994

[Anciaux 97] Anciauxm D., Roy, D., Vernadat F., "Reactive Shop floor control with Multi-Agent System", Conference on Control of Production and Logistics MCPL'97, Campinas -SP- Brazil, Set., 1997.

[Arabatzis 95] Arabatzis, T., Papaioannou, D., Didic, M., Neuscheler, F., "Elval Pilot Aluminium casting traceability supported by CIMOSA", Computers in Industry, v. 27, p. 191-202, 1995

[Araujo 00] Araujo, O. N., "Relatório de Atividades - Doutorado Sanduíche CAPES - Ferramentas para aplicação em redes de compartilhamento de habilidades no ambiente de manufatura", documento interno, Instituto Nacional de Tecnologia de Informação, 2000

[Aurelio 85] Ferreira, A. B. H.,"Médio Dicionário Aurélio", Ed. Nova Fronteira, 1985

[Barbuceanu 96a] Barbuceanu, M., Fox, M., "Capturing and Modeling Coordination Knowledge for Multi-Agent Systems", Journal of Intelligent and Cooperative Information Systems, jul., 1996

[Barbuceanu 96b] Barbuceanu, M., Fox, M., "The Design of a Coordination Language for Multi-Agent Systems", Intelligent Agents III. Proceedings of the Third International Workshop onAgent Theories, Architectures, and Languages (ATAL'96), Muller J. P., Wooldridge M. J., Jennings N. R., editores, Springer Verlag, ago., 1996

[Baker 74] Baker, K. R., "Introduction to sequencing and scheduling", John Wiley & Sons, 304 p., 1974

[Baker 97] Baker, A., Parunak, H. V. D., Erol, K., "Manufacturing over the Internet and into your living room: perspectives from the AARIA Project", ECECS Dept, Technical Report TR208-08-97, jan., 1997

[Baker 98] Baker, A. D., "A survey of Factory control algorithms which can be implemented in a multi-agent heterarchy: dispatching, scheduling and pull", Journal of Manufacturing Systems, v 17 (4), p. 297-320, SME 1998.

[Balasubramanian 95] Balasubramanian, S., Norrie, D. H., "A Multi-Agent Intelligent Design System Integrating Manufacturing and Shop-Floor Control", First International Conference on Multi-agent Systems, San Francisco, USA, p. 3-9, jun. 1995.

[Bauer 94] Bauer, A., Bowden, R., Browne J., Duggan, J., Lyons, G., "Shop Floor Control Systems, from Design to Implementation", Chapman & Hall, 1994

[Bernus 96] Bernus, P., Nemes, L., Williams, T. J. "Architectures for Enterprise Integration The findings of the IFAC/IFIP Task Force ", 368 p., Chapman & Hall, London, England, 1996

[Bongaerts 98] Bongaerts, L., "Integration of scheduling and control in holonic manufacturing systems", Tese de Doutorado, Katholieke Universiteit Leuven, Leuven, Belgica, dez., 1998

[Bradshaw 97] Bradshaw, J. M., "*An Introduction to Software Agents*", in Software Agents, Bradshaw, J.M. (ed.), Cambridge, MA: MIT Press, 1997

[Brazier 97] Brazier, F. M. T., Dunin-Keplicz, B. M., Jennings, N., Treuer, J., "DESIRE: modelling multi-agent systems in a compositional formal framework", International Journal of Cooperative Information Systems, 6(1), p. 67-94, 1997

[Brill 91] Brill, M., Gramm, U, "MMS: MAP applications services for the manufacturing industry", Computer Networks and ISDN Systems, 21, p. 357-380, 1991

[Briot 98] Briot, J. P., "Agents and concurrent objects - J.P. Briot Interviews Les Gasser", IEEE Concurrency, v. 6, n. 4, p. 74-77, 81, out. / dez., 1998

[Brooks 86] Brooks, R. A, "A Robust layered Control System for a Mobile Robot", *IEEE Journal of Robotics and Automation*, Vol. 2, No. 1, p. 14-23, mar., 1986

[Brooks 89] Brooks, R. A, "A Robot that Walks; Emergent Behaviours from a Carefully Evolved Network", IEEE International Conference on Robotics and Automation, Scottsdale, AZ, p. 292-296, 1989

[Brückner 98] Brückner, S., Wyns, J., Peeters, P., Kollingbaum, M., "Designing Agents for Manufacturing Control", Proceedings of Articial Intelligence and Manufacturing Research Planning Workshop. State of the Art & State of the Practice, SIGMAN '98, Albuquerque, p. 40-46, 1998

[Brussel 95] Brussel, H. V., Valckenaers, P., Bongaerts, L., Wyns, J., "Architectural and system design issues in holonic manufacturing systems", Proceedings of the 3rd IFAC Workshop on Intellingent Manufacturing Systems", Bucharest, 1995 (IMS'95), p. 24-26, nov. 1995

[Brussel 98] Brussel, H. V., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P., "Reference architecture for holonic manufacturing systems PROSA", Computers in Industry 37(3), p. 255-274, 1998

[Burbidge 87] Burbidge, J.L., Falster, P., Riis, J. O., Svendsen, O. M., "Integration in Manufacturing", Computers in Industry, n.9, p. 297-305, 1987

[Chauhan 97] Chauhan, D., "JAFMAS: A Java-based Agent Framework for Multiagent Systems Development and Implementation", PhD Thesis, ECECS Department, University of Cincinnati, 1997

[CIMOSA 93] ESPRIT Consortium AMICE, "Open Systems Architecture for CIM", 2nd edition, Springer -Verlag, Berlin, 1993

[Claudio 00] Claudio, K., "Relatório Final de Estágio - Bolsa Pibic / CNPQ", Documento Interno do Instituto Nacional de Tecnologia de Informação, jul., 2000

[Cloutier 99] Cloutier, L., Frayret, J.M., LeBlond, S., Lei, M., Lyonnais, P., "An agent-based software architecture for networked manufacturing systems", 3rd International Industrial Engineering Conference, p. 85- 94, Montreal, Quebec, Eds. Langevin A.,Riopel D.,Ladet P., mai. 1999

[CNC 91] "Manufacturing Message Specification, Part 4, Companion Standard for Numerical Control", ISO DIS 9506-4, 1991

[Cohen 95] Cohen, P. H., Levesque, H. J., "Communicative Actions for Artificial Agents", Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), AAAI Press, p. 65-72, Menlo Park, California, jun. 1995

[CPM 89] "Programmable Controller Message Specification", International Eletrotechnical Comission - IEC/SC65A/WG6/TF7, Rough Draft, 1989.

[Date 89] Date, C. J., "Guia para o Padrão SQL", Ed. Campus, 1989

[Davenport 93] Davenport, T. H., "Process innovation: reengineering work through information technology", Harvard Business School Press, 1993

[DeLoach 99] DeLoach, S. A., "Multi-agents systems engineering: a methodology and language for designing agent systems", Agent-Oriented Information Systems (AOIS)'99, Seattle USA, may 1999

[Didic 93] Didic, M. M., Neuscheler, F., Bogdanowicz, L., "McCim: Execution of CIMOSA Models", CIM Europe Annual Conference Proceedings, Amsterdan, 1993

[Didic 95] Didic, M. M., Couffin F., Holler, E., Lampériere S., Neuscheler F., Rogier J., de Vries M., "Open engineering and operational environment for CIMOSA", Computers in Industry, v. 27, p. 167-178, 1995

[Dilts 91] Dilts, D. M., Boyd, N.P., Whorms, H.H., "The Evolution of Control Architectures for Automated Manufacturing Systems", Journal of Manufacturing Systems, V. 10, n. 1, 1991

[Doran 96] Doran, J. E., Franklin, S., Jennings, N., Norman, T.J., "On cooperation in multi-agent systems", Firts Uk Workshop on Foundations of Multi-Agent Systems, University of Warwick, out. 1996

[Douglass 98] Douglass, B. P., "Real-Time UML – Developing efficient objects for embedded systems", Addison Wesley, 1998

[Doumeingts 84] Doumeingts, G.,"Méthode GRAI : Méthode de Conception des Systèmes en Productique". Thèse de doctorat de l'Université de Bordeaux I, Bordeaux, France, 13 nov. 1984

[Doumeingts 87] Doumeingts, G., Vallespir, B., Darricau, D., Roboam, M.,"Design Methodology for Advanced Manufacturing Systems", Computers in Industry n.9, p. 271-296, 1987

[Doumeingts 95] Doumeingts, G., Vallespir, B., Chen, D.,"Methodologies for designing CIM systems:A survey", Computers in Industry n.25, p. 263-280, 1995]

[Duffie 94] Duffie, N. A., Prabhu, V. V., "Real-time Distributed Scheduling of Heterarchical Manufacturing Systems", Journal of Manufacturing Systems, v. 13, n. 2, p. 94-107, 1994

[Duffie 96] Duffie, N. A., Prabhu, V. V., "Heterarchical control of highly distributed manufacturing systems", International Journal Computer Integrated Manufacturing, v. 9, n. 4, p. 270-281, 1996

[Fadil 97] Fadil, A., Chetouane, F., Binder, Z., "Control of Flexible Job-Shop Disturbances and Robustness", Conference on Management and Controlk of Production and Logistics, MCPL97, Campinas, Brazil, set. 1997

[Finim 93] Finim, T., Weber, J., "Specification of the KQML Agent- Communication Language", Draft, jun., 1993

[Flanagan 97] Flanagan, D., "Java in a Nutshell", updated for jdk 1 .1, O'Reilly & Associates , out., 1997

[Franlin 96] Franklin, S., Graesser, A., "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents", Proceedings of the Thir International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996

[Fox 92] Fox, M., "The TOVE Project: Towards a Common-Sense Model of the Enterprise", Enterprise integration modeling: proceedings of the first international conference, p. 310, 319, Charles Petrie Jr. Editor, MIT Press, 1992

[Gaines 88] Gaines, B. R., "Structure, Development and Applications of Expert System in Integrated Manufacturing", in Artificial Intelligence: Implications for CIM, Kusiak A. (Editor), p. 117-161, Springer Verlag, 1988

[Gasser 00] Gasser, L., "MAS infrastructure definitions, needs, and prospects", Proceedings of the Workshop on Scalable MAS Infrastructure, Barcelona, Spain, jun., 2000

[Genereseth 94] Genereseth, M. R., Ketchpel, S. P., "Sotware Agents", Communications of the ACM, v. 37, n.7, p. 100-105, 1994

[Göddertz 90] Göddertz, J., "Profibus", Klocner Moeller, Bonn, 1990.

[Gruber 93] Gruber, T. R., "A Translation Approach to Portable Ontology Specifications", Knowledge Systems Laboratory, Technical Report KSL 92-71, set. 92, revisado abr., 1993

[Hammer 94] Hammer, M., Champy, J., "Reengineering the corporation", HarperCollins Publishers, 1994

[Harel 87] Harel, D., "Statecharts: A visual Formalism for Complex Systems", Science of Computer Programming 8, p. 231-274, North-Holland, 1987

[Hatley 88] Hatley, D. J., Pirbhai, I., "Strategies for Real-Time System Specification", Dorset House, 1988

[Heragu 87] Heragu, S. S., Kusiak, A., "Analysis of expert systems in manufacturing design", IEEE Transactions on Systems, Man, and Cybernetics, v. SMC-17, n. 6, p. 898-912, nov./dec, 1987

[Hoekstra 90] Hoekstra, J., "The Manufacturing Message Specification", Open Systems Information Services, v. 2, n. 8, 1990.

[Howard 99] Howard, A., Kochhar, A., Dilworth, J., "Application of generic manufacturin planning and control system reference architecture to different manufacturing environments", Proc. Instn Mech Engs, v.213, part B, p. 381-398, 1999

[Horling 98] Horling, B. C.,"A reusable component architecture for agent construction", Department of Computer Science, University of Massachusetts, Umass Computer Science Technical Report 1998-49, out., 1998

[Huguet 95] Huguet, P., Grabot, B., "A conceptual framework for shopfloor production activity control", International Journal Computer Integrated Manufacturing, v. 8, n. 5, p. 357-369, 1995

[Imai 86] Imai, M., "Kaizen, the key to Japan's competitive success", McGraw-Hill Publishing Co., 1986

[ISO 97b] ISO International Organization for Standardization - TC 184 SC5 WG1 , Modeling and Architecture, "Mapping of GIM onto GERAM requirements", N 378, fev., 1997

[ISO 98] ISO International Organization for Standardization - TC 184 SC5 WG1 ,"ISO/DIS 15704 - Industrial automation systems - Requirements for enterprise-reference architectures and methodologies", 1998

[JATLite 97] JATLite, "http:java.stanford.edu"

[Jeon 00] Jeon, H., Petrie, C., Cutkostky, M. R., "JATLite: A Java Agent Infrastructure with Message Routing", IEEE Internet Computing, mar. / abr., 2000.

[Jennings 98] Jennings, N. R., Sycara K., Wooldridge M. , "A Roadmap of Agent Reseach and Development", Autonomous Agents and Multi-Agents Systems, v.1, p. 7-38, Kluwer Academics Publisher, Boston, USA, 1998

[Kusiak 88] Kusiak, A. (Editor), "Artificial Intelligence: Implications for CIM", 527 p., Springer Verlag, 1988

[Lapalus 95] Lapalus, E., Fang S. G., Rang C., Gerwen R. J., "Manufacturing Integration", Computers in Industry, v. 27, p. 155-165, 1995

[Lesperance 96] Lesperance, Y., Levesque H. J., Lin F., Marcu D., Reiter R., Scherl R. B., "Foundations of a logical approach to agent programming", Inelligent Agents volume II - Proceedings of the 19995 Workshop on Agent Theories, Architectures, and Languages (ATAL -95), p. 331-346, Springer Verlag, Lecture Notes in Artificial Intelligence, 1996

[Lesser 99] Lesser, V., Atighetchi, M., Benyo, B., Horling, B., Raja, A., Vincent, R., Wagner, T., Xuan, P., Zang, S. X. Q., "A multi-agent system for intelligent environment control", Proceedings of the third international conference on autonomous agents, Seattle WA USA, 1999

[Lin 91] Lin, G. Y., Solberg, J. J., "Effectiveness of flexible routing control", The International Journal of Flexible Manufacturing Systems, n. 3, p. 189-211, 1991

[Lyonnais 99] Lyonnais, P., Montreuil, B., Lefrançois, P., "Agent-oriented distributed architecture for simulation of networked manufacturing systems", 3rd International Industrial Engineering Conference, Montreal, Quebec, Eds. Langevin A.,Riopel D.,Ladet P., mai. 1999

[MAP 88] "Manufacturing Automation Protocol", Ver. 3.0, 1988.

[McLean 83] McLean, C., Mitchell, M. ,Barkmeyer, E., "A computer architecture for small-batch manufacturing", IEEE Spectrum, v. 20, n. 5, 1983

[MMS 90] Manufacturing Message Specification, ISO 9506, 1990

[Müller 98] Müller, J. P., Parunak, H. V. D., "Multi-agent Systems and Manufacturing", 9th Symposium on information control in manufacturing, INCOM'98, Nancy-Metz, France, p. 165-170, 1998

[Newman 88] Newman, P. A.,"Scheduling in CIM Systems", in Artificial Intelligence: Implications for CIM, Kusiak, A. (Editor), p. 361-402, Springer Verlag, 1988

[Nwana 96] Nwana, H. S., "Software Agents: an overview", Knowledge Engineering Review, V. 11, n. 3, p. 205-244, out. / nov., 1996

[Orfali 98] Orfali, R., Harkey, D., "Client / Server Programming with JAVA and CORBA", 2nd edition, John Wiley & Sons, 1022 p., 1998

[Oshima 98] Oshima, M., Karjoth, G., Ono, K., Aglets Specification 1.1. Draft 0.65, IBM Corp. Japan, "http:www.trl.ibm.co.jp/aglets/spec11.html", 1998

[Parunak 88] Parunak, H. V. D., "Distributed Artificial Intelligence Systems", in Artificial Intelligence: Implications for CIM, Kusiak, A. (Editor), p. 225-251, Springer Verlag, 1988

[Parunak 97] Parunak, H. V. D., Baker, A., Clark, S. J.," The AARIA agent architecture: an example of requirements-driven agent-based system design", Agents'97, Marina Del Rey CA USA, ACM, 1997

[Parunak 98] Parunak, H. D., "Practical and Industrial applications of agent-based systems", http://www.erim.org/~van/apps98.pdf, 1998

[Peng 98] Peng, Y., Finin, T., Labrou, Y., Chu, B., Long, J., Tolone, W. J., Boughannam, A., "A Multi-Agent System for Enterprise Integration", Journal of Applied Artificial Intelligence, 1998

[Petrie 96] Petrie, C., "Agent-based engineering, the web, and intelligence", IEEE Expert, dez,. 1996

[Pirlot 96] Pirlot, M., "General Local Search Methods", European Journal of Operational Research, V. 92, pp 493-511, 1996

[Plossl 87] Plossl, K. R., "Computer Integrated Manufacturing", Production Engineering, p. 38-50, jun, 1987

[prENV 99] European Comittee for Standardization, "Advanced manufacturing technology - systems architecture - enterprise model execution and integration services", final draft prENV 13550, Brussels, 1999

[prENV 95] European Comittee for Standardization, "Advanced manufacturing technology - systems architecture - Constructs for Enterprise Modelling", prENV 12204, Brussels, 1995

[Pressman 92] Pressman, R. S., "Software Engineering: Practitionner's Approach", terceira edição européia, McGraw-Hill, 1992

[Querenet 92] Querenet, B., "CIMOSA - A European Development for Enterprise Integration, Part III: Enterprise Integration Infrastructure", Enterprise integration modeling: proceedings of the first international conference, p. 205-215, Charles Petrie Jr. Editor, MIT Press, 1992

[Rabelo 94a] Rabelo, R. J., Camarinha-Matos, L.M., "Generation of multiagent infrastructures for dynamic scheduling and control architectures", 27th ISATA - Lean/ Agile Manufacturing in the Automotive Industries, Aachen, Germany, 1994

[Rabelo 95b] Rabelo, R. J., Camarinha-Matos, L.M., "HOLOS: a methodology for deriving scheduling systems", Basys'95 - IEEE/ECLA/IFIP International Conference on Architectures and Design Methods for Balanced Automation Systems, Vitória, Brazil, 1995

[Rao 95] Rao, A., Georgeff, M. P., "BDI Agents: from theory to practice", Proceedings of the First International Conference on Multi-Agents (ICMAS 95), San Francisco, USA, jun., 1995

[ROB 90] "Manufacturig Message Specification, Part 3, Robot Specific Message System", ISO DIS 9506-3, 1990.

[Rolstadas 94] Rolstadas, A., "Beyond year 2000 - production management in the virtual company", Production Management Methods (B-19), C. Walter, F. J. Klieman and J. P. M. de Oliveira (Editors), p. 3-10, IFIP 94, Elsevier, 1994

[Rondeau 95] Rondeau, E., Lepage, F., Veron, M., "MMS virtual manufacturing devices generation,: the paris subway example", Integrated Manufacturing Systems Engineering, Part three : Enterprise Engineering, P. Ladet, F.B. Vernadat (eds), IFIP TC5, Chapman & Hall, ISBN 0-412-72680-7, p. 84-98, out., 1995

[Rosário 96] Rosário, J . M ., Frachet, J P, "Méthodes et Outils en Génie Automatique et Productique pour le Développement de la Qualité et de la Productivité dans le PME/PMI", Project de Recherche CAPES COFECUB n . 195/96, 1996

[Roy 98] Roy, D, "Une architecture hierarchisée multi-agents pour le pilotage reactif d´atelier de production", Thèse de doctorat de l´Université de Metz, Metz, France, jan., 1998.

x[Santos 00] Santos, J. P. O, Ferreira J. J. P., Mendonça, J. M., "A modelling language for the design & execution of enterprise models, in manufacturing", International Journal of Computer Integrated Manufacturing, vol 13 – 1, jan. 2000

[Savage 96] Savage, C., "5th generation management: co-creating through virtual enterprise, dynamic teaming, and knowledge networking", edição revisada, Butterworth-Heinemann, 1996

[Scheer 92] Scheer, A.W., "Architecture of Integrated Information Systems, Foundations of Enterprise Modelling", Springer-Verlag, 1992.

[Scheer 94] Scheer,A.W., "Business process engineering: reference models for industrial enterprises", Springer-Verlag, 1994

[Senge 94] Senge, P. M., "The fifth discipline: the art & practice of the learning organization", Currency Doubleday, 1994

[Sethi 90] Sethi, A. K., Sethi, S. P., "Flexibility in Manufacturing: A Survey", The International Journal of Flexible Manufacturing Systems, n. 2, p. 289-328, 1990

[Shen 1999] Shen, W., N. D. H., "Agent-Based Systems for Intelligent Manufacturing: A State-of-the -Art Survey", Knowledge and Information Systems, an International Journal, 1(2), p. 129-156, 1999

[Smith 80] Smith, R. G., "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver", IEEE Transactions on Computers, v. C-29, n. 12, dez., 1980

[Smith 95] Smith, J., Joshi, S. B., "A shop floor controller class for compute-integrated manufacturing", International Journal of Computer Integrated Manufacturing, v. 8, n. 5, p. 327-339, 1995

[Spinosa 97] Spinosa, L. M., Espinasse, B., Chouraqui, E.,"For a Decision Support System model to Distributed Manufacturing Systems: a multiagent and CIMOSA based approach", IFAC/ IFIP Conference on Management and Control of Production and Logistics, MCPL'97, Campinas, Brazil, 1997

[Suraj 97] Suraj, A., Ramaswamy, S., Barber, K. S., "Extended StateCharts for the modelling and specification of manufacturing control software systems", International Journal of Computer Integrated Manufacturing, 1997, v. 10, n. 1-4, pp 160-171, Taylor & Francis, 1997

[Sycara 96] Sycara, K. et al., "Distributed Intelligent Agents", IEEE Expert, dez., 1996

[Tenenbaum 92] Tenenbaum, J. M., Weber J. C., "Enterprise Integration: lessons from SHADE and PACT", Enterprise integration modeling: proceedings of the first international conference, p. 356-369, Charles Petrie Jr. Editor, MIT Press, 1992

[Thierauf 82] Thierauf, R. J., "Decision support systems for effective planning and control", Prentice-Hall, 1982,

[Toffler 85] Toffler, A., "A Empresa Flexível", Editora Record, 1985

[Uschold 98] Uschold, M., King, M., Moralee, S., Zorgios, Y., "The Enterprise Ontology", The Knowledge Engineering review, v. 13, special issue on Putting Ontologies to use (eds. Mike Uschold and Austin Tate), 1998

[Vernadat 96] Vernadat, F., "Enterprise Modelling and Integration: Principles and Application", Chapman & Hall, London, 513 p., 1996

[Vernadat 97a] Vernadat, F. B., "An enterprise integration framework for manufacturing environments", Computer Applications in Production and Engineering, F. Plonka and G. Olling (Eds), IFIP, Chapman & Hall, 1997

[Ward 85] Ward, P. ,Mellor, S., "Structured Development for Real Time Systems", 4 vols. Englewoods Cliffs, NJ: Prentice Hall, 1985

[Williams 92] Williams, T. J., "The Purdue Enterprise Reference Architecture", Research Triangle Park, North Carolina: Instrument Society of America, USA, IN,, 235 p., 1992

[Williams 96] Williams, T. J., Rathwell, G. R., Li, H., "A Handbook on Master Planning and Implementation for Enterprise Integration Programs", Purdue Laboratory for Applied Industrial Control, Purdue University, IN, USA, 1996. 375p. (Relatório Técnico).

[Wooldridge 92] Wooldridge, M. J., "The Logical Modelling of Computational Multi-agent Systems", PhD. Thesis, University of Manchester, ago., 1992

[Wooldridge 94] Wooldridge, M, Jennings, N., "Intelligent Agents: Theory and Practice", submitted to Knowledge Engineering Review, 1994

[Wooldridge 99] Wooldridge, M, Jennings, N., Kinny, N., "A methodology for agent-oriented analysis and design", Proceedings of the third anual Conference on Autonomous Agents, p. 69-76, Seattle , WA, USA, mai., 1999

[Yang 85] Yang, J.D., Huhns, M. N., Stephens, L. M., "An architecture for control and communication in distributed artificial systems", IEEE Transactions on Systems, Man, and Cybernetics, v. SMC-15, n. 1, p. 316-326, mai. / jun., 1985

Anexo I - Detalhes de Implementação

Para a geração deste anexo foi utilizada a documentação técnica do projeto, a qual consta de aproximadamente 150 páginas. Foram selecionados diagramas e exemplos relevantes, os quais ilustram o uso da metodologia desenvolvida.

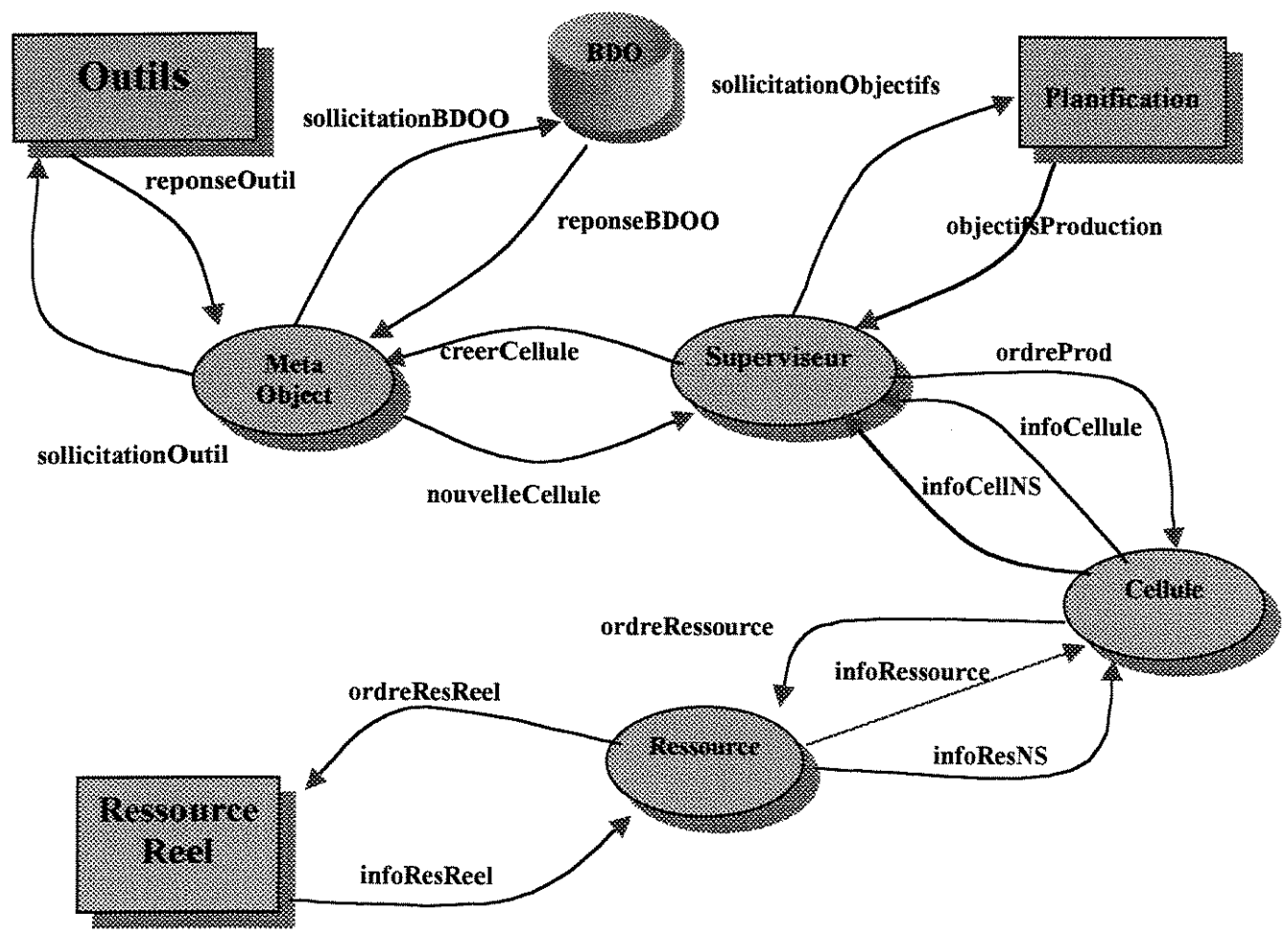


Figura A.1: Diagrama Geral de Syroco-2

Devido à colaboração internacional com o LGIPM da Universidade de Metz, optou-se por utilizar o francês para nomear as variáveis e fluxos. A Figura A.1 é uma repetição da Figura 4.4, com a utilização dos termos em francês, como consta da documentação técnica. Ela foi repetida aqui para facilidade de leitura deste anexo.

A partir do diagrama DFD proposto para o sistema , Figura A.1, são detalhadas as mensagens e sua composição:

sollicitationOutil
Description: demande l'exécution d'un outil externe.

reponseOutil
Description: données génères par l'outil

sollicitationBDOO
Description: accès à la base de données orienté objets (pour définir)

Items: nombrePhases, flux, nomProduit, nomRessource, numeroProduit, phaseGamme, posXCG, posYCG, tempsPhase, typeRessource, duree, nombreEntrees, tempsDepart;

reponseBDOO
Description: actualisation de la base de données orienté objets (pour définir)

Items: duree, nombreEntrees, nombrePhases, nomProduit, nomRessource, phaseGamme, posXCG, posYCG, tempsDepart, typeRessource;

Internal Structures:
Entity Name: Agenda
Description:
Primary Identifier:nomRessource;
Non-Primary Attributes:duree, nombreEntrees, nomProduit, tempsDepart;

Entity Name: Atelier
Description: enregistrement des ressources avec CG et type
Primary Identifier:nomRessource;
Non-Primary Attributes:posXCG, posYCG, typeRessource;

Entity Name: Solution
Description: liste des gammes techniques des en cours
Primary Identifier:nomProduit;
Non-Primary Attributes:nombrePhases, phaseGamme;

creerCellule
Description: Demande la création d'une cellule virtuelle pour produire le produit associé au objectif de production traité.

nouvelleCellule
Description: Reconnaît création de la cellule virtuelle

objectifsProduction
Description: liste de objectifs de production (*InnerObProd)

Items: dateLimite, dateOptimale, nombre, nomProduit, priorite;

Internal Structures:
Entity Name: InnerObProd
Description: variables internes de chaque obj. production
Primary Identifier:nomProduit;
Non-Primary Attributes:dateLimite, dateOptimale, nombre, priorite;

sollicitationObjectifs
Description: demande les objectifs de production à la Planification

infoCellule, infoCellNS
Description: information de fin de phase et avancement de la production.

Items: phaseGamme, commentaire, date, datedebut, duree, nomProduit, nomRessource, tempsPhase, numeroPhase, tempsRestant;

Internal Structures:
Entity Name: Panne
Description: variables pour le message Panne
Primary Identifier:nomProduit;
Non-Primary Attributes:commentaire, datedebut, duree, nomRessource, tempsRestant;

Entity Name: FinPhase
Description: variables pour le message FinPhase
Primary Identifier:nomProduit;
Non-Primary Attributes:date, nomRessource, numeroPhase;

Entity Name: Maintenance
Description: variables pour le message Maintenance
Primary Identifier:nomProduit;
Non-Primary Attributes:commentaire, datedebut, duree, nomRessource;

Entity Name: Commentaire
Description: variable pour le message Commentaire
Primary Identifier:None;
Non-Primary Attributes:commentaire;

ordreProd
Description: Ordre de Production, composé des actions de Initier, Modifier et Consulter l'ordre de production courant.

Items: nomProduit, numeroPhase;

Internal Structures:
Entity Name: Modifier
Description: variable pour le message Modifier
Primary Identifier:nomProduit;
Non-Primary Attributes:None;

Entity Name: Consulter
Description: variable pour le message Consulter
Primary Identifier:nomProduit;
Non-Primary Attributes:numeroPhase;

Entity Name: Initier
Description: variable pour le message Initier
Primary Identifier:nomProduit;
Non-Primary Attributes:None;

infoRessources, infoResNS
Description: Informations sur les ressources réelles

Items: etatRessources, nomProduit, nomRessource, datedebut, duree, commentaire, numeroPhase, date, tempsPhase;

Internal Structures:
Entity Name: Panne
Description: variable pour le message Panne
Primary Identifier:nomProduit;
Non-Primary Attributes:commentaire, datedebut, duree, nomRessource;

Entity Name: FinPhase
Description: variables pour le message FinPhase
Primary Identifier:nomProduit;
Non-Primary Attributes:date, nomRessource, numeroPhase;

Entity Name: Maintenance
Description: variables pour le message Maintenance
Primary Identifier:nomProduit;
Non-Primary Attributes:commentaire, datedebut, duree, nomRessource;

Entity Name: Commentaire
Description: variable pour le message commentaire
Primary Identifier:None;
Non-Primary Attributes:commentaire;

Entity Name: Reponse
Description: variables pour le message Reponse
Primary Identifier:nomRessource;
Non-Primary Attributes:tempsPhase;

ordreRessource
Description: Commande de fabrication envoyé au ressource.

Items: duree, nomProduit, nomRessource, numeroPhase, ressourcePrecedente;

Internal Structures: None;

ordreResReel
Description: Ordres de fabrication pour les ressources réelles

Items: duree, nomProduit, nomRessource, numeroPhase, ressourceInformatique, ressourcePrecedente;

infoResReel
Description: informations sur les ressources réelles

Items: etatRessources, etatGamme, infoOperations, nomProduit, nomRessource, duree, datedebut, numeroPhase, date, tempsPhase, commentaire;

Internal Structures:
Entity Name: Panne
Description: variables pour le message panne
Primary Identifier:nomProduit;
Non-Primary Attributes:commentaire, datedebut, duree, nomRessource;

Entity Name: FinPhase
Description: variables pour le message FinPhase
Primary Identifier:nomProduit;
Non-Primary Attributes:date, nomRessource, numeroPhase;

Entity Name: Maintenance
Description: variables pour le message Maintenance
Primary Identifier:nomProduit;
Non-Primary Attributes:commentaire, datedebut, duree, nomRessource;

Entity Name: Commentaire
Description: variable pour le message Commentaire
Primary Identifier:None;
Non-Primary Attributes:commentaire;

Os agentes (MetaObjeto, Supervisor, Célula e Recurso), que são representados pelo DFD da Figura A.1, podem interagir de diferentes maneiras para atingir o objetivo deste Sistema Multi-agentes, que é supervisionar e controlar a produção dos objetivos de produção fornecidos pelo Planejamento da Produção.

Para descrever estas diversas possibilidades (ou cenários) são utilizados os diagramas de seqüência da linguagem UML. Estes diagramas são gerados considerando-se as mensagens que são explicitadas no DFD. O uso simultâneo do DFD e dos diagramas de seqüência permite explicitar a dinâmica do sistema.

A seguir, Figura A.2, é apresentado um exemplo de cenário de falha de um dispositivo, durante a produção de um certo produto que faz parte de uma ordem de produção em andamento.

1. Recurso em falha sinaliza sua situação à Célula através de uma mensagem não solicitada de informação (isto significa que a Célula não precisa ficar monitorando o Recurso, através de seguidas consultas ao seu estado).

2. Célula informa a ocorrência da pane ao Supervisor, também através de uma mensagem de informação não solicitada. Ela fica em um estado de EXCEÇÃO (anteriormente estava em um estado de PRODUÇÃO).

3. Supervisor sabe que será necessário reescalonar a produção. Para isto ele solicita ao MetaObjeto a criação de uma nova Célula, através da mensagem creerCellule.

4. MetaObjeto irá utilizar uma inteligência externa para fazer o escalonamento, através da requisição de uma ferramenta externa de escalonamento, mensagem sollicitationOutil

5. Módulo de ferramentas externas irá realizar o escalonamento, com o algoritmo que estiver programado no momento e irá devolver o resultado ao MetaObjeto, mensagem reponseOutil.

6. MetaObjeto irá organizar as informações recebidas na forma de uma configuração de célula virtual de produção, que será enviada ao Supervisor, mensagem nouvelleCellule.

7. Supervisor irá analisar a configuração recebida, no caso ele verifica que a Célula que está em exceção pode continuar a trabalhar com um novo recurso equivalente, o qual foi atribuído no

novo escalonamento. Ele envia então um comando de produção, com parâmetro de modificar, para a Célula, mensagem ordreProd.

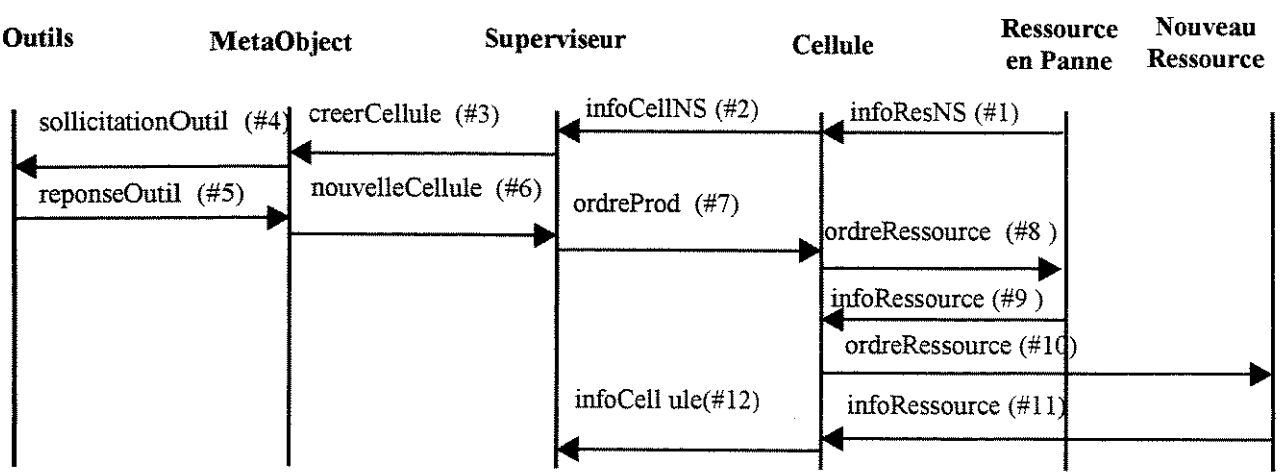
8. Célula envia uma ordem para o Recurso em falha, determinando que ele se termine, mensagem ordreRessource

9. Recurso responde com uma mensagem de informação, confirmando seu término, mensagem infoRessource.

10. A Célula que estava no estado de EXCEÇÃO passa para o estado de PRODUÇÃO e continua a produzir a partir do ponto onde estava, através de uma mensagem ordreRessource enviada ao novo Recurso atribuído.

11. Recurso responde que está executando a produção através de uma mensagem infoRessource.

12. Célula informa ao Supervisor que a produção continua.



#1 = failure, failResource
#2 = failure, failResource, failProduct
#3 = failProduct, nf, tf
#4 = search, (Agenda, Shop, ProcPlan)
#5= +, Solution
#6 = +, m1, m3, m4, m11, cellObject
#7 = modify, failProduct, list of resources
#8 = stop
#9 = +, resState
#10 = duration, failProduct, newResource, phaseNumr, precResource
#11 = +, resState
#12 = +, phaseNumber

Figura A.2: Cenário de Falha de Equipamento

A Figura A.3 mostra as classes básicas que foram utilizadas para o agente, este é um diagrama de classes UML.

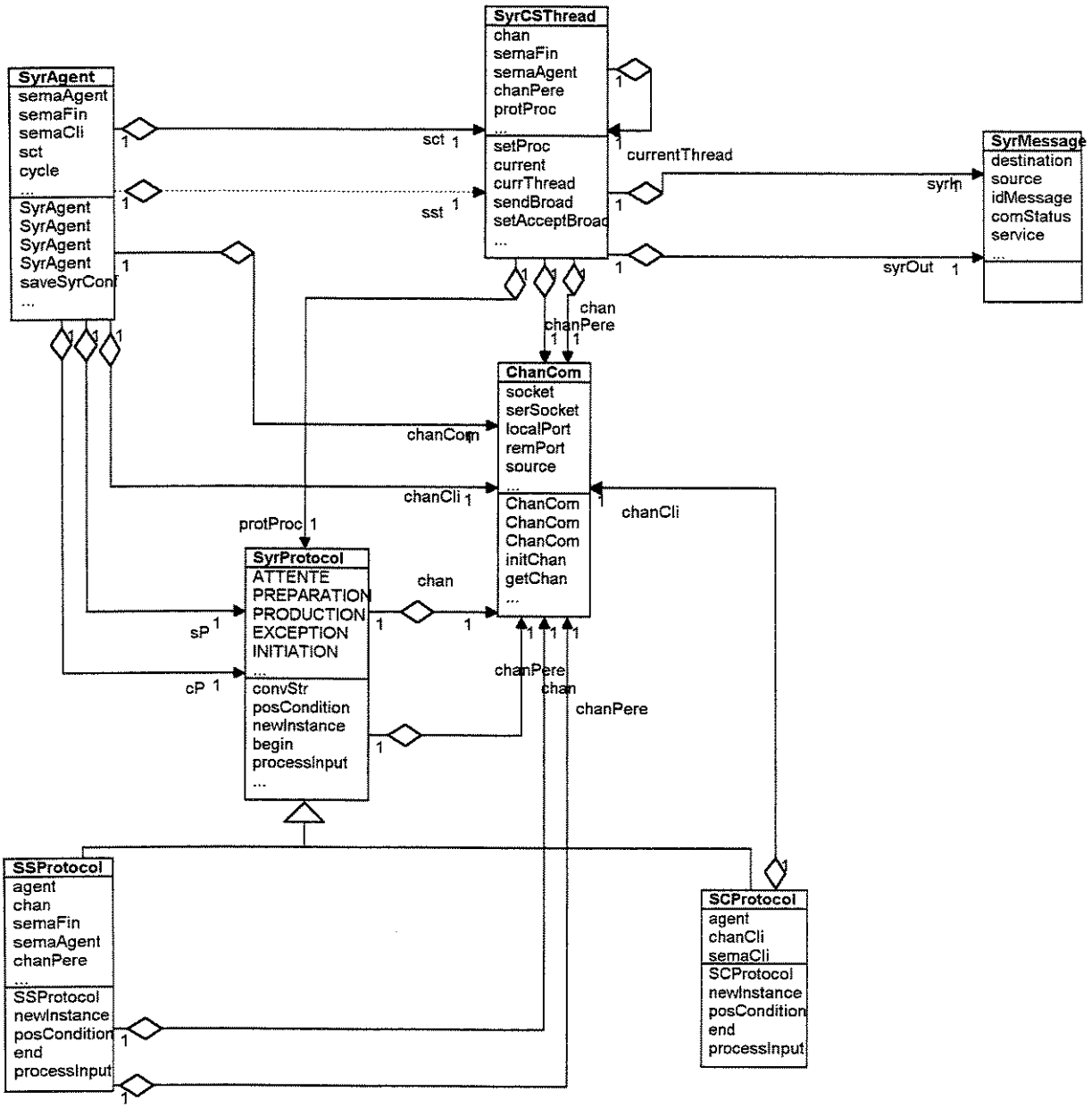


Figura A.3: Diagrama de Classes de Syroco-2

O Componente Genérico foi desenvolvido como um Agente Informático. Sua codificação foi feita usando técnicas de orientação a objetos, visando facilitar sua reutilização e especialização para compor os diversos agentes que formam a Arquitetura Experimental e a Arquitetura Syroco-2.

O agente é representado pela classe SyrAgent e será este agente que, através de mecanismos de herança, dará origem a todos os outros que compõem Syroco-2.

O Componente Genérico pode possuir vários Clientes e Servidores os quais são modelados através da classe **SyrCSThread**. Esta classe depende da troca de mensagens (entre Clientes e Servidores), representadas pela classe **SyrMessage**, através de um canal de comunicação, representado pela classe **ChanCom**.

A troca de mensagens é baseada em um protocolo de comunicação. Na implementação, o que caracteriza a diferença entre os agentes é justamente o protocolo utilizado. Este protocolo, **SyrProtocol**, é dividido em duas partes, uma cliente, classe **SCProtocol** e uma servidora, **classe SSProtocol**.

Estas classes são obtidas através de herança da classe SyrProtocol. SyrProtocol serve como classe *container* para permitir instanciação dos diferentes agentes, uma vez que cada um deles irá possuir diferentes protocolos, que podem ser representados pela superclasse SyrProtocol.

Os agentes que compõem Syroco-2 são obtidos através de herança do agente SyrAgent. Eles possuem protocolos Cliente e Servidor, que são obtidos através de herança das classes SSProtocol (Servidor) e SCProtocol (Cliente), é necessária uma especialização das classes para implementar as diferentes máquinas de estado associadas a cada protocolo.

Mostrar-se-á, na Figura A.4, um rastreamento das mensagens trocadas para um cenário específico de operação. Aparecem em destaque os estados internos dos agentes (referentes aos protocolos utilizados) e os detalhes de nomenclatura de cada protocolo gerado por herança.

Na coluna da esquerda encontra-se um agente Célula, derivado da classe SyrAgent e pertencente à classe **SyrCell**, na coluna da direita está um agente Recurso, classe **SyrRes**.

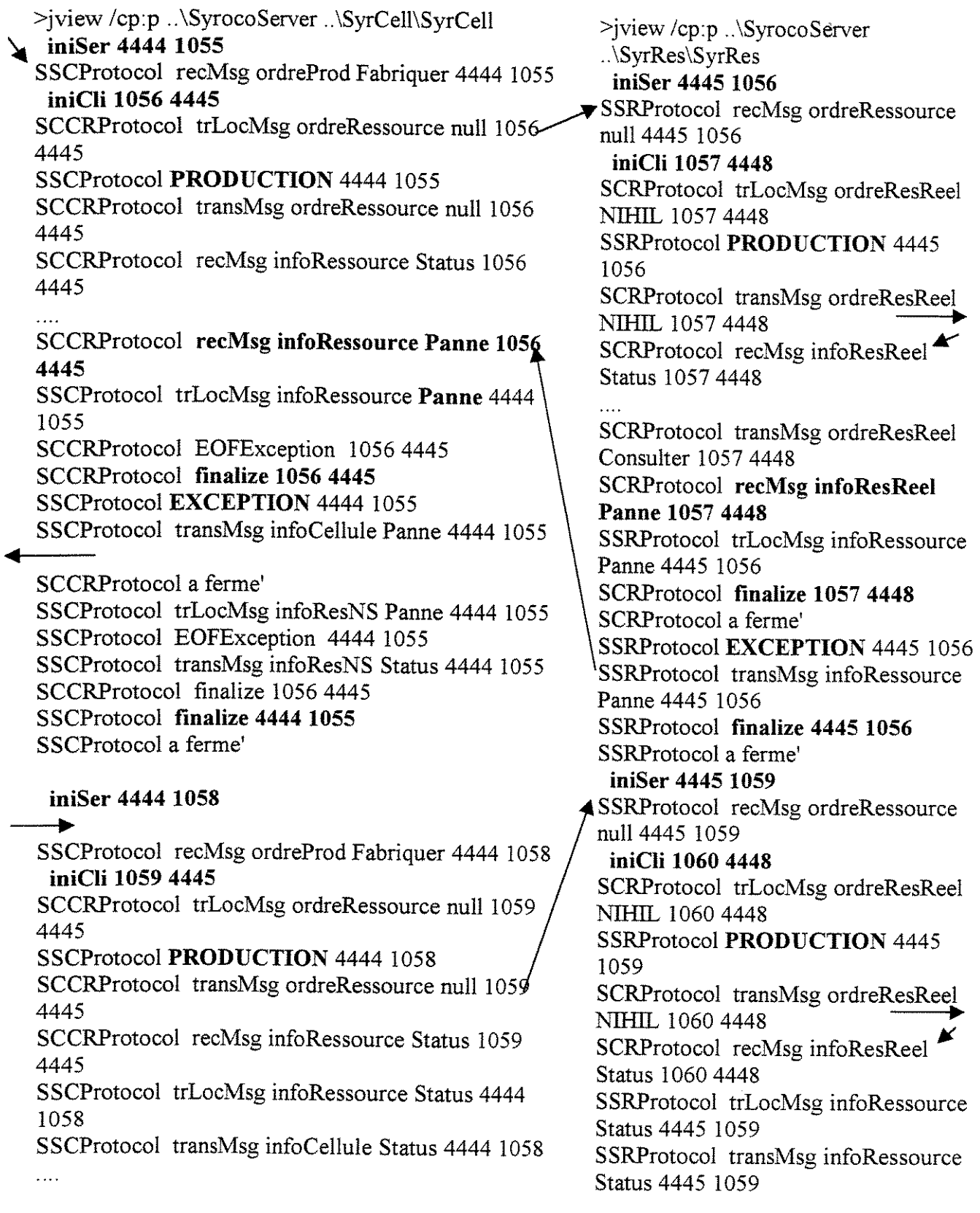


Figura A.4: Rastreamento de mensagens

Na Célula a chegada de uma mensagem ordreProd (Fabriquer) ativa um Servidor (caracterizado pelo protocolo SSCProtocol), identificado pelas portas 4444 1055, este determina a criação de um Cliente (1056 4445, protocolo SCCRProtocol) para comunicar-se com o agente Recurso e entra no estado de **PRODUÇÃO**.

No lado do Recurso, a chegada de uma mensagem ordreRessource, enviada do Cliente da Célula, ativa a criação de um Servidor do Recurso (SSRProtocol, portas 4445 1056), este cria um Cliente do Recurso (SCRProtocol, 1057 4448) para comunicar-se com o Recurso Real.

O Recurso envia mensagens ordreResReel ao Recurso Real e recebe respostas infoResReel. Enquanto não houver problemas estas mensagens são repassadas à Célula e o Recurso continua no estado de PRODUÇÃO. Quando ocorre um problema, no caso a chegada de uma mensagem infoResReel com a indicação de pane, o Cliente do Recurso que se comunicava com o Recurso Real é fechado. O Servidor do Recurso envia uma mensagem infoRessource para a Célula e entra no estado de EXCEÇÃO fechando-se a seguir. Como este Recurso está em falha não poderá continuar a ser utilizado.

O Cliente da Célula recebe a mensagem infoRessource e percebe o problema, tanto pela indicação do parâmetro pane como também por perceber a queda da conexão com o Recurso (que se terminou, causando uma falha na rede). O Servidor da Célula é informado pelo seu Cliente e entra no estado de EXCEÇÃO, transmitindo a mensagem infoCellule com uma indicação de pane ao Supervisor. O Cliente da Célula é terminado, pois o Recurso ao qual estava conectado já não existe. O Servidor da Célula foi também terminado (neste caso esta Célula específica não aceita a possibilidade de modificar a produção).

O Supervisor (não indicado no rastreamento), demanda o escalonamento da produção. Após receber a resposta do MetaObjeto irá criar uma nova Célula para continuar a produção. Esta nova Célula foi ativada na mesma máquina na qual se encontrava a outra e é criada a partir da chegada da mensagem ordreProd (Fabriquer), com as portas 4444 1058.

A nova Célula ativa um Cliente (SCCRProtocol 1059 4445), para se comunicar com o novo Recurso, o qual será criado na mesma máquina onde se encontrava o anterior. O Servidor da Célula entra então no estado de PRODUÇÃO, continuando a produzir (possivelmente a partir do

ponto onde estava anteriormente, isto depende das decisões da ferramenta externa de escalonamento).

A troca de mensagens continua normalmente até o fim da produção, quando os agentes envolvidos são desabilitados.

UNICAMP
BIBLIOTECA CENTRAL
SEÇÃO CIRCULANTE