

**UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE
UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO**

LUIZ CARLOS AIRES DE MACÊDO

**DESENVOLVIMENTO DE SISTEMAS INDUSTRIAIS COM AGENTES:
UMA PROPOSTA DE EXTENSÃO PARA A METODOLOGIA
PROMETHEUS**

MOSSORÓ,

2010

LUIZ CARLOS AIRES DE MACÊDO

**DESENVOLVIMENTO DE SISTEMAS INDUSTRIAIS COM AGENTES:
UMA PROPOSTA DE EXTENSÃO PARA A METODOLOGIA
PROMETHEUS**

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade do Estado do Rio Grande do Norte e Universidade Federal Rural do Semi-Árido como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências da Computação.

Área de concentração: Sistemas Computacionais

Orientador: Prof. Dr. Pedro Fernandes Ribeiro Neto
Co-Orientador: Prof. Dr. Francisco Milton Mendes Neto

**MOSSORÓ,
2010**

**DESENVOLVIMENTO DE SISTEMAS INDUSTRIAIS COM AGENTES: UMA
PROPOSTA DE EXTENSÃO PARA A METODOLOGIA PROMETHEUS**

FOLHA DE APROVAÇÃO

Luiz Carlos Aires de Macêdo

Dissertação de mestrado apresentada em Maio de 2010

Pedro Fernandes Ribeiro Neto, Dr.

Orientador

Francisco Milton Mendes Neto, Dr.

Orientador

Marcelino Pereira dos Santos Silva, Dr.

Componente da Banca

Componente da Banca

Mossoró, Rio Grande do Norte, Brasil

Aos meus pais e familiares que sempre
me apoiaram!

AGRADECIMENTOS

Não há como deixar de agradecer primeiramente a meus mestres orientadores, na realidade doutores, os professores Pedro Fernandes e Milton Mendes, por me guiarem até este ponto, serei sempre grato a vocês por este trabalho.

A todos os professores do Programa de Pós-Graduação em Ciência a Computação UERN/UFERSA: Marcelino Pereira, Lyrene Fernandes, Iguatemi Fonseca, Cláudia Fernandes e João de Deus, que se dedicam a este programa com muita responsabilidade. Também não esquecendo as secretárias do mestrado, Rosita e Emanuela, que cuidam de todos nós com muito carinho e atenção.

A USIBRAS, na figura do diretor Guilherme Assis e os gerentes Evandro Nepomuceno e Ueber Fontinele, por acreditarem em meu trabalho e abrir as portas desta empresa para que eu aprendesse na prática o significado de um sistema industrial.

A todos os meus companheiros neste programa, alunos com quem compartilhei momentos de decepções e alegrias, são eles: Alexandre, Alexsandra, Carlos Evandro, Cassimiro, Ceres, Diego, Edsongley, Luana, Max, Perla, Ticiane e Sebastião, vocês também são parte deste trabalho!

Aos meus companheiros de momentos de descontração (farras): Bento, Baiano, Cláudio, Evandro (de novo!) e Walberty que compreenderam minha luta e deixaram de me ligar nesses últimos três meses, facilitando o término deste trabalho. Acredito que já podem me ligar de novo pessoal!

Aos meus companheiros do Laboratório de Engenharia de Software (LES): Gracon, Jocileide, Lenardo, Luana, Luiz Cláudio dentre outros, que tive o prazer de conviver durante boa parte desse programa.

Em especial a toda a minha família que nunca deixaram de acreditar em mim (eles pensam que sou bom nisso!) e me apoiaram financeiramente e moralmente sempre que eu precisei (meus orientadores, vocês quase foram crucificados por eles!).

Agradeço ao apoio financeiro da CAPES, ao qual tive a satisfação de contar com essa ajuda, que foi determinante para me manter durante este programa de mestrado.

Por último, propositalmente, pois o último é o primeiro agradeço ao nosso DEUS, por ter me guiado sempre em todas as decisões de minha vida e ser tão generoso para mim, apesar de dizer isso todos os dias a Ele, deixo registrado neste documento para que todos saibam.

RESUMO

Este trabalho tem como foco a abordagem do desenvolvimento de sistemas industriais com o uso do paradigma de Agentes, analisando a importância dos sistemas industriais, sua complexidade e composição, além de como tais sistemas podem ser abordados pelo paradigma de agentes através do uso de uma metodologia que detalhe todas as fases do desenvolvimento para uma aplicação industrial. Este trabalho também avalia as metodologias de desenvolvimento de sistemas industriais e de sistemas multiagente objetivando a escolha de uma metodologia adequada para ser aplicada em sistemas industriais inteligentes e, dessa forma, contemplar a modelagem de todas as fases para um sistema industrial moderno.

PALAVRAS-CHAVES:

Sistemas Industriais. Metodologia Prometheus. Agentes de Software

ABSTRACT

This work focuses on the approach to development of industrial systems using the paradigm of agents, analyzing the importance of industrial systems, its complexity and composition, and how such systems can be addressed by the paradigm of agents through the use of a methodology that detail all phases of development for industrial application. This study also evaluates the methodologies of developing and industrial systems multiagent systems aiming at the choice of an appropriate methodology to be applied in industrial systems and intelligent thus include the modeling of all phases for a modern industrial system.

KEYWORDS:

Industrial Systems. Prometheus Methodology. Software Agents

CONTEÚDO

1	INTRODUÇÃO.....	1
1.1	Introdução	1
1.2	Contextualização.....	2
1.3	Motivação	4
1.4	Objetivo	5
1.5	Metodologia.....	5
2	SISTEMAS INDUSTRIAIS	6
2.1	Conceitos	6
2.2	Tecnologias para sistemas industriais	8
2.2.1	<i>Tecnologias para compor soluções específicas</i>	10
2.2.2	<i>Tecnologias de redes industriais</i>	12
2.3	Tipos de sistemas industriais	13
2.4	Exemplos de sistemas industriais	15
2.5	Tendências futuras para sistemas industriais	17
2.6	Considerações finais do capítulo	18
3	AGENTES	20
3.1	Definição	20
3.2	Classificação dos agentes	21
3.3	Sistema multiagente.....	22
3.4	Importância do paradigma do agente.....	24
3.5	Agente em sistemas industriais.....	25
3.6	Considerações finais do capítulo	27
4	METODOLOGIAS.....	29
4.1	Conceito e importância da metodologia	29
4.2	Metodologias para sistemas industriais.....	30
4.2.1	<i>Análise das metodologias para sistemas industriais</i>	33
4.3	Metodologias para sistema multiagente	34
4.3.1	<i>Metodologia Mase</i>	35

4.3.2 Metodologia Prometheus	37
4.3.3 Metodologia Gaia	40
4.3.4 Análise das metodologias para sistema multiagente	42
4.4 Considerações finais do capítulo	44
5 EXTENSÃO DA METODOLOGIA PROMETHEUS E ESTUDO DE CASO	45
5.1 Descrição da extensão	45
5.1.1 Análise de requisitos	48
5.1.2 Micro-Arquitetura	48
5.1.3 Macro-Arquitetura	50
5.1.4 Implementação	51
5.2 Estudo de caso	52
5.2.1 Fase 1da metodologia Prometheus: Especificação do Sistema	53
5.2.2 Fase 2 da metodologia Prometheus: Projeto Arquitetural	55
5.2.3 Fase 3 da metodologia Prometheus: Projeto detalhado	58
5.2.4 Modelagem do sistema de automação	60
5.3 Considerações finais do capítulo	65
6 CONCLUSÕES	66
6.1 Contribuições	66
6.2 Limitações e trabalhos futuros	67
BIBLIOGRAFIA	68
ANEXO A: DESCRIÇÃO DO ESTUDO DE CASO	80
ANEXO B: DETALHAMENTO DA MODELAGEM DA AUTOMAÇÃO	88
ANEXO C : DETALHAMENTO DA MODELAGEM	91

LISTA DE SÍMBOLOS E ABREVIATURAS

Abreviatura	Descrição
AGV	Veículo Guiado Automaticamente
AUML	Agent Unified Modeling Language
BDI	Crenças Desejos e Intenções
CAD	Computer Aided Design
CAN	Computer Aided Manufacturing
CIM	Manufatura Integrada por Computador
CLP	Controlador Lógico Programável
CNC	Controle Numérico computadorizado
ERP	Enterprise Resource Planning
IHM	Interface Homem Máquina
ISO	International Organization for Standardization
OPC	OLE for Process Control
RUP	Rational Unified Process
SCADA	Sistema de Controle Supervisório e Aquisição de Dados
SED	Sistema de Eventos Discreto
SIM	Sistema Industrial Moderno
SMA	Sistema Multiagente
UML	Unified Modeling Language
XP	Extremme Programing

LISTA DE TABELAS

Tabela 3.1: Comparação entre as características desejadas aos SIM e as características dos SMA	28
Tabela 4.1: Comparação entre as metodologias de sistemas industriais	34
Tabela 4.2: Comparação entre as metodologias de SMA.....	43

LISTA DE FIGURAS

Figura 2.1: Exemplo de uma aplicação industrial contendo diversas tecnologias	13
Figura 3.1: Arquitetura de decisão baseada em agentes em vários níveis para sistema de produção	26
Figura 4.1: Fases da metodologia Mase	36
Figura 4.2: Fases da metodologia Prometheus	38
Figura 4.3: Fases da metodologia Gaia	41
Figura 5.1: Extensão da metodologia Prometheus.....	46
Figura 5.2: Adicionndo a extensão a metodologia Prometheus.....	47
Figura 5.3: Modelando objetos da automação como classes.....	49
Figura 5.4: Diagrama de classes de um controlador de temperatura.....	49
Figura 5.5: Diagrama de sequência de um controlador de temperatura	50
Figura 5.6: Criando pacotes a partir de classes	50
Figura 5.7: Criando componentes a partir de pacotes	51
Figura 5.8: Exemplo de diagrama de implantação	51
Figura 5.9: Diagrama de casos de uso da especificação do sistema.....	52
Figura 5.10: Diagrama de metas do estudo de caso	53
Figura 5.11: Diagrama de cenários do estudo de caso	54
Figura 5.12: Diagrama de análise do sistema do estudo de caso.....	54
Figura 5.13: Diagrama de papéis do estudo de caso.....	55
Figura 5.14: Diagrama de papéis de agente do estudo de caso	56
Figura 5.15: Diagrama de acoplamento de dados do estudo de caso	57
Figura 5.16: Diagrama de relacionamento do agente do estudo de caso.....	57
Figura 5.17: Diagrama de revisão do sistema do estudo de caso	58
Figura 5.18: Digrama de revisão do agente cozinheiro	59
Figura 5.19: Diagrama de revisão de capacidades do agente cozinheiro	60
Figura 5.20: Diagrama de casos de uso da automação	61
Figura 5.21: Diagrama de classes do sistema de automação	62
Figura 5.22: Diagrama de sequencia do sistema de automação	62
Figura 5.23: Diagrama de pacotes do sistema de automação	63
Figura 5.24: Componentes do sistema de automação.....	64

Figura 5.25: Diagrama de implantação do sistema de automação	64
--	----

CAPÍTULO 1

INTRODUÇÃO

1.1 Introdução

Uma indústria é composta por uma coleção de processos, onde tais processos representam a transformação da matéria-prima em um produto acabado (Corrêa e Corrêa, 2005). Para que uma indústria possa garantir a qualidade de seu produto, ela deve primeiramente garantir a qualidade de seus processos, sendo essa qualidade um índice positivo de satisfação do cliente com um produto (Duran, 2004).

Um estudo realizado pelo BNDES (Banco Nacional de Desenvolvimento) sobre investimentos em sistemas de informação voltados para a indústria aponta como vantagens principais (Gutierrez e Pan, 2008):

Minimização dos erros humanos: Pois os processos humanos passam a ser executado por máquinas que garantam a precisão na execução;

Qualidade no processo: Isso devido à minimização de possíveis variações que poderiam ocorrer quando processos são executados por humanos;

Dados do processo: Os dados sobre um processo poderão ser coletados e armazenados e podem ser utilizados para verificação da execução dos processos;

Redução de perdas no processo: Uma vez que desperdícios de matéria-prima, equipamentos e horas trabalhadas, podem ser evitados pela previsão do processo.

Tais fatores citados são determinantes para a adoção do uso de sistemas de informação voltados ao domínio industrial. Em alguns casos esses sistemas são essenciais em determinados segmentos industriais, como no setor automotivo, onde os sistemas são

ferramentas que ajudam a manter a competitividade das empresas através da melhora na produtividade (Boneli, 2001).

Porém, as soluções para sistemas de informações industriais se deparam com o problema dos fatores diferenciadores entre as indústrias, tais como: a natureza das operações e dos processos, as tecnologias adotadas e os modelos de manufatura (Slack, Chambers e Johnson, 2008). Tais fatores fazem com que os sistemas industriais tenham de ser adaptados para contemplar a realidade de cada indústria, mesmo que existam soluções disponíveis e que se tenham indústrias que atuem em um mesmo segmento.

1.2 Contextualização

Um sistema de informação envolvendo uma ou mais atividades correlacionadas à manufatura de uma indústria pode ser classificado como sendo um sistema industrial (Quingi, 2006). É comum relacionarmos a designação de sistema industrial aos sistemas de automação industrial, que corresponderá a um sistema com capacidade de interagir diretamente em um ou vários processos de uma indústria (Tommila et al., 2001). Porém, um sistema industrial pode conter um ou mais sistema de automação industrial e exercer operações de controle, monitoramento e intervenção em um sistema de automação industrial.

Normalmente, um sistema de automação industrial pode ser abordado em sua análise pela teoria dos sistemas de eventos discretos (SED), na qual tais sistemas possuem a característica de serem dinâmicos com mudanças de estados em pontos discretos do tempo em decorrência de eventos isolados (Queiroz e Cury, 2003). Para modelar esses sistemas de eventos discretos com relação ao tempo, o formalismo proposto por Ramage e Wohan (1989) é bastante utilizado, uma vez que estabelece a distinção entre a planta, que representa o comportamento possível para o sistema, e o supervisor, que exerce uma função restritiva à planta (Cury, 2001). As redes de Petri também são usadas para a modelagem dos sistemas de eventos discretos (Cassandras e Lafortune, 1999).

Compondo uma solução industrial, temos a possibilidade de escolha de diversas tecnologias, dentre elas podemos citar o controlador lógico programável (CLP), sensores e atuadores, que juntos podem ser usados no controle e monitoramento de um processo. Também podem ser citadas as tecnologias prontas, como máquinas específicas para um processo provida de tecnologia de processamento e comunicação (Bruciapaglia, Farines e Cury, 2001). Porém, quando há a necessidade de coletar, armazenar dados, gerenciar, prover informações em tempo real ou através de relatórios do processo, um sistema industrial deve

ser construído para atender essas necessidades juntamente com o sistema de automação, que será essencial para atingir o objetivo de atender tais necessidades.

Dessa forma, a construção de sistemas industriais envolve o desenvolvimento de um sistema que contemple questões como a comunicação com os equipamentos usados em um processo de automação, que podem ser de diversos fabricantes constituindo tecnologias diferentes, conexões com banco de dados, o desenvolvimento de interfaces para prover uma melhor interação entre o ser humano e a máquina, sistemas de supervisão avançados e de alarmes de monitoramento dos processos. Devido às exigências dessa natureza, a construção de sistemas industriais nem sempre é uma tarefa simples e em muitos os casos as funcionalidades do sistema podem ser distribuídas entre os diversos equipamentos com poder de processamento exigidos na solução, como computadores tipo PC, CLP e máquinas do processo, tornando complexo o projeto de desenvolvimento destes sistemas (Da'na et al., 2008).

Plataformas de desenvolvimento para sistemas industriais, referenciados por SCADA (*Supervisory Control e Data Aquisition System*), são plataformas de desenvolvimento voltadas para o domínio das aplicações industriais. Muitas delas usam a tecnologia de programação orientada a objetos, oferecendo diversos *drivers* de comunicação com equipamentos e permitem a comunicação entre a aplicação e o banco de dados além de imagens e ícones apropriados para o desenvolvimento de interfaces de comunicação através de telas IHM (Interface Homem Máquina). Tais plataformas também podem implementar sistemas de alarmes pré-configurados e permitem a construção de algoritmos complexos de controle para as aplicações. Como exemplo dessas plataformas que utilizam-se da tecnologia da programação orientada a objetos podemos citar: *InTouch* da *Wonderware* (WonderWare, 2009), *E3* da *Elipse software* (Elipse, 2009) e o *Genesis* da *Iconics* (Iconics, 2009).

À medida que as aplicações industriais vão se tornando maiores e conseqüentemente mais complexas, novas técnicas e paradigmas de desenvolvimento de sistemas se fazem necessárias para contemplar o desenvolvimento para esses sistemas (Giret e Botti, 2009), (Monostori, Váncza e Kumara, 2006).

O paradigma de desenvolvimento de software baseado em agentes desponta como uma solução promissora para o desenvolvimento de sistemas industriais modernos (Marick e McFarlani, 2005). Isso, devido à propriedades que um agente pode ter, tais como: autonomia, habilidades sociais, reatividade, orientação por objetivos e aprendizado.

A Engenharia de Software Baseada em Agentes fornece técnicas, métodos e ferramentas para a modelagem de um software através do paradigma de desenvolvimento de software baseado em agentes (Girardi, 2004).

As metodologias de software orientadas a agentes são alvos de estudos por parte da Engenharia de Software Baseada em Agentes, fornecendo uma forma organizada para a construção de um software por meio de passos preestabelecidos, permitindo o uso de uma ou várias técnicas, ferramentas e processos de construção do software pelo paradigma dos agentes. São exemplos de metodologias orientadas a agentes: Mase (DeLoach et al., 2001), Passi (Cossentino & Potts, 2001), Gaia (Zambonelli, Jennings e Woodridge, 2003) e Prometheus (Padghan e Winiko, 2002).

1.3 Motivação

Gutierrez e Pan (2008) enfatizam que existe uma demanda substancial por sistemas industriais em todo o mundo e que tal demanda cresce à medida que novas tecnologias se tornam mais acessíveis às indústrias.

Kumar et al (2004), Madejski (2007) e Giret e Botti (2009) acrescentam que fatores externos às indústrias, como: globalização, novas relações de consumo, exigências de selos que atestem qualidade do produto e cuidados com o meio ambiente e ciclo de vida reduzido dos produtos, influenciaram diretamente para aumento da complexidade dos sistemas industriais.

Os benefícios da adoção dos sistemas indústrias e o fato de haver uma crescente demanda por esses sistemas, levando em consideração o aumento da complexidade e a necessidade de termos novos métodos para o desenvolvimento de tais sistemas, bem como o uso do paradigma dos agentes no domínio industrial, nos motivou para a realização deste trabalho.

1.4 Objetivo

Apresentar uma metodologia para o desenvolvimento de sistemas industriais modernos contemplando características inerentes a esse tipo de sistema, como: natureza distribuída, interoperabilidade, heterogeneidade, estruturas abertas e dinâmicas e tolerância a falhas.

Como objetivos específicos deste trabalho, temos:

1. Apresentar a importância do paradigma de agentes no domínio dos sistemas indústrias;
2. Propor um método para abordar o desenvolvimento de sistemas industriais através do uso de uma metodologia orientada a agentes.

1.5 Metodologia

Para atingirmos os objetivos propostos neste trabalho, organizamos este segundo a distribuição dos próximos capítulos: No Capítulo 2, apresentamos uma revisão bibliográfica sobre os sistemas industriais. No Capítulo 3, temos uma revisão bibliográfica do paradigma de agentes. No Capítulo 4, apresentamos uma revisão bibliográfica sobre metodologias de desenvolvimento de sistemas industriais e metodologias de sistemas baseados em agentes. No Capítulo 5, apresentamos uma metodologia de desenvolvimento orientada a agente para sistemas industriais com um estudo de caso. No Capítulo 6, apresentamos as conclusões e trabalhos futuros.

CAPÍTULO 2

SISTEMAS INDUSTRIAIS

Neste capítulo apresentamos uma revisão bibliográfica sobre os sistemas industriais, buscando uma breve contextualização sobre o tema. Para isso apresentamos na Seção 2.1 os conceitos de sistemas industriais. Na seção 2.2, discutimos sobre os componentes e tecnologias que são usados para compor um sistema industrial. Na seção 2.3 apresentamos uma classificação para os sistemas industriais. Na seção 2.4, são apresentados alguns exemplos de sistemas industriais. Na seção 2.5 apresentamos as tendências futuras de tecnologias para o domínio dos sistemas industriais. Na seção 2.6 apresentamos as considerações finais sobre o capítulo.

2.1 Conceitos

Um sistema de automação industrial pode ser definido como uma coleção de dispositivos distribuídos e interconectados por meio de uma rede de comunicação que são usados no monitoramento e/ou controle de um processo de manufatura (Tommila et al., 2001). O processo de manufatura pode ser entendido como a utilização de recursos para transformar as entradas em uma saída desejada, ou seja, um processo de produção onde a entrada pode ser a matéria prima, os recursos de máquinas e ou mão-de-obra e como resultado temos o produto manufaturado (Corrêa e Corrêa, 2005).

Um sistema industrial é conceituado por Quingui (2006) como um sistema de informação que envolve uma série de atividades correlacionadas com a manufatura de uma indústria. Neste trabalho, o termo sistema industrial significará o mesmo que sistema de informação industrial.

A relação entre um sistema de automação e um sistema industrial é que os sistemas de automação abordam principalmente questões sobre a tecnologia a ser usada na automação, controle e monitoramento do processo de manufatura. Os sistemas industriais abordam as questões dos sistemas de automação e outras relacionadas às atividades que se fazem necessárias em uma indústria, tais como: controle do processo de produção, planejamento da

produção, controle de qualidade, administração de estoques e comercialização dos produtos, bem como atividades relacionadas de alguma forma com o processo de manufatura de uma indústria, tornando o desenvolvimento de um sistema industrial complexo, por abranger questões multidisciplinares (Quingi, 2006).

Nos últimos anos, as pressões exercidas pela globalização às indústrias fizeram com que indústrias localizadas em outros países se tornassem concorrentes das indústrias nacionais. Fatores como (i) as novas relações de consumo, onde os clientes fazem exigências de maior variedade de produtos e produtos cada vez mais personalizados, (ii) a aquisição de selos e certificados que comprovem os cuidados da empresa com o meio ambiente e (iii) um menor tempo de resposta ao mercado, resultando na diminuição do ciclo de vida de projeto de produtos, têm feito com que as empresas mudem suas estratégias de manufatura, mudando também a forma como optam por uma tecnologia que as auxiliem a enfrentar tais desafios, refletindo em novas exigências para os sistemas industriais (Kumar et al., 2004), (Madejski, 2007), (Giret e Botti 2009).

O termo Sistema Industrial Moderno (SIM) é utilizado para designar um sistema industrial que auxilie as indústrias a enfrentar essas novas exigências. Para isso, tais sistemas devem possuir características como (Monostori, Váncza e Kumara, 2006), (Madejski, 2007):

Natureza distribuída: As diversas funções do sistema deverão ser distribuídas entre unidades autônomas de produção, de modo a explorar da melhor forma possível o potencial de cada unidade.

Interoperabilidade: A capacidade de interagir com outras unidades do sistema, mesmo soluções tecnológicas diferentes, dará o poder de comunicação e interação entre as unidades.

Heterogeneidade: Os sistemas poderão ser compostos de soluções tecnológicas diferentes.

Estruturas abertas e dinâmicas: Provendo a evolução do sistema sempre que necessário e permitindo mudanças rápidas na reconfiguração do mesmo.

Tolerância a falhas: As falhas nos sistemas deverão ser tratadas de modo a não prejudicar o sistema como um todo.

Desenvolver sistemas industriais para contemplar tais características é uma tarefa complexa e soluções já prontas não se aplicam a uma grande variedade de aplicações. É comum o uso de técnicas provenientes da inteligência artificial para a abordagem de tal complexidade de desenvolvimento (Leitão e Restivo, 2008).

Dentre as técnicas da inteligência artificial destacam-se os sistemas de manufatura holonic, do termo em inglês *Holonic Industrial System* (Giret e Botti 2009), e a inteligência artificial distribuída, como o paradigma dos agentes (Madejski, 2007), que será detalhado posteriormente no capítulo 3 deste trabalho.

2.2 Tecnologias para sistemas industriais

Dentre as soluções que podem ser utilizados na construção de um SIM, podemos dividir em dois grandes grupos: as tecnologias denominadas de “*key-on-hands*”, que são soluções tecnológicas prontas para serem utilizadas na automação de processos (Bruciapaglia, Farines e Cury, 2001) e as tecnologias de soluções específicas, que podem ser utilizadas para compor uma solução específica a um problema de automação do processo.

Dentre as soluções prontas que podem ser adquiridas para automação de processos, podemos citar as estações de automação da *Honeywell*, que foram consideradas como os primeiros sistemas distribuídos de controle digital surgidos na década de 80 do século passado, constituindo-se de uma arquitetura completa de automação (Samad, McLaughlin e Lu, 2006).

Dentre as diversas tecnologias prontas existentes, que podem ser adquiridas e adaptadas a um sistema industrial, destacamos as seguintes:

Máquinas CNC: Máquinas de Controle Numérico Computadorizado (CNC) são máquinas que contêm instruções de trabalho em um computador acoplado para que possam trabalhar com o mínimo de intervenção humana. Surgiu na década de 70 como uma alternativa de automação para a fabricação de lotes pequenos e variados de um item produzido (Xu, Newman, 2006). Seu funcionamento é baseado na tecnologia CAD/CAM, que corresponde à integração das duas técnicas: a técnica CAD (*Computer Aided Design*), que consiste no desenvolvimento de projetos com o auxílio

de um software gráfico e a técnica CAM (*Computer Aided Manufacturing*), que significa todo e qualquer processo de fabricação controlado por computador (Demec, 2009). Tais máquinas podem ser utilizadas para projetar um componente qualquer em uma tela de computador e transmitir as informações diretamente para a manufatura, que construirá o componente projetado.

AGV: A sigla vem do termo em inglês *automated guided vehicles*, que corresponde a tecnologia de veículos guiados automaticamente. Essa tecnologia proporciona mobilidade e flexibilidade a manufatura, uma vez que permite o transporte de itens para diversos pontos da manufatura de forma automática e autônoma, sendo a autonomia uma das preocupações dessa tecnologia (Perez, 2008).

Robô: Robô industrial é uma tecnologia que foi inicialmente desenvolvida com o objetivo de substituir o ser humano em determinadas tarefas em uma manufatura. Hoje, essa tecnologia pode superar o ser humano em algumas funções que exigem maior precisão e força. É uma tecnologia cara, bastante empregada na indústria automotiva em funções como soldagem e pintura de automóveis (Brograd, 2007).

FMS: Um sistema flexível de manufatura, tradução do termo inglês *Flexible Manufacturing Systems*, pode ser definido como um sistema de fabricação formado por estações de trabalho que compartilham um sistema de transporte, podendo ser um AGV, de forma a permitir maior flexibilidade para a produção de itens diferentes (Chan, 2006).

CIM: A manufatura integrada por computador, tradução do termo em inglês *Computer Integrated Manufacturing*, é uma arquitetura para integração da manufatura com outros setores estratégicos da empresa. Pode ser composta de tecnologias como CAD/CAM, equipamentos CNC, Robôs, FMS integrados por um sistema de computador que concilia o processo de manufatura com o planejamento da produção. Tal tecnologia tem que ser vista sob a perspectiva de toda a organização e não simplesmente como uma tecnologia (Gunasekaran, 1997).

A principal vantagem da aquisição de uma tecnologia pronta está na economia de tempo, quando comparado a um projeto de desenvolvimento de toda a automação, e na confiabilidade da solução, quando se leva em conta que outras empresas já adotaram determinada solução. Porém, fatores como o custo de aquisição da tecnologia, vantagens que a mesma proverá quando for incorporada à manufatura, manutenção exigida e o custo de operação limitam a viabilidade da aquisição deste tipo de tecnologia para a automação em uma indústria de manufatura (Slack, Chabers, Johnnton 2008).

Quando uma indústria possui exigências que as tecnologias “*key-on-hands*” não podem atender ou mesmo quando o custo de aquisição não for interessante, tal indústria pode optar por desenvolver sua própria solução de automação. Tal problema é comum na indústria de beneficiamento de petróleo e gás, que normalmente tendem a desenvolver suas próprias soluções em automações, devido, principalmente, à natureza específica de seus problemas e ao fato de não haver soluções que contemplem suas exigências. Souza et al. (2006) e Barbosa (2009) apresentam soluções de automação para a indústria de petróleo e gás, partindo do desenvolvimento de soluções específicas.

2.2.1 Tecnologias de soluções específicas

As tecnologias que iremos citar nesta subseção, algumas delas, também podem ser enquadradas como soluções prontas, pois podem ser aplicadas diretamente na solução de um problema específico como, por exemplo, um controlador de pressão ou um controlador de temperatura. Porém, tais soluções podem vir a ser utilizadas como componentes de uma solução maior, como a automação de uma caldeira que pode necessitar utilizar os controladores citados como exemplos.

Abaixo, citamos as principais tecnologias para compor soluções para sistemas industriais:

Sensor: Um sensor é um dispositivo com a capacidade de transformar uma grandeza física em uma grandeza elétrica correspondente. Os sensores são comumente utilizados para medir grandezas como: temperatura, pressão, fluxo, dentre outros. Sua importância é determinante para o sucesso de automações em áreas como a indústria da manufatura, medicina e aplicações aeroespaciais (Sensores, 2009).

Atuador: Um atuador é um dispositivo que pode exercer uma ação em um processo. Dentre os tipos de atuadores existentes se destacam os pneumáticos, que usam a energia do ar comprimido para executar movimentos essenciais ao seu funcionamento, intervindo em um processo (Mecatrônica, 2008).

Controlador: Um controlador pode ser definido como um dispositivo que irá controlar automaticamente um determinado processo de forma esperada. O controlador lógico programável (CLP) é considerado como um exemplo de controlador, pois pode conter uma lógica interna que determinará o controle de um processo com a possibilidade de interagir com tal processo em conjunto com sensores e atuadores (Erickson, 1996).

Controlar um processo industrial significa atuar neste processo de forma adequada. Esse controle pode ser através do uso de sensores, atuadores e controladores específicos para uma função de controle em um determinado processo (Mecatrônica, 2008).

Porém, em modernas instalações industriais, os sistemas de controles devem garantir um desempenho satisfatório na operação de processos complexos. Uma técnica de controle bastante utilizada é a *Proporcional Integral Derivativa* (PID), que consiste em uma função matemática para ajustes no controle de um processo (ARRUDA et al., 2008). Outras técnicas de controle surgiram nos últimos anos, como o controle nebulosos, baseada na lógica nebulosa (CALLAI et al., 2007), e o controle preditivo, baseado em cálculos de previsões para decisões de controle (SILVA et al., 2007).

Técnicas mais avançadas de controle não são suportadas por dispositivos controladores comuns, como o CLP, por necessitarem de um grande poder de processamento. O computador pessoal vem sendo utilizado na indústria para suprir as necessidades de maior poder de processamento e grande capacidade de armazenagem de informação a um custo relativo baixo (Kummar et al., 2005).

O uso do PC na indústria favoreceu a questão de aumento das exigências aos softwares industriais, que passaram de simples programas de controle de processo a sistemas de gerenciamento de produção autônomos e integrados a outros setores da empresa, como contabilidade, estoque, comercial, dentre outros, para prover informações estratégicas, auxiliando no processo de administração em uma indústria (Quingi, 2006).

Neste cenário de evolução dos sistemas industriais, surge o Sistema de Controle Supervisório e de Aquisição de Dados (SCADA), que pode ser conceituado como uma

plataforma para desenvolvimento de sistemas industriais que tem como principal característica prover um ambiente de desenvolvimento para aplicações industriais com várias ferramentas de auxílio. Ferramentas estas que possibilitam a criação de banco de dados, interfaces IHM, comunicação com dispositivos controladores, criação de alarmes, gráficos, relatórios, dentre outras funcionalidades, facilitando o desenvolvimento de uma aplicação industrial e reduzindo o tempo e o custo de um projeto dessa natureza, quando comparado ao desenvolvimento de um sistema industrial completo através de uma linguagem de programação de uso comum como a Java e C++ (Daneels, Salter, 1999).

2.2.2 *Tecnologias de redes industriais*

Uma rede industrial possibilita a comunicação entre dois ou mais equipamentos usados na automação de processos industriais através de componentes presentes que transmitem e recebem informações, através de um meio, regido por um protocolo de comunicação, ou padrão, que possibilita a lógica para a troca de dados. Muitos são os protocolos e padrões que podem ser usados para garantir a troca de informações entre equipamentos industriais, dentre eles temos: *Modbus*, *DeviceNet*, *Profibus*, *Fieldbus* e *Ethernet* (Mackay et al., 2004).

As redes industriais podem ser compostas de uma ou mais tecnologias de rede, como as citadas acima, objetivando a comunicação entre diversos tipos de equipamentos no intuito de prover a interoperabilidade entre os dispositivos e possibilitar a computação distribuída. Eles devem suportar as características comuns ao ambiente industrial, tais como: ruídos, interferências eletromagnéticas, e altas temperaturas. Podemos ter uma rede de comunicação entre Máquinas CNC, sensores e atuadores com CLP, comunicação entre CLP e SCADA. Enfim, a ligação entre os diversos tipos de soluções tecnológicas que se façam presentes em um sistema industrial através das diversas soluções de comunicação, provendo a interoperabilidade entre os diversos dispositivos (Moyne e Tylbury, 2007).

A Figura 2.1 ilustra uma aplicação industrial contendo diversas soluções tecnológicas, como Robôs, AGV, Máquinas CNC, sensores, atuadores e CLP's, interligadas por diversas soluções como OPC,¹ *DeviceNet*, *Profibus* e *Ethernet*.

¹ OPC (*OLE for Process Control*): Padrão aberto de comunicação utilizado para a disponibilização de dados coletados na planta industrial para aplicativos diversos, como sistemas supervisórios e interfaces IHM (Moyne e Tylbury, 2007)

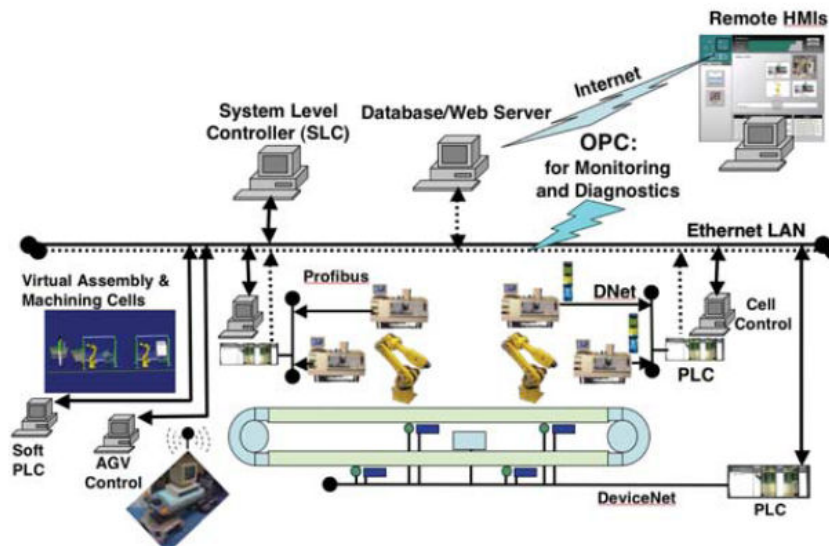


Figura 2.1: Exemplo de uma aplicação industrial contendo diversas tecnologias (Moyne e Tylbury, 2007).

De todas as tecnologias de comunicação existentes no domínio industrial, a comunicação baseada no protocolo Ethernet e TCP/IP vem sendo adotada pelos fornecedores de máquinas e equipamentos como um padrão em seus equipamentos. A razão pela qual isso está acontecendo é devido à possibilidade de conexão desses equipamentos à internet, possibilitando que o mesmo possa ser operado e monitorado remotamente, embora isso possa gerar problemas como a segurança e a confiabilidade dos dados industriais que trafeguem pela internet (Decotignie, 2005).

Tais tecnologias de comunicação aliadas ao PC e à tecnologia da internet possibilitam os sistemas industriais a quebrar barreiras físicas impostas pela distância e possibilitam também que esses sistemas possam ser controlados e monitorados remotamente através de redes de longa distância (Moyne e Tilbury, 2007).

2.3 Tipos de Sistemas Industriais

Devido à grande quantidade de situações que podem ser encontradas no ambiente industrial, refletindo em uma grande variedade de sistemas com características diferentes sendo utilizados na indústria, variando desde os mais simples, como a automação de um processo de manufatura apenas com um CLP, a sistemas de maior complexidade, contendo diversos dispositivos dos mais variados tipos para prover uma solução completa para uma

indústria, classificar tais sistemas não é uma tarefa fácil. Wang e Tan (2006) descrevem alguns grupos de sistemas industriais que podem ser encontrados:

Sistema de controle e medição: São sistemas que trabalham com sensores, atuadores e controladores medindo variáveis importantes de um processo industrial podendo atuar para controlar tais variáveis. Tais sistemas normalmente são encontrados no nível de controle de processo, com o uso de controladores tipo CLP's por exemplo, ou podem chegar ao nível de supervisão, com um sistema do tipo SCADA. Tais sistemas são encontrados comumente na indústria química e petrolífera.

Sistema de monitoramento e alarmes: São sistemas usados para o monitoramento de variáveis críticas de um processo de manufatura. Normalmente esse tipo de sistema pode utilizar-se de controladores para a automação do processo. Porém, há casos em que só encontraremos sensores no nível da planta, com o uso de um sistema tipo SCADA para o acionamento de alarmes. Este tipo de sistemas é comum em indústrias que possuem processos críticos.

Sistemas de aquisição de dados e apresentação de processos: São sistemas que coletam dados importantes diretamente da planta de produção, mostrando o processo em tempo real através de monitores de vídeo e armazenando dados referentes a esses processos em banco de dados. Sistemas desta natureza são comuns em muitos tipos de indústria que necessitem de dados sobre o processo.

Sistema de manufatura integrada: A automação da manufatura é integrada a outros sistemas, que podem reger o funcionamento do sistema da automação. Esse tipo de sistema pode ser encontrado em indústria com alto nível de gerenciamento da produção.

Sistemas de supervisão de qualidade: Neste tipo de sistemas, temos a supervisão e atuação em variáveis do processo importantes para a qualidade do produto, sendo emitidos relatórios destes dados a fim de controlar a qualidade dos produtos. São sistemas auxiliares importantes para o controle de qualidade em uma indústria.

A classificação de um sistema industrial por tipo, não é aplicável pelo fato de que eles normalmente serão classificados em mais de um tipo. Por essa impossibilidade de classificação é que consideramos o termo sistema industrial abrangente a todo e qualquer sistema, software e hardware, presente na indústria ligado, de alguma forma, a manufatura desta.

2.4 Exemplos de sistemas industriais

Vários trabalhos apresentam exemplos de soluções para os sistemas industriais. A seguir, apresentamos alguns trabalhos relevantes classificando-os segundo os tipos descritos por Wang e Tang (2006):

Sistema de monitoramento e alarme: em Shankar (2008), é apresentada a automação de uma caldeira industrial com o uso de CLP, sensores, atuadores e um sistema supervisório. A automação da caldeira possibilitou que a mesma funcionasse de forma automática, pois os atuadores permitiram que insumos como água e combustível entrassem automaticamente, alimentando a caldeira de forma controlada pela CLP, com a instalação de sensores, principalmente os sensores de temperatura e pressão. O operador, que antes visualizava os valores das variáveis de controle da operação da caldeira através de instrumentos como manômetros e indicadores, passa a visualizar as variáveis através de uma tela de computador com uma interface intuitiva de forma a tornar mais fácil o entendimento da situação que ocorre na caldeira em tempo real. O supervisório tem a função de monitoramento, principalmente dos valores de pressão e temperatura da caldeira, de forma a emitir alarmes se alguns dos parâmetros monitorados estiverem fora do valor aceitável.

Sistemas de aquisição de dados, apresentação de processos e manufatura integrada: em Pouty (2005), é apresentado um sistema desenvolvido para uma indústria de cimento com uso de CLP's, sensores e atuadores, na automação da planta industrial e um sistema supervisório com funcionalidades como: (a) representação visual dos processos (b) telas IHM, facilitando a interação do operador com o processo (c) alarmes para situações críticas (d) gráficos de desempenho dos equipamentos e (e) o armazenamento de dados da planta. Com essas funcionalidades, o sistema trouxe vantagens como o monitoramento eficaz da produção, facilidade na operação de

máquinas através da IHM e melhoria para segurança da indústria através dos alarmes. Com o banco de dados do sistema industrial compartilhado com o sistema da administração ERP², foi possível disponibilizar informações provenientes da produção para outros setores da empresa, tornando as operações da administração mais dinâmicas. Como desvantagens desse sistema foram apontadas: o treinamento dos operadores, para que pudesse tirar melhor proveito das informações que o sistema supervisorio iria dispor, e a re-configuração do sistema, calibração dos equipamentos, todas as vezes que um sensor ou atuador for trocado.

Sistema de Manufatura Integrada: Djassemi e Sena (2006), aponta para uma tendência onde o modelo de produção na planta industrial é determinada por metas de produção, ditadas por estratégias comerciais. O artigo define um *framework* que pode ser utilizado para avaliar os índices de produção na planta industrial, através de monitores instalados em diversas partes da planta, onde as ordens de produção são recebidas via internet pelo sistema ERP da indústria e repassadas ao sistema industrial. O destaque do trabalho é para o não uso de papel durante todo o processo, apenas utilizando formulários eletrônicos que percorrem os diversos níveis necessários para que chegue à produção.

Sistema de Manufatura Integrada e Flexível: Em Ferrarini et al. (2006), é apresentado um estudo de caso de uma empresa que produz lápis, onde há necessidade de que as máquinas sejam re-configuradas a cada novo item de um pedido. A solução apresentada é baseada em agentes, onde cada pedido proveniente do sistema ERP é direcionado ao sistema de manufatura, que cria agentes para representar os pedidos de forma que os mesmos negociam entre si o uso das máquinas da produção, coordenados por um agente supervisor do processo. O agente assume a máquina e a re-configura de acordo com suas necessidades em cumprir seu objetivo de atender ao pedido. O agente supervisor trabalhará no sentido de garantir que o tempo de cada pedido seja respeitado e de que as máquinas sejam utilizadas da melhor forma, provendo economia em não haver paradas para reconfiguração das máquinas da fábrica. Um sistema do tipo SCADA supervisiona o processo e cuida das máquinas, monitorando variáveis importantes. Em Marik et al. (2005), temos o uso de agentes controlando

² *Enterprise Resource Planning*: sistema de software de informação empresarial que reúne os diversos setores de uma empresa em um único sistema de informação (Moom, 2007).

uma máquina de produção de barras de aço, caracterizada também pela necessidade de ser re-configurada conforme o tipo de barra a ser fabricada, seguindo as ordens de produção que chegam do sistema ERP da indústria. Neste caso, os agentes estão embarcados na própria máquina. Os agentes re-configuram a máquina dinamicamente de acordo com suas necessidades, em um framework que permite a comunicação entre eles localizado internamente nas CLP's que controlam o processo.

2.5 Tendências futuras para os sistemas industriais

Apesar das particularidades do domínio industrial, as tendências futuras para esses sistemas em termos de emprego de novas tecnologias, sejam elas desenvolvidas especialmente ou adaptadas ao domínio industrial, e os benefícios que as mesmas proverão aos sistemas industriais, são apresentadas nos trabalhos de Marick e McFarlane (2005), Monostori, Váncza e Kumara (2006) e Morel et al. (2007). Estes autores apontam como tendências futuras para os sistemas industriais:

Computação Embarcada: Os dispositivos serão cada vez menores e com maior capacidade de processamento, sendo facilmente integráveis aos sistemas industriais favorecendo as soluções distribuídas.

Equipamentos Plug and Play: Os equipamentos adquiridos serão facilmente integrados ao sistema industrial, provendo a interoperabilidade entre as soluções tecnológicas.

Controle Descentralizado: O controle do sistema estará espalhado nos diversos equipamentos que compõem a solução do sistema.

Autonomia: Os componentes do sistema terão autonomia para tomar decisões.

Arquitetura orientada a serviços (SOA): Os sistemas industriais poderão utilizar módulos adquiridos através de um serviço de fornecimento de componentes através da internet para evoluírem.

Orientação a Objetos: Apesar de compor boa parte dos sistemas industriais atuais a orientação a objetos não é uma realidade para a programação de controladores, como a CLP.

Wireless: Cada vez mais os fios, que caracterizam barreiras aos projetos de redes, serão substituídos pela tecnologia wireless.

Internet: As soluções industriais que utilizam a internet para romper distâncias entre unidades de produção tenderão a crescer cada vez mais.

Padrões Abertos: Cada vez mais as soluções serão baseadas em arquiteturas abertas para permitirem a integração entre os componentes da solução.

2.6 Considerações finais do capítulo

Este capítulo abordou a temática dos Sistemas Industriais, descrevendo brevemente sobre suas características, complexidade de desenvolvimento e tecnologias que podem vir a serem utilizadas na composição destes sistemas, como as tecnologias prontas para automação e as tecnologias de componentes para uso na automação além das tecnologias de comunicação entre as soluções.

Conceituamos um Sistema Industrial Moderno como sendo uma evolução dos Sistemas Industriais através das características que tais sistemas devem contemplar. Características estas provenientes dos novos desafios e tendências exigidas às indústrias neste novo século.

Apresentamos uma classificação em tipos para os sistemas industriais, que embora não usual, devido às muitas situações encontradas no domínio industrial, pode ser usada para agrupar os sistemas industriais.

Também apresentamos na Seção 2.4 exemplos encontrados de aplicações de sistemas industriais, enfatizando suas características e benefícios. Dentre os exemplos apresentados de sistemas industriais, alguns deles, apresentavam características inerentes aos SIM, tais como a interoperabilidade e a natureza distribuída encontrada no sistema descrito por Prouty (2005) e Djassemi e Sena (2006). Além destas características, nos trabalhos de Marik et al. (2005) e Ferrarine et al. (2006) encontramos também as características de estruturas abertas e

dinâmicas e a heterogeneidade, dessa forma caracterizando-os como exemplos de SIM, sendo ambas as soluções através da abordagem dos agentes.

Dessa forma, concluímos então que o uso dos agentes como solução para os sistemas industriais modernos é aplicável, uma vez que fornece técnicas e conceitos que permitem abordar a complexidade inerente a tais sistemas. Tal fato nos motivou a abordar o paradigma dos agentes no próximo capítulo.

CAPÍTULO 3

AGENTE

Neste Capítulo apresentamos o paradigma de desenvolvimento de software baseado em agentes, discutindo sobre os conceitos e características do agente na seção 3.1. Na seção 3.2 apresentamos uma classificação para os agentes segundo os tipos de agentes. Na seção 3.3 apresentamos o conceito e as características dos sistemas multiagentes. Na seção 3.4 discutiremos a importância do paradigma dos agentes. Na seção 3.5 mostramos a importância desse paradigma para os sistemas industriais. Na seção 3.6 são apresentadas as considerações finais deste capítulo.

3.1 Definição

Um agente de software pode ser conceituado como uma entidade computacional capaz de ações flexíveis e autônomas em um ambiente dinâmico, aberto e imprevisível (Jennings e Bussmann, 2003). Para Russel e Norvig (2004), a definição de um agente é a de uma entidade que percebe o ambiente por meio de sensores e atua por meio de atuadores.

O agente de software deve ser uma entidade dotada de certa autonomia e que apresenta um comportamento “racional” no ambiente ao qual ele está inserido. Russel e Norvig (2004) conceituam a racionalidade de um agente como sendo a forma pela qual o agente toma suas decisões a partir de uma sequência de percepções do ambiente ao qual ele está inserido, de forma que tais decisões sejam as melhores decisões possíveis a serem tomadas.

Além de autonomia e comportamento racional, um agente de software pode ser dotado de outras capacidades. Garcia et al. (2004) aponta as seguintes capacidades que um agente de software pode possuir:

Interação: O agente pode interagir com o ambiente ao qual ele está inserido como também pode interagir com outros agentes através de mensagens.

Adaptação: Um agente pode adaptar-se ou modificar-se de acordo com mensagens recebidas de outros agentes ou da percepção de modificações no ambiente.

Autonomia: Um agente deve ser capaz de atuar em um ambiente sem a intervenção direta de uma pessoa ou de outro agente.

Aprendizado: Um agente pode ser capaz de aprender e armazenar conhecimentos sobre o ambiente ao qual está inserido.

Mobilidade: Um agente pode mover-se de seu ambiente a outros ambientes caso seja necessário.

Colaboração: Um agente pode colaborar com outros agentes a fim de cumprir com seus objetivos.

Apesar de podermos ter todas essas características juntas em um agente, as propriedades de interação, adaptação e autonomia são consideradas como suficientes para classificar uma entidade de software como sendo um agente de software. Sendo a autonomia de um agente a propriedade fundamental que deve estar presente em um agente (OMG, 2000).

3.2 Classificação do Agente

Um agente pode ser classificado segundo sua arquitetura interna. A arquitetura definirá a forma como o agente tomará decisões em um ambiente. Algumas classificações para as arquiteturas de um agente são encontradas em (Russel e Norvig, 2004) e (Jennings e Woodridge, 1996) sendo que os trabalhos apresentam nomes diferentes para uma arquitetura. Girardi (2004) apresenta um agrupamento entre as classificações das arquiteturas que podemos ter para os agentes, são elas:

Arquitetura deliberativa: O agente possui um modelo simbólico do ambiente ao qual atua, sendo as decisões tomadas através de um processo de raciocínio lógico. O agente pode possuir um conjunto de metas e intenções e elaborar um plano de ação para alcançar um objetivo. Essa arquitetura pode ser encontrada com outras denominações,

tais como: Arquitetura baseada em lógica, arquitetura BDI,³ arquitetura intencional, arquitetura baseada em metas e arquitetura baseada em utilidade.

Arquitetura reativa: O agente não possui um modelo simbólico do ambiente, sendo suas decisões tomadas mediante o mapeamento direto de uma situação, que ocorre no ambiente, para a atuação neste ambiente. Essa arquitetura pode também ser encontrada com a denominação de arquitetura reflexiva.

Arquitetura em camada: Nesta arquitetura há a presença de componentes encontrados na arquitetura deliberativa e na arquitetura reativa, sendo que as decisões são tomadas em uma estrutura de camadas de decisão. Essa arquitetura também pode ser encontrada com a denominação de arquitetura híbrida.

3.3 Sistema multiagente

Um sistema multiagente (SMA) pode ser conceituado como um conjunto de agentes autônomos situados em um mesmo ambiente se comunicando e cooperando para atingir objetivos pré-estabelecidos (Shen et al., 2006). O SMA pode ser do tipo homogêneo, quando há vários agentes de um mesmo tipo ou heterogêneo, onde há agentes de diversos tipos. Quanto ao ambiente no qual os agentes estão inseridos, podem ser do tipo físico, quando o agente interage com o ambiente físico (ex. um robô atuando em um ambiente) ou virtual, onde os agentes interagem em um ambiente virtual (ex.: um SMA na internet) (Weyns et al., 2005).

A comunicação entre agentes é essencial para que os agentes possam interagir uns com os outros, podendo ser através do uso de linguagens apropriadas para agentes, como a FIPA-ACL (FIPA, 2000) que estabelece uma ontologia e uma arquitetura necessária para prover a comunicação entre agentes em um SMA. Como principal vantagem na utilização de uma linguagem apropriada para a comunicação entre agentes pode ser destacado a propriedade de interoperabilidade entre os sistemas, onde há a possibilidade de um agente se comunicar com outros agentes em outros ambientes ou até mesmo com outros tipos de sistemas, desde que este utilize a mesma linguagem para comunicação (Greenwood et al.,

³ BDI: forma pela qual um agente toma decisões com base em um conjunto de crenças, desejos e intenções, do termo em inglês *beliefs, desires, and intentions* (Wooldridge e Jennings, 2009).

2008). Outra linguagem que pode ser citada é a KQML, que também estabelece uma estrutura que possibilita a comunicação entre agentes (Sajja, 2008).

Embora a comunicação entre agentes permita a interação entre eles, também se faz necessário estabelecer um padrão para controlar a comunicação. As arquiteturas de modelos de comunicação estabelecem uma forma organizada para tornar possível a comunicação entre agentes. Em Ferreira e Girardi (2002), as arquiteturas de comunicação são agrupadas como mostrado a seguir:

Arquitetura quadro-negro: Os agentes compartilham um espaço de memória comum para a comunicação com outros agentes, solicitando e oferecendo serviços por meio de publicações de mensagens em tal espaço de memória.

Arquitetura de troca de mensagens: Os agentes comunicam-se de forma direta, através do conhecimento prévio da existência e dos serviços providos por outros agentes, enviando mensagens diretamente ao agente necessário.

Arquitetura Federativa: Há a presença de agentes especiais que ficam responsáveis pela troca de informações entre os agentes.

Além da comunicação, um modelo de coordenação entre os agentes se faz necessário para que haja organização em uma sociedade de agentes. Isso é possível através de uma estratégia de coordenação entre os agentes, especificando como deve ser o comportamento coletivo em um SMA. Muitos são os modelos organizacionais para os SMA, dentre eles podemos citar os modelos: hierárquico, *holarchies*, coalizão, times, sociedades, federações, mercados dentre outros (Horling e Lesser, 2005). Em Ferreira e Girardi (2002), os modelos organizacionais de SMA são referenciados como arquiteturas organizacionais, sendo classificadas como:

Arquitetura mestre-escravo: Os agentes são divididos em mestres, agentes responsáveis pela distribuição de tarefas, e escravos, agentes executores de tarefas.

Arquitetura de mecanismos de mercado: Todos os agentes estão no mesmo nível hierárquico e conhecem os serviços que podem ser fornecidos por outros agentes.

Para o desenvolvimento de um SMA podem ser utilizados frameworks de desenvolvimento, tais como: Jack (Jack, 2009), Jade (Bellifemine et al., 2008) e o FIPA-OS (FIPA-OS, 2009). Usando um framework para o desenvolvimento de um SMA, abstraímos a complexidade de desenvolvimento do ambiente para os agentes e funções de comunicação, que são predefinidas nos frameworks, tornando o desenvolvimento de uma aplicação de SMA menos complexa.

3.4 Importância do agente para a Engenharia de Software

O paradigma do agente é uma área de pesquisa relativamente nova, quando comparada com outras abordagens de pesquisa da Engenharia de Software, como a orientação a objetos, mas que conseguiu atingir um alto nível de reconhecimento internacional devido ao fornecimento de bases teóricas para abordar o desenvolvimento de aplicações do tipo distribuídas, descentralizadas e inseridas em ambientes abertos e dinâmicos. Dessa forma, o paradigma de agente torna mais fácil a abordagem de aplicações complexas (Zambonelli e Omicini, 2004), (Girardi, 2004).

A Engenharia de Software, em sua constante busca por meios que facilite o desenvolvimento de software, encontra no paradigma dos agentes algumas técnicas e conceitos que ajudam no processo de desenvolvimento de aplicações complexas. Jennings (1999) destaca algumas dessas técnicas da engenharia de software que são utilizadas pelo paradigma de agentes, que são:

Decomposição: A decomposição é a técnica de dividir um problema de grande complexidade em vários problemas menos complexos.

Abstração: É uma técnica que visa desconsiderar detalhes menos importantes e concentrar-se nos detalhes que realmente importam, simplificando um modelo de sistema.

Organização: Identifica e gerencia os relacionamentos entre os componentes usados para a resolução dos problemas.

Além dessas técnicas, os agentes contribuem através de um novo paradigma para o desenvolvimento de software baseado na decomposição de um sistema de software em agentes autônomos executores de tarefas, fornecendo novas técnicas e conceitos, tais como: orientação a objetivos, modelagem de papéis, arquiteturas exclusivas para a tomada de decisões, modelos de organização e modelos de comunicação. Técnicas e conceitos como estes podem facilitar o desenvolvimento de sistemas complexos. Isso faz com que a Engenharia de Software Baseada em Agentes tenha muita atenção por parte da comunidade de pesquisa em informática neste século XXI (Lind, 2001).

3.5 Agentes em sistemas industriais

O paradigma de agente já foi empregado com sucesso em aplicações industriais na execução de tarefas como (i) planejamento e controle da produção, onde os agentes são responsáveis pela alocação de recursos e matéria-prima obedecendo às necessidades de produção; (ii) na integração entre sistemas ERP e Sistemas industriais, onde agentes recebem e fornecem informações a um sistema ERP e controlam os recursos necessários à produção; (iii) no monitoramento de processos industriais, atuando no controle de processos críticos. Alguns exemplos desses sistemas foram apresentados na Seção 2.4, refletindo a importância sobre o uso do paradigma dos agentes em sistemas industriais complexos (Madejski, 2007).

A Figura 3.1 ilustra uma arquitetura de um sistema industrial baseado em agentes para a tomada de decisões sobre a produção. Objetivo deste sistema é a descentralização das tomadas de decisões, para o melhor aproveitamento da planta fabril com economia de recursos e tempo através da coordenação e das decisões entre os agentes em cada nível. As decisões partem dos agentes de mais alto nível, decisões de planejamento a nível global do sistema, para níveis mais baixos, que podem representar máquinas e processos da planta fabril e a negociação de matéria-prima para a produção (Bruccoleri et al., 2005).

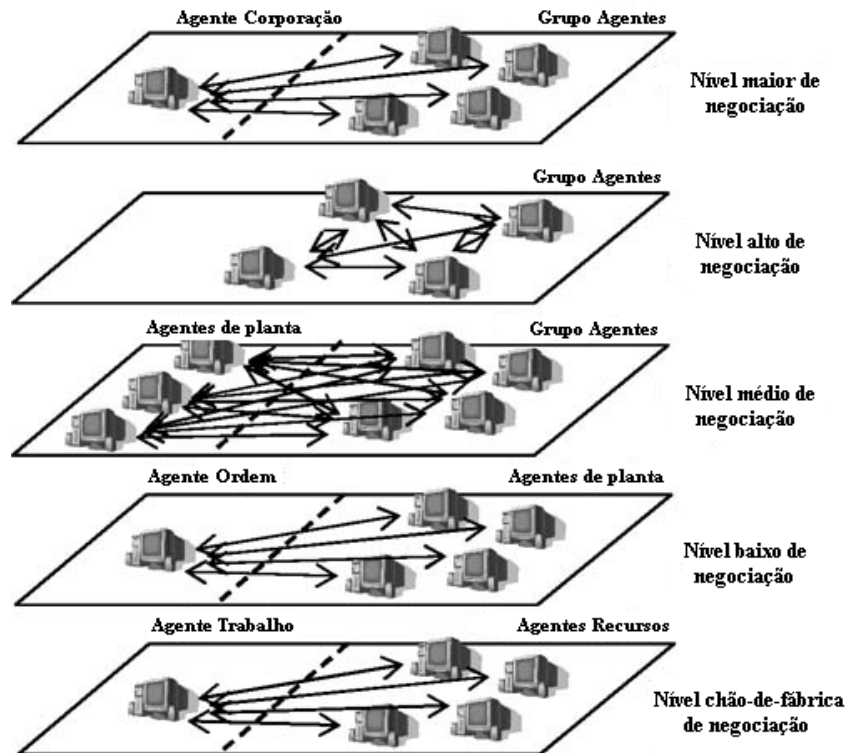


Figura 3.1: Arquitetura de decisão baseada em agentes em vários níveis para um sistema de produção (Bruccoleri et al., 2005).

Uma das formas de abordar o problema sobre o controle da manufatura é pelo paradigma da produção holônico, termo derivado da palavra inglesa “holons” que faz referência a uma unidade autônoma de produção e divide uma manufatura em unidades de holons autônomos e cooperativos (Lima, 2003). A teoria da produção *holônica* encontra no paradigma do agente o embasamento necessário para se tornar realidade, devido semelhanças entre ambos os conceitos, permitindo a concepção de sistemas de software baseados no modelo *holônico* de produção implementados através dos agentes (Giret e Botti, 2009).

Marick e McFarlane (2005) apontam alguns dos benefícios do uso do paradigma dos agentes em determinadas situações comuns no domínio industrial, são eles:

Conflitos: Os agentes tornam viáveis soluções de problemas complexos envolvendo conflitos através de técnicas de negociação e resolução de conflitos que podem ser adotadas;

Controle e Robustez: Uma solução com agentes torna possível a descentralização do controle de um sistema, uma vez que cada agente é autônomo, deixando a aplicação mais robusta em caso de falhas;

Flexibilidade: Os agentes podem permitir aplicações flexíveis e facilmente reconfiguráveis, uma vez que permitem a adição de novos agentes e novas funcionalidades a um sistema ou mesmo a re-configuração dinâmica do sistema;

Interoperabilidade: A interação de um sistema multiagente com outros sistemas, dentro ou fora de seu domínio, é possível pela adoção de uma linguagem padrão de agentes, tornando tais sistemas interoperáveis.

Encapsulamento: A propriedade de encapsulamento permite que processos industriais, sistemas ou máquinas possam ser representados por agentes.

Estes benefícios do uso do paradigma dos agentes fornecem o suporte necessário ao desenvolvimento de aplicações industriais modernas, por tratar de forma natural as novas exigências para os sistemas industriais modernos, apresentados na Seção 2.2.

3.6 Considerações finais sobre o capítulo

Este capítulo apresentou os conceitos e características do paradigma de agente. Conceituamos um agente, seus tipos e suas características assim como os sistemas multiagente (SMA), suas características e suas classificações.

Na Tabela 3.1 elencamos as propriedades que um SIM deve possuir e na mesma linha confrontamos com as características que um SMA pode possuir para atender a tais características. Dessa forma, podemos concluir a importância do paradigma de desenvolvimento dos agentes para contemplar os requisitos exigidos aos sistemas industriais modernos.

Propriedades do SIM Seção 2.2	Característica dos SMA Seção 3.5
Natureza distribuída	Sistema compostos de vários agentes em um ou vários ambientes.
Interoperabilidade	Propriedade de comunicação e interação entre os agentes.
Heterogeneidade	Propriedade de encapsulamento dos agentes.
Estruturas abertas e dinâmicas	Flexibilidade e reconfiguração dos SMA.
Tolerância a falhas	Agentes autônomos e distribuídos.

Tabela 3.1: Comparação entre as características desejadas aos SIM e as características dos SMA.

Marick e McFarlane (2005), Shen et al. (2006) e Monostori, Vancza e Kumara (2006) destacam o paradigma dos agentes como uma forte tendência frente a outros paradigmas para o desenvolvimento de aplicações industriais modernas por fornecer bases teóricas que atendem às exigências desses sistemas.

Com relação ao controle da manufatura, a teoria da produção holônica, que favorece a flexibilidade e adaptabilidade em uma linha de produção, tornando processos de produção autônomos, também pode ser abordada pelo paradigma dos agentes, permitindo que sistemas industriais possam adotar diversos modelos de configuração da manufatura.

Dessa forma, podemos concluir que os agentes se configuram como forte paradigma para abordar os diversos problemas inerentes aos sistemas industriais modernos.

CAPÍTULO 4

METODOLOGIAS

Neste capítulo, faremos um estudo sobre as metodologias para desenvolvimento de sistemas industriais com o objetivo de definirmos uma metodologia para ser usada na modelagem de uma aplicação industrial moderna usando agentes. Para isso, apresentamos os conceitos de metodologia e sua importância na Seção 4.1. Na Seção 4.2, é realizado um levantamento bibliográfico sobre as metodologias de desenvolvimento para sistemas industriais. Na seção 4.3 é feito um estudo das metodologias orientadas a agentes. Na seção 4.4, apresentamos as conclusões deste capítulo.

4.1 Conceito e importância de uma Metodologia

A engenharia de software preocupa-se em estudar assuntos relacionados com a produção de um software, desde os métodos, técnicas e ferramentas para desenvolvimento de um software a assuntos relacionados à sua produção, como: custos, recursos humanos, tempo, dentre outros. Os métodos, uma abordagem estruturada para o desenvolvimento de um software, as técnicas, uma forma de realizar uma determinada tarefa para o desenvolvimento do software, e as ferramentas, que fornecem o apoio automatizado para a produção de um software, são considerados como um conjunto fundamental para a construção de um software com qualidade (Pressman, 2002).

Um processo de desenvolvimento de software é um conjunto de procedimentos organizados de modo a permitir a construção de um produto de software que satisfaça uma série de objetivos e padrões. Um modelo de processo é uma descrição de uma série de atividades comum para um processo de software. Os modelos de processo de software considerados como clássicos são: Modelo Cascata, Modelo Espiral e o Modelo Incremental. Porém, existem outros modelos, como o Modelo em V (VModel, 2009) e o RUP (Rational, 2009), que são considerados como evoluções dos modelos clássicos. Seguir um modelo de software pode garantir a qualidade para a produção do software. Existem etapas definidas que devem ser seguidas, possibilitando transmitir conhecimentos sobre a produção do software, bem como a possibilidade de melhoria e adaptação do processo de software (Pfleeger, 2004).

Sommerville (2000) destaca que um processo de desenvolvimento de um software deve possuir quatro atividades básicas:

1. Especificação do software: Levantamento das funcionalidades e restrições que o software deve possuir;
2. Projeto e implementação de software: O processo de produção do software segundo as especificações;
3. Validação do software: Garantir que o software foi desenvolvido de acordo com as especificações dos clientes;
4. Evolução do software: o software deve ser propenso a evolução para atender as necessidades mutáveis do cliente.

Segundo Rezende (2005), uma metodologia é uma abordagem organizada para atingir um objetivo por meio de passos preestabelecido, como um roteiro, permitindo o uso de um ou vários métodos, técnicas e ferramentas. Ainda segundo o autor, as metodologias devem auxiliar no desenvolvimento do sistema de software para que os mesmos atendam, de maneira adequada, as necessidades dos clientes e/ou usuários com os recursos, prazos e pessoal disponível.

Para Wolf (2001), uma metodologia é importante por três motivos:

1. Para descrever o desenvolvimento de um sistema;
2. Para possibilitar o uso de ferramentas para auxílio no processo de desenvolvimento;
3. Para estabelecer uma comunicação comum para um projeto de software.

Grande parte das metodologias para desenvolvimento de software é baseada nos modelos de desenvolvimento de software clássicos. As metodologias denominadas de ágeis, como SCRUM e XP, possuem uma abordagem baseada em pessoas ao invés de processos, evitando o uso de muitos artefatos para o processo de modelagem do software (Soares, 2009).

4.2 Metodologias para sistemas industriais

Neste tópico, realizamos uma pesquisa bibliográfica sobre as metodologias para o desenvolvimento de sistemas industriais relatando trabalhos relevantes encontrados na área e

buscando extrair informações sobre os passos necessários para a modelagem de sistemas industriais.

Em Seo (1999), é apresentada uma metodologia para desenvolvimento de sistemas industriais em que a idéia principal é a virtualização da fábrica em um modelo computacional, para poder simular e testar o desempenho da aplicação. A metodologia é composta por quatro fases:

I - Fase de requisitos: Onde a manufatura é tratada como os requisitos funcionais que o sistema deve atender.

II - Fase de análise: São selecionados os hardwares que melhor atenderão aos requisitos a serem implementados.

III - Fase de detalhamento: Os requisitos são modelados através de linguagem formal para simulações em modelos virtuais de hardwares que poderão compor uma solução.

IV - Fase de validação e implantação: Nesta fase, ocorrem as simulações do modelo virtual para a validação das funcionalidades do sistema e verificação do desempenho da solução. Posteriormente, a solução está pronta para ser implantada.

Em (Souza, 2006), uma metodologia denominada de MARPASI (Metodologia de Aplicação das Redes de Petri em Automação Industrial) foi apresentada para o desenvolvimento de aplicações industriais. Na metodologia, temos o uso das Redes de Petri na modelagem e simulação do sistema de controle industrial, gerando código fonte para os controladores do tipo CLP na linguagem Ladder, linguagem padronizada comumente usada para programação de CLP (Manesis e Akantziotis, 2005). A metodologia é composta por duas fases, projeto e execução. Na fase de projeto, temos as seguintes atividades: definição do sistema, análise, especificação e validação dos requisitos, modelagem dos requisitos por Rede de Petri, conversão de Rede de Petri para linguagem Ladder. Na fase de execução da metodologia, temos as atividades de implementação e testes do sistema.

Em Carpanzano e Jovane (2007), é apresentada uma metodologia para o desenvolvimento de sistemas industriais baseado em uma arquitetura hierárquica pré-definida, com a implementação das funcionalidades do sistema em blocos de funções. A metodologia é

baseada no modelo de desenvolvimento de software incremental⁴, sendo composta pelos cinco passos, descritos a seguir:

I – Definição do sistema: Descrição do processo a ser automatizado através da descrição de atividades que o sistema deverá executar e a descrição dos objetivos do sistema.

II – Especificação do sistema: As tarefas e funções necessárias para a execução do sistema são agrupadas segundo uma arquitetura em blocos de funções. Os blocos são compostos de eventos de entradas, dados de entradas, funcionalidades, eventos de saída e dados de saída.

III – Projeto do sistema de automação: O sistema de controle e supervisão é modelado usando notação formal e seguindo a norma IEC-61499, norma que fornece a base teórica para aplicações distribuídas usando blocos de funções (Thramboulidis, 2005).

IV – Integração e teste da automação: Os códigos modelados formalmente são testados por simulação e posteriormente inseridos aos controladores, obedecendo à arquitetura da aplicação formada por blocos de funções definidos.

V – Integração e teste do sistema: A integração é realizada ligando fisicamente os blocos a outros hardwares do sistema. Posteriormente, os testes são executados para avaliação do comportamento da aplicação.

Uma metodologia ágil para sistema industrial é apresentada no trabalho de Aiello et al. (2007), onde uma técnica para a geração automática do código da aplicação para CLP é descrita baseada na norma IEC 61131-3, norma da ISO (*International Organization for Standardization*) que estabelece as diretrizes para a programação de controladores do tipo CLP (Tisserant et al., 2007). A metodologia consiste no uso dos conceitos presentes no desenvolvimento ágil de sistemas só que adaptados ao domínio industrial. São usadas técnicas como o desenvolvimento incremental do sistema, simulação e testes mediante a construção de

⁴ Desenvolvimento incremental de um sistema é a técnica onde o sistema é desenvolvido e testado por partes, evoluindo em forma de novos incrementos com novas funcionalidades (Sommerville, 2000).

um framework para a aplicação, que servirá como a base para o desenvolvimento e teste do sistema. A metodologia simula e gera códigos automáticos para uma CLP.

Em Moura e Guedes (2007), é apresentada uma metodologia para desenvolvimento de sistemas de controle industrial baseada na UML⁵ utilizando-se de diagramas de máquinas de estado e cenários definidos através de uma sequência de eventos. A metodologia consiste em modelar os elementos físicos da aplicação, sensores e atuadores, decompondo os estados possíveis de cada elemento e criando cenários baseados em uma sequência de eventos possíveis, mostrando o funcionamento do sistema através desta modelagem.

4.2.1 Análise das metodologias para sistemas industriais

Comparar essas metodologias estudadas nem sempre é uma tarefa fácil, devido às particularidades de cada uma delas. Para analisarmos e entendermos requisitos essenciais para construirmos uma aplicação industrial, analisamos tais metodologias segundo as fases de desenvolvimento que cada uma delas contemplam, os modelos sugeridos como artefatos e as características particulares. A Tabela 4.1 apresenta uma análise comparativa entre as metodologias estudadas na Seção 4.2, segundo os critérios estabelecidos para a análise.

Analisando as características da Tabela 4.1 sobre as metodologias estudadas, é possível perceber que todas demonstram claramente a preocupação em descrever e, de certa forma validar, o comportamento do sistema, pelo menos no tocante do sistema de automação. Também é possível perceber a preocupação em descrever a arquitetura da solução de automação, onde as funcionalidades do sistema serão implantadas.

⁵ UML linguagem para modelagem de sistemas orientados a objetos, termo do inglês *Unified Modeling Language* (Fowler, 2005).

Metodologia	Fases Contempladas	Modelos Sugeridos	Características Particulares
Seo (1999)	Requisitos Arquitetura Formalização Validação Implantação	-	Simulação virtual para a arquitetura de hardware e validação da aplicação através de simulações virtuais.
Souza (2006)	Requisitos Análise Formalização Validação Implementação Teste	Formalização por Redes de Petri	Geração automática do código fonte para os controladores.
Carpanzano e Jovane (2007)	Definição Especificação Projeto Implementação Testes	Diagrama de blocos de funções	Preocupa-se com as funcionalidades do sistema a ser implementado na fase de automação.
Diego et al. (2007)	Requisitos Implementação testes	-	Metodologia ágil simula as características da aplicação.
Moura e Guedes (2007)	Requisitos Arquitetura Validação	Diagramas UML	Valida o comportamento da aplicação através de modelos.

Tabela 4.1: Comparação entre as metodologias de sistemas industriais.

A preocupação com o comportamento do sistema faz com que as metodologias modelem o comportamento, através do formalismo ou através de simulações, validando-o ainda na fase de modelagem.

Sommerville (2000) e Pressman (2002) concordam quanto ao fato de que os modelos formais permitem a especificação de um sistema de software sem ambigüidade e inconsistência. Porém, há problemas com essa utilização, pois pode tornar, em alguns casos, o desenvolvimento do software lento e dispendioso, além do fato que o modelo gerado pode ser difícil de ser utilizado como mecanismo de comunicação.

4.3 Metodologias para Sistemas Multiagente

Uma metodologia de desenvolvimento de software orientado a agentes deve fornecer métodos para o desenvolvimento desses sistemas, apoiado em um ciclo de vida de desenvolvimento do software (Giorgini e Henderson-Sellers, 2005).

A metodologia deve fornecer métodos essenciais a um sistema orientado a agentes, tais como: estratégia de tomada de decisão do agente, forma de coordenação, comunicação e organização de um sistema multiagente (Gómez-sanz e Pavóm, 2004).

As metodologias para desenvolvimento de SMA podem originarem-se da orientação a objetos, tais como: MaSe (Deloach, Wood e Sparkman, 2001), Gaia (Zambonelli, Jennings e Wooldridge, 2003), Prometheus (Padghan e Winiko, 2002), MESSAGE (Caire et al., 2001) e ADELFE (Bernon et al., 2002), sendo que MESSAGE e ADELFE são derivações da metodologia RUP. Há também metodologias que são provenientes da engenharia de conhecimento, como a MASCommomCADs (Iglesias et al., 2005), e técnicas para a modelagem dos agentes, como a AUML (Odell et al., 2000).

Diante de tantas metodologias, para escolhermos uma delas, optamos por usar critérios como: facilidade de compreensão da metodologia, material disponível para consulta e exemplos de uso no domínio de sistemas industriais e, principalmente, a maturidade da metodologia, ou seja, o tempo de vida delas. Levando em consideração tais critérios, escolhemos as metodologias: Mase, Gaia e Prometheus, para um estudo mais detalhado sobre as fases do desenvolvimento de um SMA que estas contemplam, para que dessa forma possamos compará-las e escolher uma para ser utilizada na modelagem de um sistema industrial.

Ao detalharmos as três metodologias que escolhemos, podemos observar melhor como cada uma delas trata o desenvolvimento de um sistema orientado a agentes. Isso é importante por permitir encontrar semelhanças entre o desenvolvimento de um sistema orientado a agentes e um sistema industrial, justificando a escolha da metodologia Prometheus para a modelagem do estudo de caso deste trabalho.

4.3.1 Metodologia Mase

A metodologia Mase (*Multiagent Systems Engineering Methodology*) (Deloach, Wood e Sparkman, 2001) é uma metodologia para a construção de sistemas multiagentes que fornece uma técnica baseada em objetivos e tarefas para a coordenação do sistema. A metodologia não especifica uma arquitetura para os agentes, apenas pressupõem que os agentes sejam guiados por seus objetivos.

A metodologia deve ser iniciada após a fase de requisitos do sistema, porém sugere que os requisitos estejam modelados em um diagrama de casos de uso da UML. Todas as etapas da modelagem podem ser realizadas com o auxílio da ferramenta AgentTools

(DeLoach, 2001), que além de gerar os artefatos gráficos pode gerar o código implementável para o sistema na linguagem Java. A metodologia Mase possui duas fases: análise e projeto, que descrevemos a seguir. A Figura 4.1 detalha as fases da metodologia Mase.

Na fase de análise, temos as seguintes etapas a serem seguidas:

Captura dos objetivos: Ao analisar os requisitos do sistema, o sistema deve ser decomposto em objetivos, que serão modelados segundo uma hierarquia de objetivos.

Aplicação dos casos de uso: Os casos de uso são modelados em um diagrama de sequência, para permitir que sejam identificados papéis a serem desempenhados no sistema.

Refinamento dos papéis: Os papéis a serem desempenhados no sistema são refinados e geram uma ou mais tarefas, que podem ser concorrentes. Os papéis serão compostos pelos objetivos do sistema e a ligação entre os papéis representa o processo de comunicação mínima que o sistema deverá implementar.

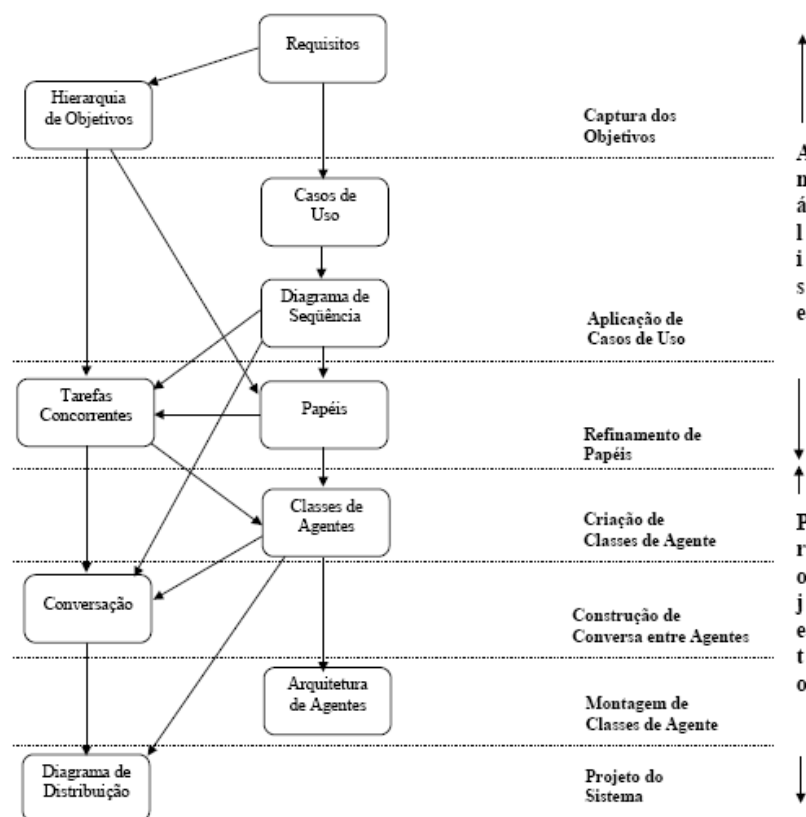


Figura 4.1: Fases da metodologia Mase (DeLoach, 2001)

Na fase de projeto, temos as seguintes etapas a serem seguidas:

Criação das classes de agentes: Cada agente será relacionado a um ou mais papéis, formando a estrutura mínima que o agente deverá implementar sendo representadas por uma classe e suas interações.

Construção das conversas entre agentes: A modelagem da comunicação entre agentes são descritos nesta fase com base nas interações entre as classes de agentes.

Montagem da classe de agente: Nesta etapa é definida a arquitetura interna do agente, ao qual um agente é formado por uma ou mais classes, não definindo o tipo da arquitetura.

Projeto do sistema: As classes são modeladas, instanciadas e distribuída compondo o projeto da arquitetura do sistema como um todo.

A metodologia Mase evoluiu para a O-Mase (Ojeda et al., 2008), Acrescentando a fase de requisitos e o uso de ontologias na modelagem do domínio da aplicação.

4.3.2 Metodologia Prometheus

A metodologia Prometheus (Padghan e Winiko, 2002) é considerada pelos autores prática, completa e detalhada, voltada para profissionais da área de desenvolvimento de sistemas e estudantes que não necessariamente possuam maiores conhecimentos sobre agentes. Os agentes baseiam-se na arquitetura deliberativa do tipo BDI. A metodologia Prometheus é baseada na metodologia RUP, onde se aplica o conceito de processo iterativo entre as fases da metodologia. As fases da metodologia são: especificação do sistema, projeto da arquitetura e projeto detalhado. A Figura 4.2 ilustra as fases da metodologia Prometheus.

A fase de especificação do sistema tem como objetivo descrever o modelo do ambiente do sistema, sendo composta pelas seguintes atividades:

Objetivos do sistema: O sistema é descrito como um conjunto de objetivos a serem atingidos.

Funcionalidades: São definidas as funcionalidades básicas que o sistema deverá atender a nível de entradas (percepções) e saídas (ações).

Cenários: São usados para a descrição detalhada das funcionalidades do sistema e podem ser baseados nos casos de uso do sistema.

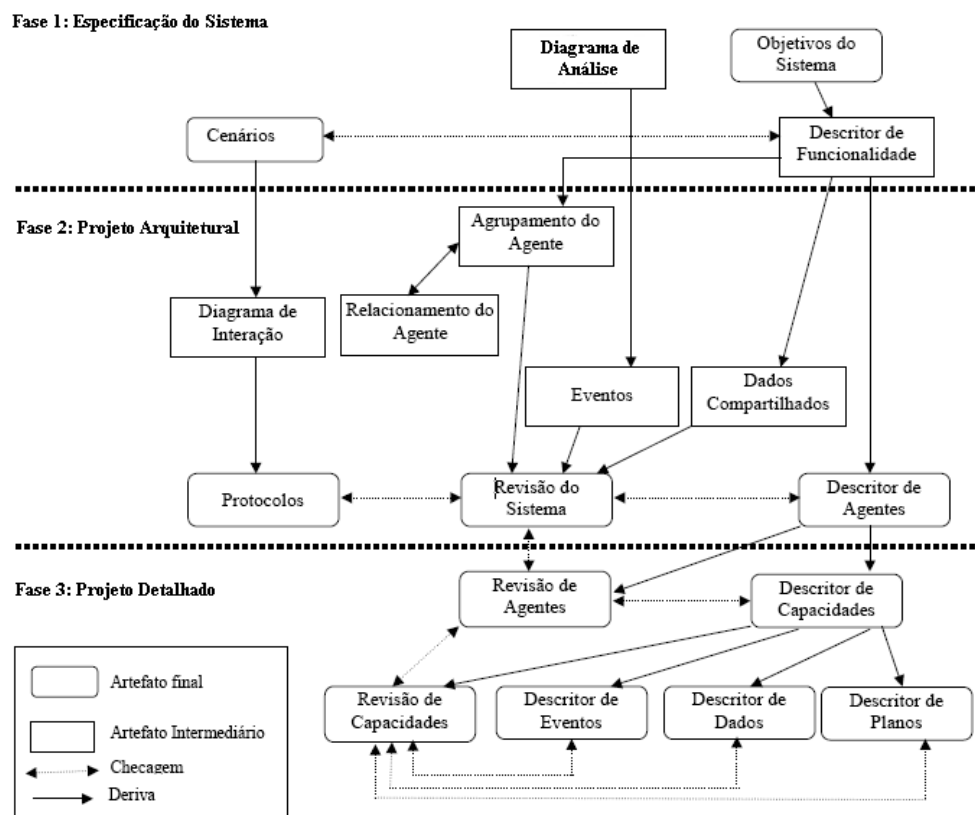


Figura 4.2: Fases da metodologia Prometheus (Padghan e Winiko, 2002).

A fase de projeto de arquitetura determina os tipos de agentes e suas interações, sendo composta pelas seguintes atividades:

Descrição dos agentes: Os agentes são identificados mediante a análise das funcionalidades, onde as mesmas serão agrupadas por relacionamento

(funcionalidades que compartilham os mesmos dados) ou por ligação (onde existem relações significativas entre as funcionalidades).

Descrição das interações: As interações necessárias entre os agentes para que os mesmos cumpram com seus objetivos são modelados no diagrama de relacionamento de agentes, bem como são descritos os protocolos de interação dos agentes.

Revisão do Sistema: A estrutura estática do sistema é mostrada através do diagrama de revisão do sistema, que é um resumo do sistema contendo os agentes, as interações entre os agentes, ações e percepções e os dados.

Na fase de projeto detalhado, é desenvolvida a estrutura interna de cada agente sendo composta das seguintes atividades:

Diagrama de revisão dos agentes: Um artefato que fornece a visão de alto nível da estrutura interna do agente, apresentando a arquitetura interna de cada agente.

Diagrama de revisão de capacidades: Descreve a estrutura interna de um agente em termos de planos, capacidade do agente, dados recebidos ou enviado e eventos.

Descritor de planos: Artefato que contém uma lista de planos e uma avaliação de sucesso para tais planos.

Descritor de dados: Descreve os dados necessários ao agente, sendo estes provenientes de outros agentes (mensagens) ou de percepções.

Descritor de eventos: Especifica todos os eventos e suas finalidades bem como os dados utilizados pelos eventos.

A metodologia é suportada por duas ferramentas, a Jack (Jack, 2009) e a Prometheus Tools (RMIT, 2009).

4.3.3 Metodologia Gaia

Gaia (Zambonelli, Jennings e Woodridge, 2003) é uma metodologia fundamentada na visão de um sistema como uma organização computacional com vários papéis a serem desempenhados. A metodologia trata de aspectos nos níveis macro (sociedade de agentes) e micro dos agentes.

A metodologia é composta de duas fases: análise e projeto, na qual a fase de projeto é subdividida entre projeto da arquitetura e projeto detalhado. As fases da metodologia GAIA são apresentadas na Figura 4.3.

A fase de análise da metodologia mostra o entendimento do sistema como uma organização e sua decomposição em sub-organizações e papéis a serem desempenhados. As etapas a serem seguidas são descritas a seguir:

Divisão do sistema em sub-organizações: O sistema deve ser decomposto de modo formar organizações, blocos com funcionalidades semelhantes.

Modelo do ambiente: Descrição em alto nível dos ambientes nos quais os agentes serão inseridos, obedecendo a decomposição do sistema em sub-organizações.

Papéis da Organização: Identifica os papéis existentes no sistema. Um papel pode ser visto como uma descrição abstrata da funcionalidade esperada de uma entidade.

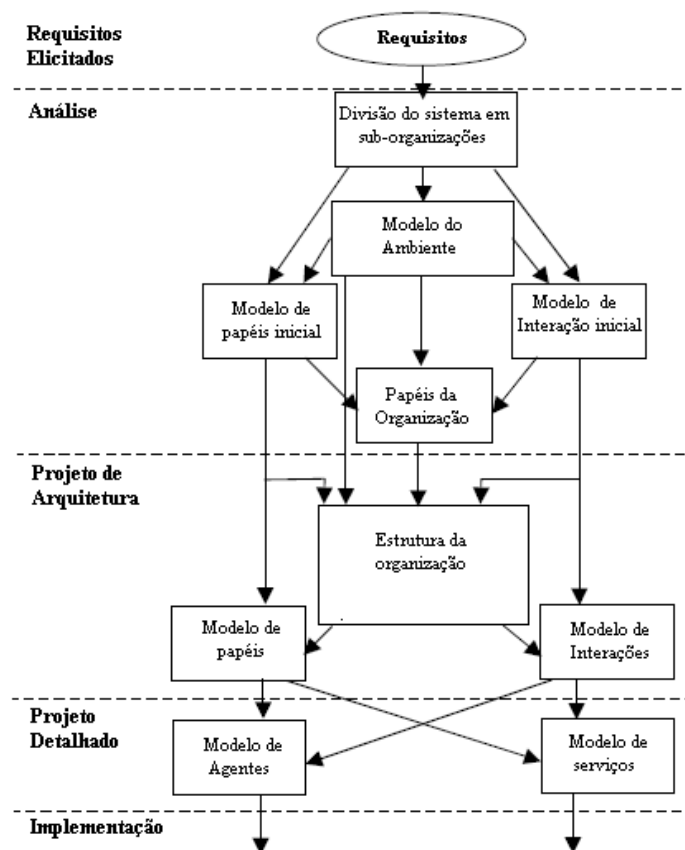


Figura 4.3: Fases da metodologia GAIA (Zambonelli, Jennings e Woodridge, 2003).

Na fase de projeto de arquitetura temos a modelagem da organização da sociedade de agentes, sendo suas principais etapas descritas a seguir:

Estrutura da organização: Apresenta um modelo estrutural da organização ao nível de agrupamento de papéis, sendo este definido por quatro atributos: Protocolos (define a interação entre os papéis), Permissões (Direitos associados a um papel), Atividades (Define as tarefas por ele executadas) e Responsabilidades (determinam as funcionalidades).

Modelagem das interações entre papéis: Consiste em um conjunto de definições de protocolos, um para cada interação entre os papéis.

Modelo de papéis: A definição final dos papéis que deverão ser desempenhados pelo sistema, derivado do modelo de papéis iniciais do sistema.

Na fase de projeto detalhado, há a preocupação com o desenvolvimento do agente, sendo compostas das etapas são descritas a seguir:

Modelo de Agente: Definirá os tipos de agentes que irão compor o sistema e suas instâncias.

Modelo de Serviço: Define os principais serviços que serão atribuídos a cada tipo de agente.

A metodologia GAIA não possui uma ferramenta para auxílio às fases da metodologia.

4.3.4 Análise das metodologias de Sistemas Multiagente

As metodologias SMA detalhadas neste trabalho compartilham as características de fornecer uma estratégia para a modelagem da solução, a partir dos requisitos previamente elicitados, em uma solução baseada em agentes. Todas elas compartilham a visão de que o agente de software é uma entidade que cumpre papéis no sistema, papéis esses necessários para que eles atinjam os objetivos do sistema.

Porém, a forma como essas metodologias modelam a solução, as fases e os artefatos propostos são diferentes. Isso torna difícil a escolha de parâmetros para comparação entre elas. Levaremos em consideração as características inerentes ao paradigma dos agentes, conforme apresentadas no Capítulo 3 deste trabalho, para escolhermos os critérios de avaliação para as metodologias. São eles:

Modelagem do ambiente: A forma como a metodologia lida com o ambiente no qual o agente estará inserido.

Modelagem da arquitetura do agente: Arquitetura influencia diretamente na estrutura do agente, determinando o tipo do agente e seu comportamento diante de um objetivo.

Modelagem da comunicação entre agentes: A estratégia da metodologia para prover a interação entre os agentes.

Modelagem da coordenação entre agentes: A forma como a metodologia define mecanismos para o controle das interações entre agentes.

A partir desses parâmetros, a Tabela 4.2 apresenta os resultados da comparação entre as metodologias.

Metodologia:	Mase	Prometheus	Gaia
Modelagem do ambiente	Modela através de cenários de caso de uso	Modela o ambiente no nível de percepções e ações	Modela o ambiente como processos organizacionais
Modelagem da arquitetura do agente	Não define	BDI	BDI
Modelagem da comunicação entre agentes	Troca de mensagem através da definição de protocolos	Troca de mensagem através da definição de protocolos	Troca de mensagem através da definição de protocolos
Modelagem da coordenação entre agentes	Todos os agentes conhecem os agentes ao qual devam interagir	Todos os agentes conhecem os agentes ao qual devam interagir	Todos os agentes conhecem os agentes ao qual devam interagir

Tabela 4.2: Comparação entre as metodologias de SMA.

Analisando a Tabela 4.2, encontramos semelhanças entre as metodologias, no que diz respeito à modelagem da comunicação e a modelagem da coordenação entre os agentes. Na modelagem da arquitetura do agente, apenas a Mase difere das demais. Porém, na modelagem do ambiente há estratégias diferentes entre as metodologias.

Considerando a característica da modelagem do ambiente na metodologia Prometheus, onde percebemos que o ambiente é modelado segundo as percepções e ações. E considerando o ambiente industrial, onde podemos ter no ambiente a presença de sensores (que percebem o ambiente) e atuadores (que atuam no ambiente) nos processos industriais, a metodologia Prometheus foi escolhida para a modelagem de um sistema industrial moderno devido a essa similaridade da modelagem com o ambiente industrial.

4.4 Considerações finais do capítulo

Neste Capítulo, conceituamos e abordamos a importância das metodologias para a modelagem de sistemas, apresentando algumas metodologias para desenvolvimento de sistemas industriais encontradas em trabalhos recentes e as metodologias para desenvolvimento de sistemas multiagente, escolhendo três delas para uma avaliação.

Podemos concluir que as metodologias apresentadas para o desenvolvimento de sistemas industriais se preocupam com questões referentes à automação, como a arquitetura da solução e o comportamento da aplicação. Já as metodologias de desenvolvimento de sistemas multiagentes se preocupam em transformar as especificações do sistema em soluções baseadas em agentes.

Uma vez concluído no Capítulo 3 que o paradigma dos agentes satisfaz os requisitos para o desenvolvimento de uma aplicação industrial moderna e, levando em consideração que as metodologias para sistemas industriais, estudadas na Seção 4.2 deste trabalho, não contemplam o desenvolvimento orientado a agentes, escolhemos trabalhar com a metodologia Prometheus, conforme justificamos na Seção 4.3.

Porém, se por um lado as metodologias de sistemas industriais, apresentadas na Seção 4.2 deste trabalho, dão maior ênfase à construção de sistemas de automação industrial e as metodologias de sistemas multiagentes, discutidas na Seção 4.3, tratam do desenvolvimento de um sistema através do paradigma do agente e revendo o conceito de sistemas industrial, no Capítulo 2, que diz que um sistema industrial pode ser composto de um ou mais sistemas de automação, podemos concluir que as metodologias de SMA são incompletas para a abordagem de todos os passos necessários para o desenvolvimento de um sistema industrial moderno por não contemplar a fase da automação industrial.

Dessa forma, usamos uma metodologia de SMA, a Prometheus, para modelar a solução do sistema, nos beneficiando do paradigma dos agentes para contemplar os requisitos exigidos para o desenvolvimento de um sistema industrial moderno, conforme podemos concluir na Seção 3.6, adicionando à metodologia uma extensão, derivada de características encontradas nas metodologias de desenvolvimento de sistemas industriais, Seção 4.2.1, contemplando as exigências de modelagem dos sistemas de automação industrial.

CAPÍTULO 5

EXTENSÃO DA METODOLOGIA PROMETHEUS E ESTUDO DE CASO

Neste capítulo, propomos uma extensão para a metodologia Prometheus com o objetivo de que a mesma possa contemplar todas as fases do desenvolvimento de uma aplicação industrial. Apresentamos a proposta de extensão na Seção 5.1. Na Seção 5.2 apresentamos o estudo de caso para validar a extensão da metodologia Prometheus e o uso da metodologia na modelagem do estudo de caso. Na Seção 5.3 apresentamos as conclusões sobre este capítulo.

5.1 Descrição da extensão

Um sistema industrial moderno pode ser composto de duas fases distintas, a fase de automação e controle de processos e a fase de desenvolvimento de um modelo de gerenciamento do controle de processo, sendo que ambas as fases envolvem tecnologias e conhecimentos específicos a cada uma delas (Quingi, 2006).

A parte de automação do processo exigirá conhecimentos em tecnologias específicas, como programação de CLP e conhecimentos sobre tecnologias de sensores, atuadores e comunicação entre esses componentes. Tais conhecimentos são comuns aos profissionais da área de engenharia elétrica e mecatrônica. Enquanto que na parte de gerenciamento do controle de processo podemos ter o uso de uma plataforma do tipo PC, envolvendo conhecimentos em assuntos como: banco de dados, protocolos de comunicação e plataformas de desenvolvimento de sistemas, tais conhecimentos são comuns aos profissionais da área de sistemas de informação (Kumar et al., 2004).

Também deve ser considerado o fato de que um dispositivo tipo CLP pode ser uma estação de automação inteligente (Da'na et al., 2008). Dessa forma, seu uso propicia criar projetos de aplicações distribuídas, onde as funcionalidades podem ser implementadas em vários dispositivos interconectados. Outro ponto importante seria considerar um equipamento do tipo *Key-on-hand* na automação de um processo, onde o mesmo pode ser considerado uma estação de automação inteligente.

A metodologia Prometheus contempla as fases necessárias para o desenvolvimento de um sistema multiagente. A fase da automação e controle de processos industriais não é contemplada, embora seja essencial para suprir os requisitos de automação. Porém, pode haver casos em que a automação é modelada na Metodologia Prometheus como uma entidade do sistema, ou seja, um ator. No entanto, mesmo assim não seria possível modelar os detalhes desta automação. Por este motivo, propomos uma extensão para a metodologia Prometheus, conforme a Figura 5.1.

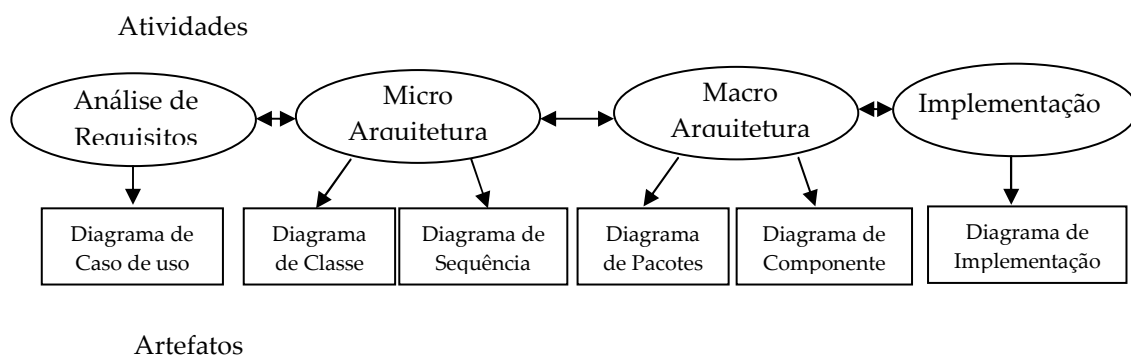


Figura 5.1 Extensão da metodologia Prometheus

Dessa forma, a extensão tem como objetivo principal a modelagem da solução da automação industrial, contemplando desde os requisitos a ser implementado ao projeto final de implantação do sistema.

A Figura 5.2 destaca as fases da extensão proposta e suas ligações com a metodologia Prometheus, onde a partir de uma análise da etapa de descrição dos agentes em conjunto com as funcionalidades que tal agente deverá implementar poderá ser percebido a necessidade do desenvolvimento de um sistema de automação. A ligação entre a extensão proposta e a metodologia será detalhada durante o estudo de caso.

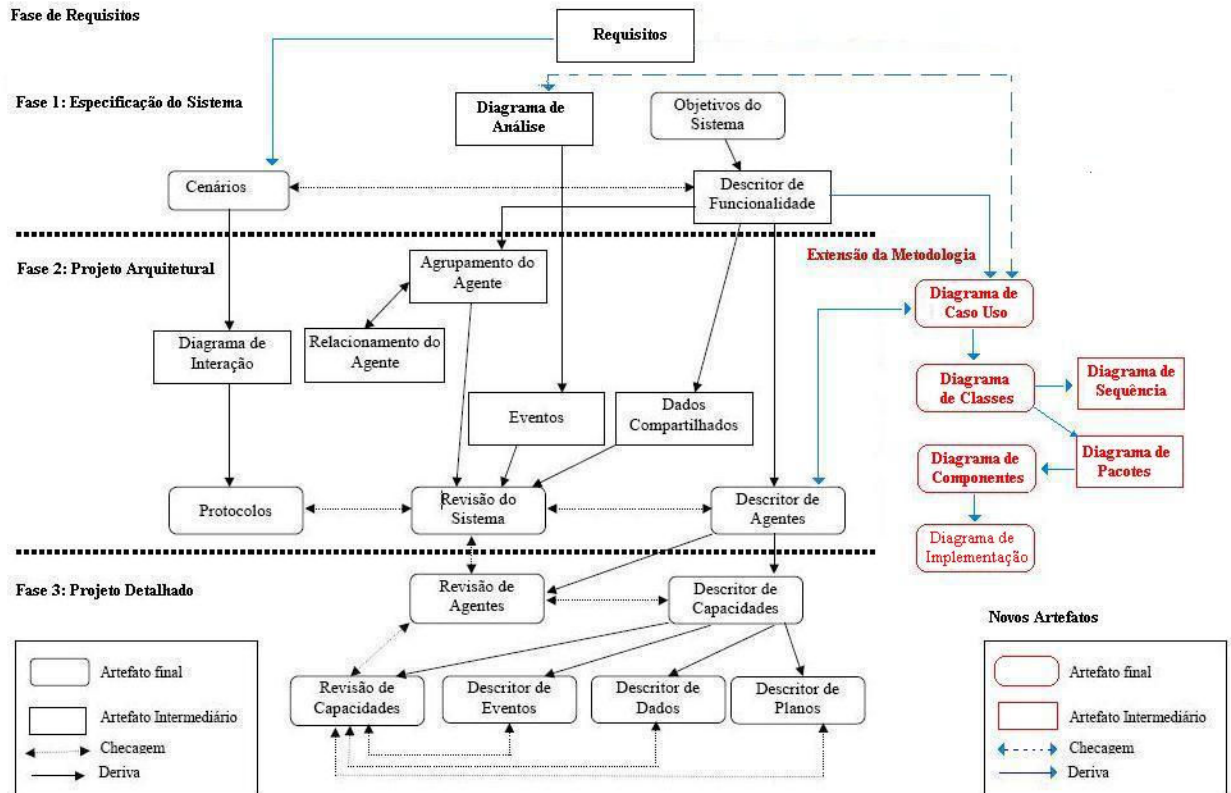


Figura 5.2: Adicionando a extensão a metodologia Prometheus.

Podemos perceber na Figura 5.2 a adição dos requisitos como uma fase inicial a metodologia. Porém tal fase é necessária e defendida pelos autores (Padgham e Winiko, 2002) que, apesar de não incluir esta fase na especificação de sua metodologia, defende que a mesma é essencial para iniciar a modelagem. Devido a este fato, colocamos a fase de requisitos no início da modelagem, buscando a ênfase na importância desta para o desenvolvimento de um sistema.

A extensão constitui-se de atividades que geram artefatos necessários ao melhor entendimento do sistema. Para os artefatos, escolhemos o uso da UML (*Unified Modeling Language*) e suas notações gráficas para a modelagem nesta extensão, devido à mesma ser considerada como padrão e ser bastante difundida na comunidade de desenvolvimento de software de um modo geral (Fowler, 2005). As atividades da extensão são descritas nas subseções abaixo.

É importante considerarmos que a extensão da metodologia proposta neste trabalho é essencial para a modelagem da solução da automação em um sistema industrial. Isso não quer dizer que no processo de modelagem de um sistema industrial a metodologia Prometheus não trata a questão do sistema de automação industrial e sim que a metodologia Prometheus especifica o sistema de automação industrial. Porém, não fornece suporte para a modelagem deste sistema de automação necessário ao sistema industrial modelado, conforme ficará claro no estudo de caso.

5.1.1 *Análise de Requisitos*

Nesta primeira etapa deve-se analisar os diagramas de funcionalidades e o diagrama de análise gerados na primeira fase da metodologia Prometheus, conforme pode ser visto na Figura 5.2, buscando identificar as funcionalidades que estão relacionadas à necessidade de desenvolvimento da automação de um processo industrial. Tais funcionalidades apontarão de alguma forma (direta ou indiretamente) para um processo específico em uma operação de automação.

Além das funcionalidades atribuídas a automação, em alguns casos, uma entidade externa modelada no diagrama de análise pode estar representando uma máquina ou um processo, sendo necessário então modelar a solução que represente tais entidades com base nas funcionalidades requeridas pelo sistema.

O Diagrama de Caso de Uso deverá ser utilizado para especificar os requisitos que o sistema de automação deverá contemplar para atender as necessidades do sistema. Lembrando que, estes requisitos estarão, algumas vezes, implícitos em uma descrição de uma funcionalidade e outras vezes já serão identificados diretamente pelo diagrama de análise da metodologia Prometheus.

5.1.2 *Micro Arquitetura*

Esta etapa compreende a identificação da estrutura necessária para atender às funcionalidades que desejamos implementar no sistema de automação. A técnica consiste em modelar a solução de automação, através da modelagem dos componentes que irão compor o sistema, como classes. Estas classes, representando os dispositivos e suas funcionalidades. A Figura 5.3 exemplifica tal método modelando um sensor de temperatura como uma classe.



Figura 5.3: Modelando objetos da automação como classes.

As interações entre as classes, necessárias para que se cumpram com as funcionalidades desejadas, são modeladas em um diagrama de classes da UML. Dessa forma, representando a arquitetura estática da solução. A Figura 5.4 apresenta um exemplo de um diagrama de classes para um controlador de temperatura.

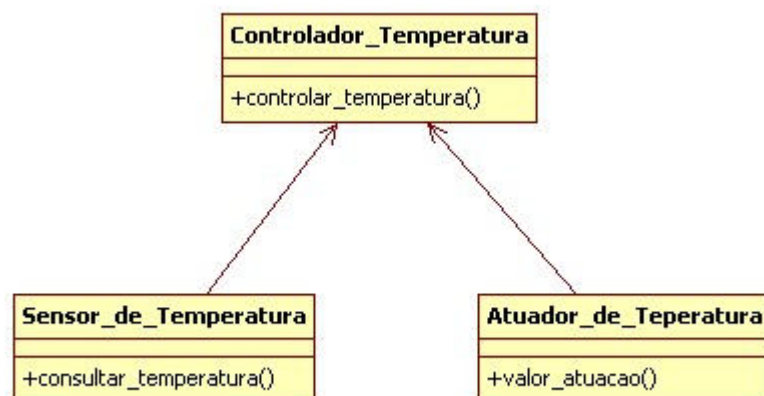


Figura 5.4: Diagrama de classes de um controlador de temperatura.

Para modelarmos as interações entre as classes de forma dinâmica utilizamos o diagrama de sequência, que mostrará a troca de mensagens entre os objetos para que os mesmos possam cumprir com suas funcionalidades. A Figura 5.5 ilustra um diagrama de sequência da classe de um controlador de temperatura solicitando uma informação sobre uma temperatura em um dado instante a classe sensor de temperatura.

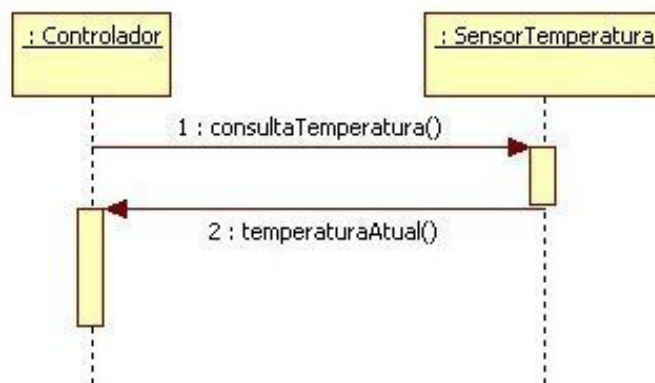


Figura 5.5: Diagrama de sequência do controlador de temperatura.

5.1.3 Macro Arquitetura

Esta etapa de macro arquitetura consiste no agrupamento de um conjunto de classes que possuem semelhanças para compor um pacote. A Figura 5.6 apresenta o processo de criação de um pacote, onde uma ou várias classes podem ser agrupadas em pacotes para prover as funcionalidades necessárias.

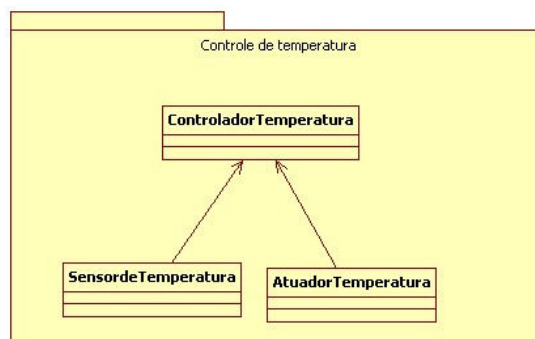


Figura 5.6: Criando pacotes a partir de classes.

Um componente pode conter um ou mais pacotes e representar um determinado equipamento físico que fornece a estrutura necessária para que o sistema forneça as funcionalidades desejadas.

Na Figura 5.7 ilustramos o processo de criação de componentes a partir de pacotes, onde um componente pode implementar um ou mais pacotes de funcionalidades. Os componentes por sua vez podem representar desde componentes para a automação de processos (hardwares da automação) a uma ou várias máquinas que interagem em um processo.

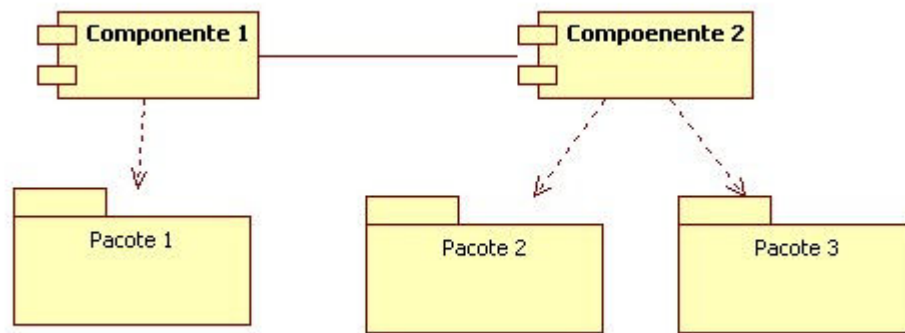


Figura 5.7: Criando componentes a partir de pacotes

5.1.4 Implementação

Nesta etapa, temos a modelagem do diagrama de implantação, apresentando uma visão de como deve ser a estrutura do sistema de automação, a arquitetura da solução. Esta pode ser composta por vários hardwares e de subsistemas da aplicação, representados pelos componentes e a comunicação entre estes. A Figura 5.8 exemplifica um diagrama de implantação.

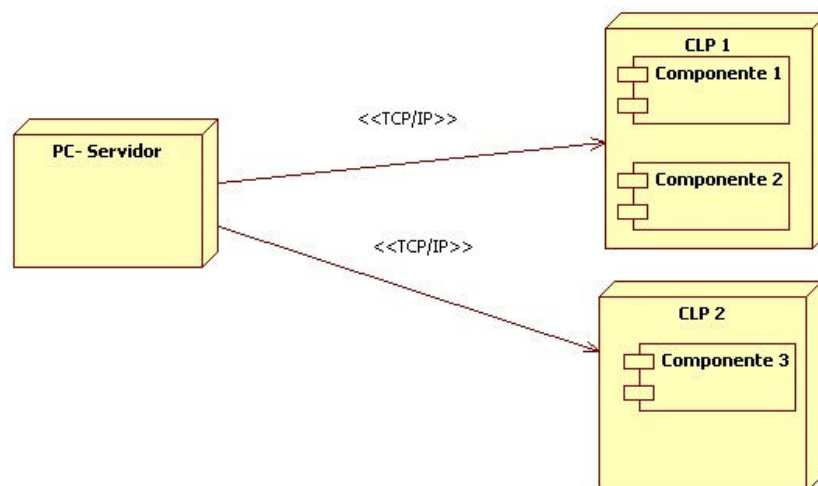


Figura 5.8: Exemplo de Diagrama de Implantação.

Dessa forma, o diagrama de implantação estará representando a arquitetura da solução do sistema de automação, podendo ser a representação de um agente de software ou parte dele. Assim, podemos afirmar que a Figura 5.8 está representando uma solução, em termos de

hardware e funcionalidades, ao qual o agente, que está relacionado à automação, deve possuir para contemplar a solução exigida pelo sistema a ser implementada.

5.2 Estudo de Caso

O estudo de caso consiste na modelagem de um sistema industrial para o controle de qualidade em um processo de cozimento da castanha de caju, processo anterior ao processo de extração da amêndoa da castanha. Tal sistema deve garantir a qualidade do processo e aprender, através das análises inseridas no sistema, quais são os melhores valores para parâmetros como: produtividade e temperatura do processo para cada tipo de castanha.

Apresentamos maiores detalhes sobre o estudo de caso no Anexo A deste trabalho.

Na Figura 5.9 é apresentado o diagrama de caso de uso da especificação do sistema, diagrama este baseado nos requisitos elicitados. Os requisitos e o detalhamento dos casos de uso da especificação do sistema encontram-se no Anexo A.

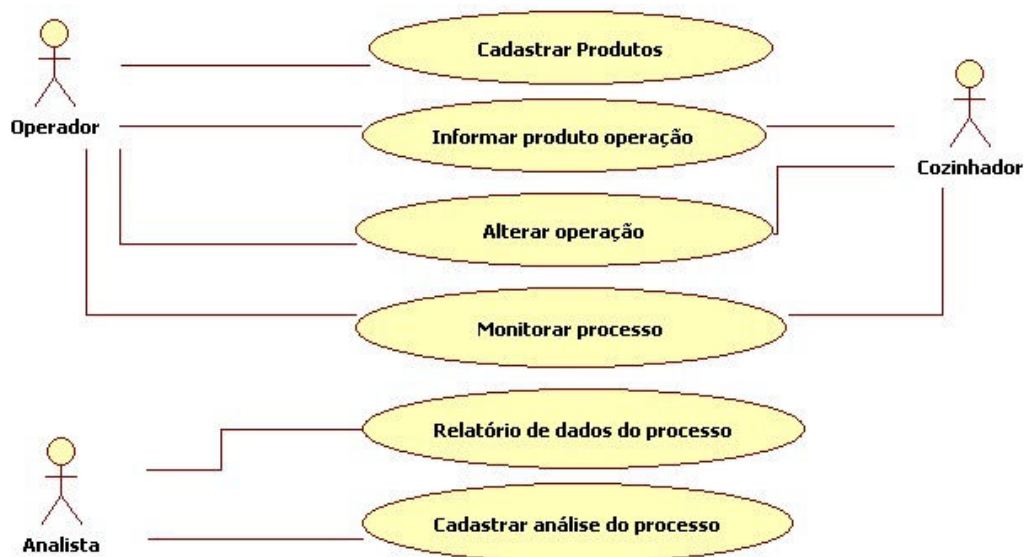


Figura 5.9: Diagrama de casos de uso da especificação do sistema.

Para a modelagem dos artefatos da UML, usamos a ferramenta StarUML (StarUml, 2009) e para a modelagem dos artefatos da metodologia Prometheus, usamos a ferramenta Prometheus Design Tools (RMIT, 2009).

Considerando os requisitos elicitados, devemos começar a modelagem a partir da primeira fase da metodologia Prometheus.

5.2.1 Fase 1 da metodologia Prometheus: Especificação do Sistema

O objetivo desta fase é a identificação das funcionalidades básicas do sistema: percepções, ações e dados compartilhados, assim como as metas e principais funcionalidades do sistema.

Inicialmente analisamos os casos de uso do sistema para extrairmos as metas do sistema. Na metodologia Prometheus, uma meta possui conceito semelhante ao conceito de objetivo. As metas são importantes por fornecerem uma visão de alto nível para o desenvolvimento do sistema e podem derivar de um ou mais casos de uso da especificação do sistema. O diagrama de metas do estudo de caso é apresentado na Figura 5.10, onde é possível visualizar as metas segundo uma hierarquia de importância. A descrição de cada meta é apresentada no Anexo C.

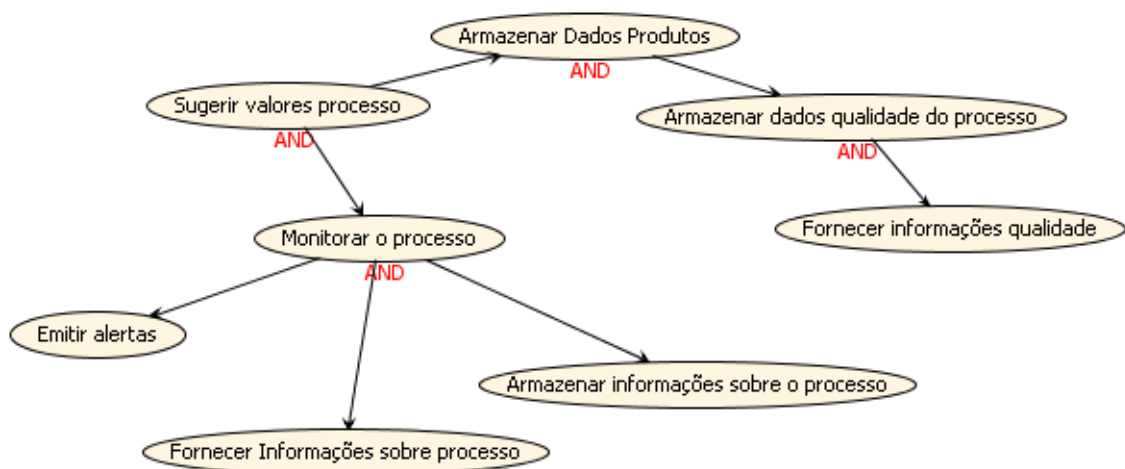


Figura 5.10: Diagrama de metas do estudo de caso.

A construção dos cenários que irão compor o sistema pode ser realizada como atividade paralela à descrição das metas do sistema. Um cenário é uma descrição dos passos necessários para que se cumpra uma meta. Um cenário pode estar associado a uma ou mais metas, representando casos de uso e a descrição do cenário poderá ser semelhante à descrição dos casos de uso que as compõem.

A Figura 5.11 apresenta o diagrama de cenários do nosso estudo de caso, onde podemos visualizar os cenários que estarão presentes na modelagem, representando um ou mais casos de uso do sistema. A descrição dos cenários e a relação com os casos de uso do sistema encontram-se no Anexo C.

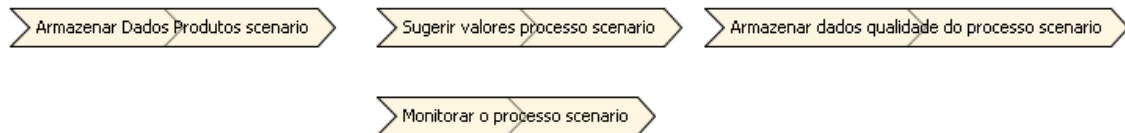


Figura 5.11: Diagrama de cenários do estudo de caso.

A construção do diagrama de análise é o artefato de grande importância nesta primeira fase, pois descreve o ambiente da aplicação através de atores, ações, percepções e cenários. Esse diagrama mostra uma visão detalhada do ambiente, identificando elementos importantes que servirão a outros modelos como as funcionalidades, ações e percepções, bem como os dados produzidos pelo sistema ou os dados externos necessários ao sistema. A Figura 5.12 apresenta o diagrama de análise do sistema, onde é possível ter uma visão geral do ambiente do sistema da aplicação com as percepções, ações, cenários em que eles ocorrem, fluxo de dados e atores. As descrições de cada um dos itens encontram-se no Anexo C.

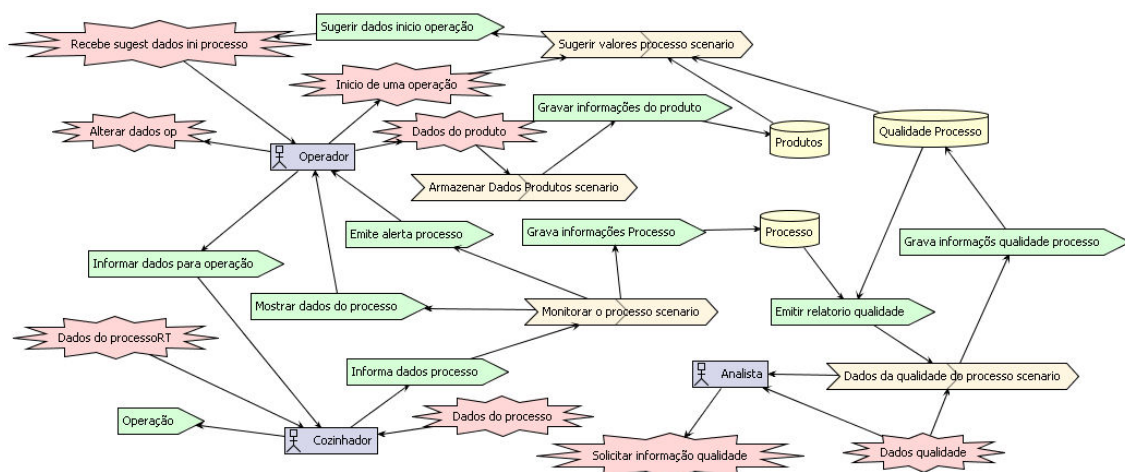


Figura 5.12: Diagrama de análise do sistema do estudo de caso.

O diagrama de funcionalidades do sistema irá descrever o sistema em termos de funcionalidades a serem desempenhadas. Uma funcionalidade pode ser descrita como uma operação que o sistema deverá realizar a fim de cumprir com uma meta. As funcionalidades

podem estar ligadas a percepções e ações podendo estar atrelados a uma ou mais metas do sistema. O conceito de funcionalidade é semelhante ao conceito de papéis. Devido a essa semelhança as funcionalidades do sistema são modeladas como papéis na ferramenta *Prometheus Design Tools*, produzindo-se então o diagrama de papéis. O diagrama de papéis do estudo de caso é apresentado na Figura 5.13, onde podemos observar os papéis que deverão ser executados no sistema podendo estar atrelados a metas, ações e percepções. Cada item do diagrama de papéis está descrito no Anexo C.

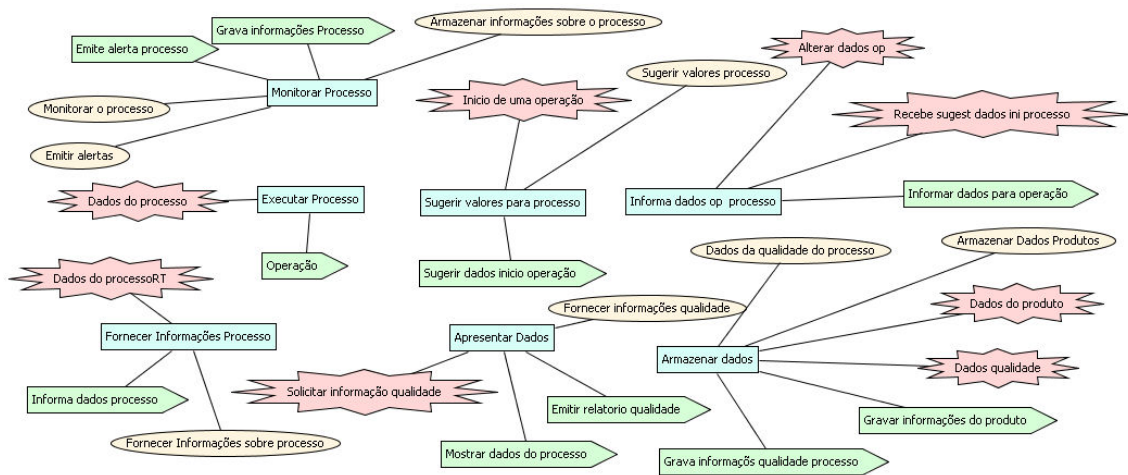


Figura 5.13: Diagrama de papéis do estudo de caso.

Analisando o diagrama de análise do sistema, Figura 5.12, percebemos então que a entidade externa, modelada com o nome de Cozinheiro, está representando um processo industrial. Ou seja, o processo de cozimento da castanha, sendo essencial para o sistema que este processo seja automatizado para que se possa obter informações e controlar o processo. Outro indício desse fato, encontramos no diagrama de funcionalidades do sistema, Figura 5.13, onde encontramos funcionalidades que se referem diretamente ao processo, sendo necessária a automação em tal processo, como a funcionalidade “*Executar Processo*” e “*Fornecer informações do processo*”.

5.2.2 Fase 2 da metodologia Prometheus: Projeto Arquitetural

Esta fase é caracterizada pela identificação dos agentes necessários ao sistema, a modelagem das interações entre agentes e a descrição do comportamento dinâmico do sistema

por meio dos protocolos de interação. Nesta fase também modelamos as soluções de automações necessárias para atender às funcionalidades requeridas pelo sistema, conforme pode ser observado na Figura 5.2.

A fase inicia-se com a modelagem do diagrama de papéis de agente, onde as funcionalidades descritas no diagrama de papéis são agrupadas por semelhança ou compartilhamento de dados para criar um agente. O tipo do agente será determinado pelos papéis que os mesmos executarão no sistema. A Figura 5.14 apresenta o diagrama de papéis de agente do estudo de caso, onde é possível observar os agentes do sistema com os papéis que devam desempenhar. As descrições dos itens deste diagrama encontram-se no Anexo C.

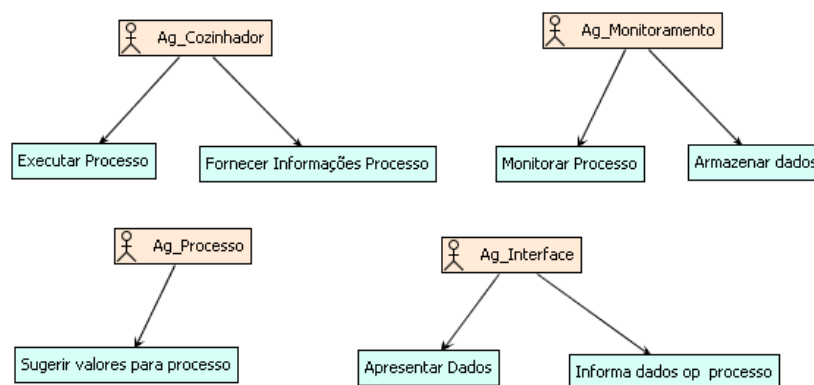


Figura 5.14: Diagrama de papéis de agente do estudo de caso.

É importante esclarecer que o diagrama de papéis é um artefato modelado com base no diagrama de papéis e as entidades agentes são criadas para receberem as tarefas impostas pelos papéis, conseqüentemente as ações, percepções e objetivos associados aos papéis. Tais agentes não possuem relação direta com os atores modelados no diagrama de análise. Se bem que os agentes podem ter nomes atribuídos a eles semelhante aos nomes dos atores para facilitar o entendimento do sistema.

Continuando com o processo de modelagem na metodologia Prometheus, temos a modelagem do acoplamento dos dados em relação às tarefas que se relacionam de alguma forma com os dados modelados no sistema, seja consultando ou manipulando tais dados. Para isso, usamos o diagrama de acoplamento de dados, que mostra quais papéis usam um mesmo

conjunto de dados e, dessa forma, busca identificar semelhanças entre os papéis, que podem levar a junção de papéis ou a necessidade de criação de novos papéis.

A Figura 5.15 apresenta o diagrama de acoplamento de dados da modelagem do estudo de caso, onde podemos observar os papéis a serem desempenhados no sistema relacionando-se a partir da necessidade de informação contida nas bases de dados. Detalhes das bases de dados são apresentados no Anexo C.

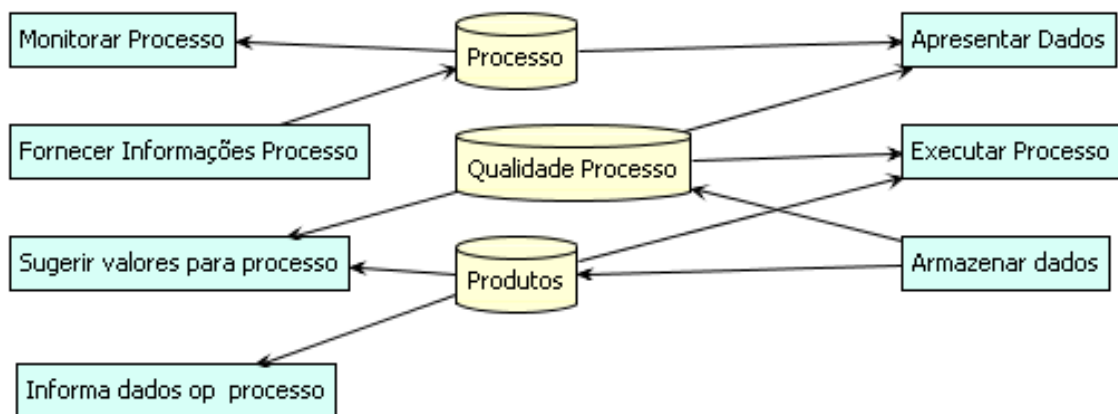


Figura 5.15: Diagrama de acoplamento de dados do estudo de caso.

No diagrama de relacionamento de agente, é apresentada a comunicação entre os agentes, ou seja, o nível de conhecimento que os agentes devem possuir com relação à existência de outros agentes com os quais ele deva interagir. A Figura 5.16 mostra o diagrama de relacionamento do agente do estudo de caso, mostrando o nível de conhecimento do agente necessário para que eles possam cumprir com suas tarefas, bem como demonstra a necessidade de comunicação entre eles.

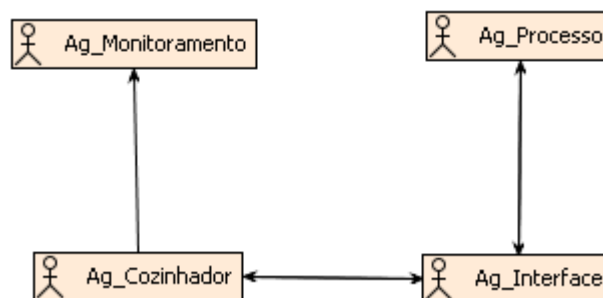


Figura 5.16: Diagrama de relacionamento dos agentes do estudo de caso.

O diagrama de relacionamento de agentes é modelado de forma automática pela ferramenta *Prometheus Design Tools* após a definição dos protocolos de comunicação existentes entre os agentes no diagrama de revisão do sistema, que apresenta uma visão completa do sistema através dos agentes em seu ambiente e das interações entre eles através de protocolos. Em um protocolo de interação, definimos com quem o agente se relaciona e qual mensagem o agente transmitirá. Essa definição deve ser realizada utilizando a linguagem de comunicação especificada para uso na metodologia, a AUML (*Agent unified modeling language*), que é uma adaptação da UML para contemplar modelagem de sistemas baseado em agentes (Padghan, Thangarajah e Winikoff, 2007).

O diagrama de revisão do sistema do estudo de caso é apresentado na Figura 5.17, mostrando a visão geral do ambiente do sistema com os agentes, contendo ações, percepções, dados e os relacionamentos com outros agentes através dos protocolos de comunicação. Os itens deste diagrama são apresentados no Apêndice C deste trabalho.

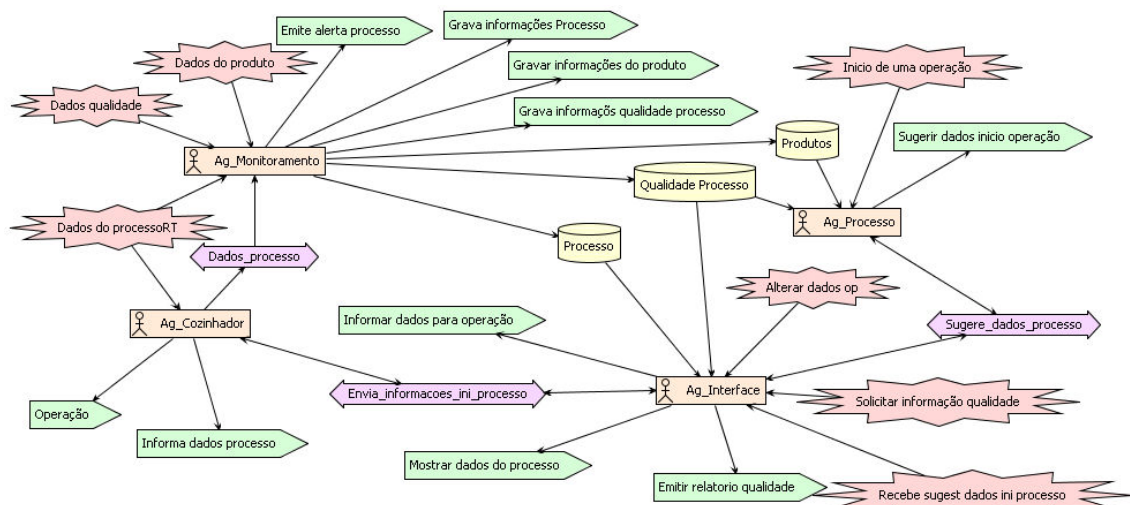


Figura 5.17: Diagrama de revisão do sistema.

O diagrama de revisão representa toda a arquitetura da solução para o sistema modelado.

5.2.3 Fase 3 da metodologia Prometheus: Projeto Detalhado

Nesta fase, a estrutura interna do agente é detalhada para mostrar como o agente realizará suas tarefas dentro do sistema, apresentando descrições minuciosas em termos de capacidades, eventos internos, planos e estrutura dos dados.

A ferramenta *Prometheus Design Tools* cria o diagrama de revisão do sistema de forma automática contendo as ações, percepções e mensagens que o agente necessita, restando apenas a tarefa de interligá-las através do conceito de capacidade, que se refere a uma ação que o agente pode realizar para atingir uma ou mais metas.

A Figura 5.18, apresenta o diagrama de revisão para o agente Cozinheiro, onde temos modeladas as capacidades deste agente. Sendo que cada capacidade pode ser composta por um conjunto de percepções, ações e informações recebidas de outros agentes ou enviadas a outros agentes.

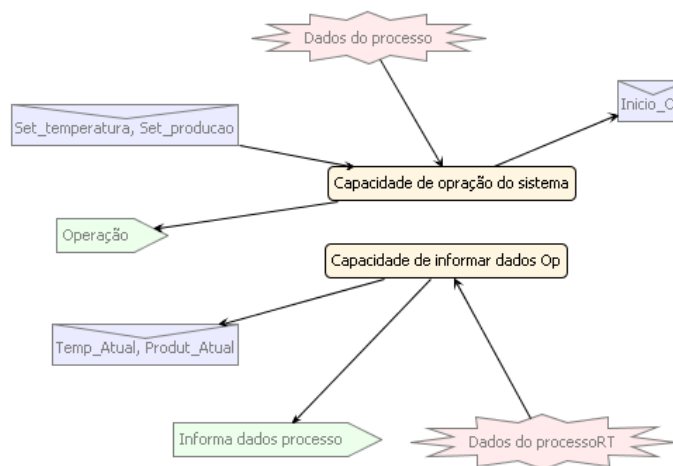


Figura 5.18: Diagrama de revisão do agente cozinheiro.

As capacidades modeladas no diagrama de revisão do agente devem ser decomposta em planos, onde cada capacidade irá gerar um novo diagrama, denominado de diagrama de revisão de capacidades. Um plano pode ser descrito como o detalhamento de uma capacidade. A Figura 5.19 mostra os diagramas de revisão de capacidade e as relações com as ações, percepções e dados necessários para os planos modelados no diagrama de revisão do agente Cozinheiro.

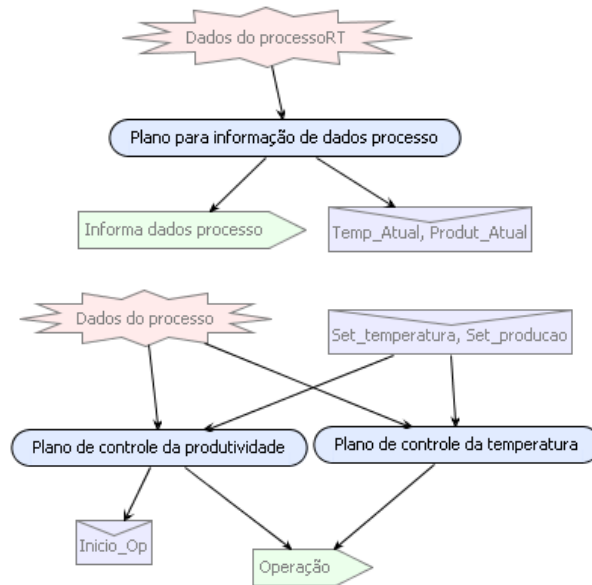


Figura 5.19: Diagrama de revisão de capacidades do agente Cozinheiro.

Os detalhes de cada item modelado no diagrama de revisão e de capacidades do agente Cozinheiro bem como os outros diagramas dos outros agentes que compõem a modelagem estão presentes no Anexo C.

A descrição da arquitetura interna do agente está diretamente ligada ao comportamento desejado para este agente no sistema. Dessa forma, o diagrama de revisão do sistema descreve o comportamento desejado do agente.

5.2.4 Modelagem do sistema de automação

Partindo da análise do diagrama de papéis de agentes e do diagrama de papéis do sistema, podemos identificar a necessidade do desenvolvimento do sistema de automação bem como extrair desses diagramas os requisitos para a aplicação de forma detalhada, pois o diagrama de papéis do sistema detalha cada funcionalidade do sistema ao nível de ações e percepções. Dessa forma, o diagrama de caso de uso do sistema de automação do nosso estudo de caso é apresentado na Figura 5.20, estando as descrições dos casos de uso presentes no Anexo B.

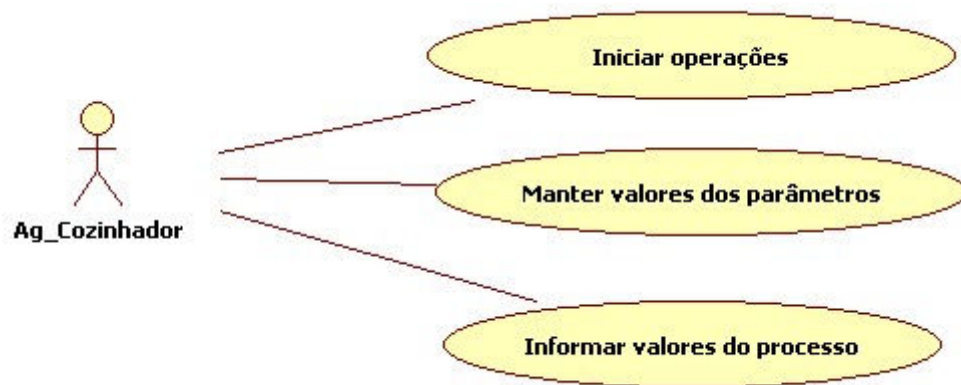


Figura 5.20: Diagrama de Casos de Uso da automação.

Uma vez que os requisitos do sistema de automação estejam modelados, o processo de desenvolvimento do sistema de automação pode ser desenvolvido de forma paralela aos processos da metodologia Prometheus, permitindo que soluções ótimas para o sistema sejam definidas no processo de modelagem.

Após a etapa de levantamento dos requisitos do sistema de automação, o próximo passo é a descrição da arquitetura do sistema para atender de forma satisfatória os requisitos, onde começaremos a modelagem da arquitetura necessária para que se cumpra com os requisitos elicitados.

Na fase de desenvolvimento da micro-arquitetura da solução, modela-se o diagrama de classes da aplicação, onde temos as classes, representando os componentes da automação, as operações, representando as funcionalidades que cada componente deve implementar e as ligações representando as interações necessárias para realizar as operações de forma que todas as funcionalidades necessárias para o sistema de automação sejam modeladas. A Figura 5.21 apresenta o diagrama de classe do sistema de automação, apresentando as funcionalidades necessárias à automação.

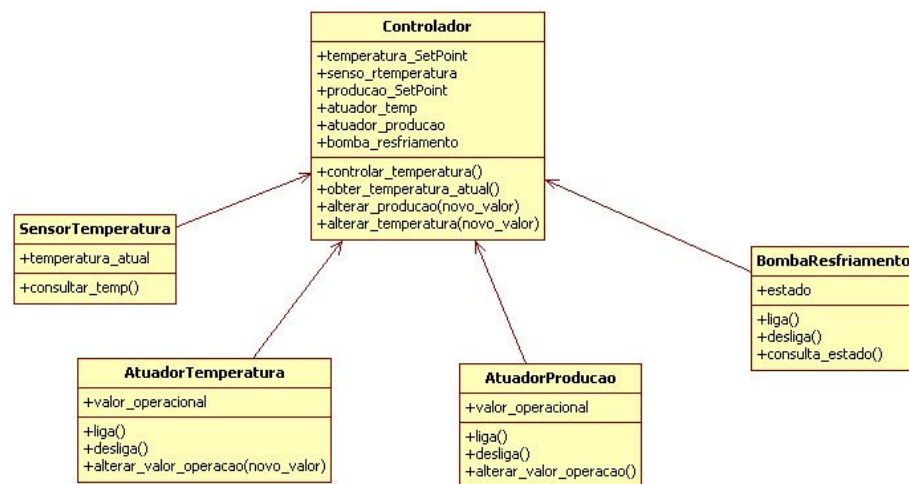


Figura 5.21: Diagrama de classes do sistema de automação.

O diagrama de sequência apresenta a estrutura dinâmica da arquitetura da solução, as interações entre as classes, mostrando como acontecem as interações entre o sistema de controle e o subsistema de automação, bem como as interações entre os objetos da aplicação.

A Figura 5.22 apresenta o diagrama de sequência para a operação de controle do sistema de automação, representando a funcionalidade de operação do Cozinheiro, com o sistema multiagente, representado pela classe “fronteira do sistema”. O diagrama mostra as classes do sistema interagindo para que se cumpram as operações de controle do processo e informação de valores para o sistema.

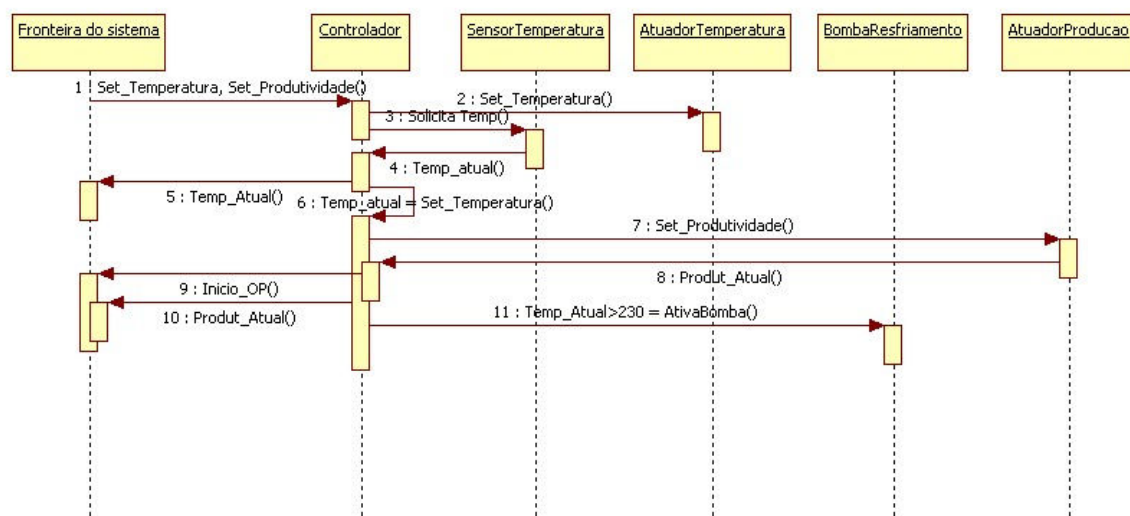


Figura 5.22: Diagrama de sequência do sistema de automação.

Na etapa de macro-arquitetura, as classes são agrupadas para dar origem aos componentes do sistema, conforme podemos ver detalhes dessa operação na Figura 5.23, representando uma solução de hardware e software para o subsistema, onde as funcionalidades implementadas pelas classes são agrupadas formando “macro-componentes” com funcionalidades específicas para o sistema.

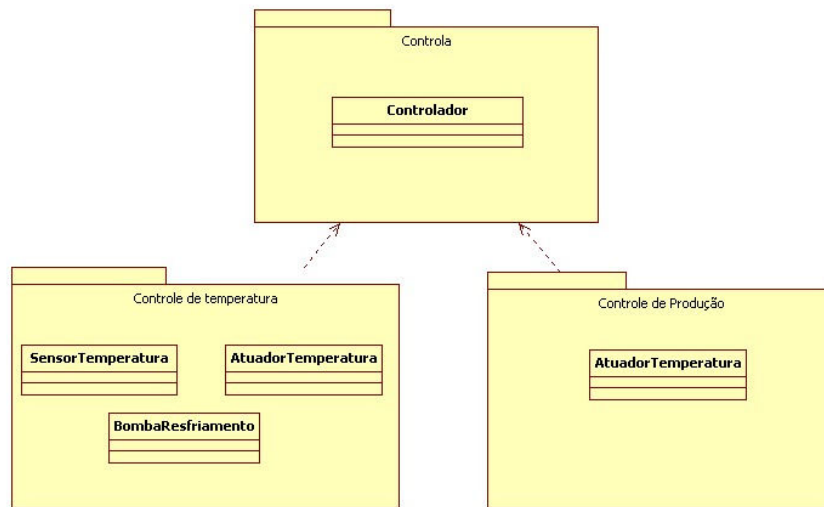


Figura 5.23: Diagrama de pacotes do sistema de automação.

Podemos perceber a partir da Figura 5.23 que as classes podem ser agrupadas, onde funcionalidades básicas provida por uma classe de agrupa a outras para fornecer funcionalidades específicas requeridas pelo sistema.

Na Figura 5.24 temos então a disposição dos componentes, que pode ser um ou mais hardware, sendo formados pelo agrupamento das classes para compor duas funcionalidades específicas: controlar a temperatura do sistema e controlar a produtividade do sistema.

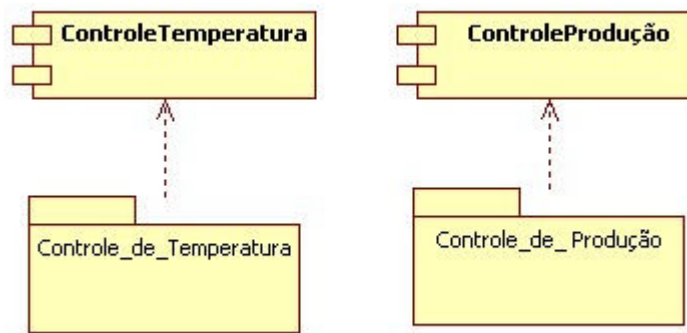


Figura 5.24: Componentes do sistema de automação.

Podemos concluir a partir da figura 5.24 que a solução pode ser composta de componentes, estes poderão ser formado por outros componentes menores, de forma a que tais componentes consigam oferecer os serviços essenciais requeridos pelo sistema.

O último passo da extensão é a modelagem do diagrama de implantação, que mostra como o sistema será implementado em termos de hardwares e funcionalidades, bem como a comunicação entre os equipamentos da solução. Na Figura 5.25 é demonstrada a solução da automação sendo composta por duas CLP's, onde cada CLP implementa os componentes da aplicação, através do protocolo de comunicação *fieldbus* e provê a interface de comunicação com o servidor. As CLP's são ligadas ao computador servidor da aplicação através do protocolo de comunicação TCP/IP. Cada nó da rede possui poder de processamento e comunicação, implementando as funcionalidades que o sistema de automação deve oferecer ao sistema multiagente.

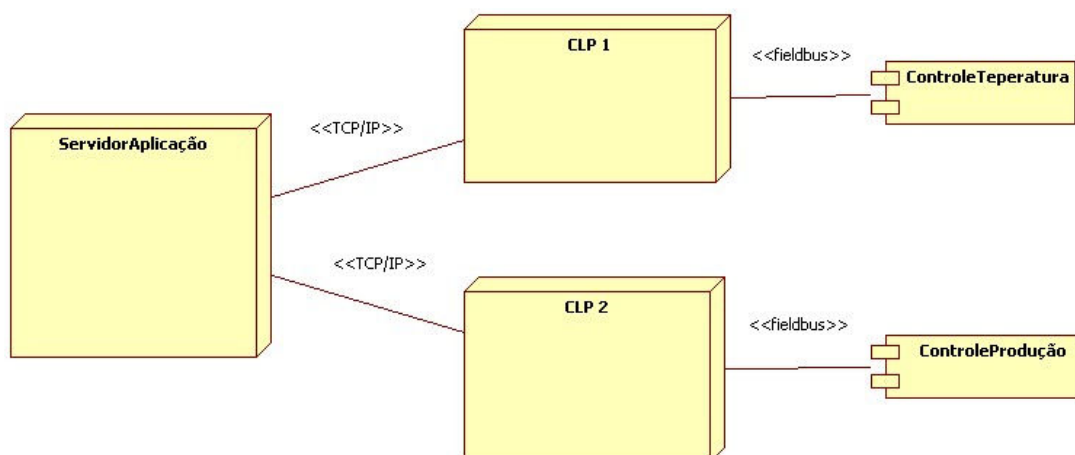


Figura 5.25: Diagrama de implantação do sistema de automação.

5.3 Considerações Finais do Capítulo

Neste capítulo, apresentamos uma proposta para estender a metodologia Prometheus que, conforme concluímos no Capítulo 4, não contempla o desenvolvimento de sistemas de automação. Dessa forma, propomos a extensão da metodologia Prometheus em mais 4 fases, descrevendo os passos necessários.

Apresentamos também um estudo de caso, demonstrando o uso da metodologia Prometheus na modelagem de sistemas industriais e aplicamos a extensão proposta para gerar o sistema de automação industrial, abstraído na metodologia em forma de agente. Dessa forma, a metodologia Prometheus passa a contemplar todas as fases necessárias para do desenvolvimento de um sistema industrial e a modelagem do sistema de automação industrial passa a ser discutido antes de ser implementada, dando a possibilidade aos projetistas de discutir as soluções para o sistema de automação ainda na fase de modelagem. A modelagem da solução da automação pode ser realizada em paralelo com a modelagem da solução como um todo.

CAPÍTULO 6

CONCLUSÕES

Neste trabalho abordamos a temática do desenvolvimento de sistemas industriais, onde nos deparamos com a complexidade de desenvolvimento de tais sistemas devido aos novos desafios e tendências que estes sistemas devem atender neste início de século.

Estudamos o paradigma de agentes, mostrando como suas características são importantes para a abordagem de soluções para os sistemas industriais modernos, apresentando alguns exemplos de aplicações dos agentes em soluções de sistemas industriais.

Com o objetivo de modelarmos aplicações industriais modernas, estudamos as metodologias de desenvolvimento de sistemas industriais e concluímos que as mesmas não se aplicam às novas exigências para os sistemas industriais. Então, buscamos o desenvolvimento de tais sistemas pelo paradigma de agentes, onde o paradigma de agentes fornecia conceitos que permitiam contemplar os requisitos exigidos aos sistemas industriais modernos.

Assim, escolhemos a metodologia Prometheus, para a modelagem de uma aplicação e nos deparamos com o fato da metodologia não fornecer detalhes suficientes para a modelagem da automação industrial. Por isso, propomos uma extensão à metodologia Prometheus para que a mesma possa contemplar essa etapa essencial aos sistemas industriais.

E, dessa forma, mostramos a importância do paradigma de agentes e como um sistema industrial pode ser modelado através de uma metodologia orientada a agentes que contemple todas as fases do desenvolvimento da solução.

6.1 Contribuições

Este trabalho contribui diretamente para a abordagem da modelagem dos sistemas industriais pelo paradigma de agentes. Porém, podemos citar outras contribuições, como:

- A importância dos agentes para o domínio industrial;
- Detalhamento da modelagem de um sistema orientado a agente pela metodologia Prometheus;
- Apresentação e comparações entre as metodologias orientadas a agentes e as metodologias para desenvolvimento de sistemas industriais.

- Contribuição com uma técnica de modelagem para os sistemas industriais;
- Extensão da metodologia Prometheus;
- Demonstração de como abordar um problema real do domínio industrial e seu processo de modelagem.

6.2 Limitações e Trabalhos futuros

Tendo em vista a natureza dos sistemas de automação industrial de serem reativos a eventos, há a necessidade de se incluir um modelo para descrever o comportamento da automação na extensão da metodologia contemplando essa limitação. Dessa forma, sugerimos como trabalho futuro a especificação de um modelo para descrever o comportamento da solução para que o mesmo seja, de alguma forma, acoplada a extensão.

Em nossa solução proposta contemplamos apenas a modelagem da solução para desenvolvimento de uma aplicação de automação industrial. Sabendo que, a própria solução do sistema orientado a agente também deverá possuir um modelo que descreva sua arquitetura, pois tal arquitetura não é detalhada na metodologia Prometheus, deixaremos esta essa solução para ser contemplada como trabalho futuro.

Uma avaliação sobre a eficácia dos modelos propostos na extensão poderá ser realizada através do desenvolvimento da aplicação modelada no estudo de caso deste trabalho.

O modelo proposto não neste trabalho não foi aplicado em outras metodologias. Seria interessante tentar acoplar a nossa extensão em outras metodologias orientadas a agentes e observar como ela se comporta, ficando este como trabalho futuro.

BIBLIOGRAFIA

AIELLO,G.; ALESSI, M.;BRUCCOLERI C.; ONOFRIO,D.; VELLA,G. **An Agile methodology for manufacturing control system development.** In 5th IEEE International Conference on Industrial Informatics. Vienna, Australia. 2007

ARRUDA, L. V. R.; SWIECH, M.C.S.; NEVES-JR, F.; DELGADO, M.R. **Um método Evolucionário para Sintonia de Controladores PI/PID em Processos Multivariáveis.** Revista Controle & Automação, Vol. 19. No.1, 2008.

BARBOSA, V.L.R. **Desenvolvimento de um Micro Sistema Supervisório: Aplicado ao Transporte de Gás Natural.** E-TECH: Tecnologia para Competitividade Industrial. v.2, n.1, p. 49-65. Florianópolis, SC. 2009.

BELLIFEMINE, F.; POGGI, A.; RIMASSA, G. **Developing Multi-agent System With JADE.** Intelligents Agents VII, LNAI 1986, p. 89-103 Berlin Heidelberg. 2001.

BERNON, C.; GLEIZES, M.P.; PICARD, G.; GLIZE, P. **The ADELFE Methodology for an Intranet System Design.** Proceeding of Agent-Oriented Information System, AOIS-2002 p. 1-15 . 2002.

BONELI, R. **Políticas de Competitividade Industrial no Brasil – 1995/2000.** Instituto de Pesquisa Econômica Aplicada (IPEIA). Rio de Janeiro – RJ. Julho, 2001.

BROGRAD, T. **Present and Future Robot Control Development. – An Industrial Perspective.** Annual Reviews in Control, 31(1) p. 69-79. 2007.

BRUCCOLERI, M., LO NIGRO, G., PERRONE, G., RENNA, P., NOTO LA DIEGAS. **Production Planning in Reconfigurable Enterprises and Reconfigurable Production Systems.** Annals of the CIRP, 54/1: 433-436, 2005.

BRUCIAPAGLIA, A.H.; FARINES, J.M.; CURY, J.E.R. **Automação no Processo Produtivo: Desafios e Perspectivas.** Revista Nexus, Vol.1 p. 28-31. 2001.

CAIRE, G.; LEAL, F.; CHAINHO P.; EVANS, R.; GARIJO, F.; GOMEZ, J.; PAVON G.; KEARNEY J. **Agent Oriented Analysis using MESSAGE/UML**. AOSE. 2001.

CALLAI, T. C.; COELHO, L.S.; COELHO A.A.R. **Controle Nebuloso Adaptativo por Modelo de Referência: Projeto e Aplicação em Sistemas Não-Lineares**. Revista Controle & Automação Vol.18, p. 479-489. 2007.

CARPANZANO, E.; JEOVANE, F. **Advanced Automation Solutions for Future Adaptive Factories**. Annals of the CIRP. Vol. 56, p. 435-538. Jan, 2007.

CASSANDRAS, C. G.; LAFORTUNE, S. **Introduction to discrete event systems**. Kluwer Academic Publishers, Massachusetts, USA, 1999.

CHAN, F.T.S.; SHANKAR, R. **Ant colony optimization approach to a goal programming model for a machine tool selection and operation allocation problem in an FMS**. Robotics and Computer-Integrated Manufacturing, Vol. 22, pp. 353-362, 2006.

CORRÊA, H. L.; CORRÊA, C.A. **Administração da Produção e de Operações: Manufatura e Serviços – Uma abordagem Estratégica**. Editora Atlas, São Paulo-SP. 2005.

COSENTINO, M.; POTTS, C. **A process for specifying and implementing multi-agent systems using UML**. 2001.

CURY, J.E.R. **Teoria de Controle Supervisório de Sistemas a Eventos Discretos**. V Simpósio Brasileiro de Automação Inteligente, Gramado-RS, 2001.

DAN'Á, S.; SAGAHYROON, A.; ELRAYES, A.; AL-ALLI, A. R.; and AL-AYDI, R. **Development of a monitoring and control platform for PLC-based applications**. Computer Standards & interfaces, Vol. 30; pp. 157-166. 2008.

DAM, K. H. **Evaluating and comparing Agent-Oriented Software Engineering Methodologies.** Master thesis of applied Science in Information Technology, RMIT University, Australia. 2001.

DANEELS, A.; SALTER W. **What is SCADA?** ICALEPCS'99 International Conference on Accelerator and Large Experimental Physics Control Systems, Trieste, Italy. Out 1999.

DECOTIGNIE, J.-D. **Ethernet-based Real-Time and Industrial Communications.** Proc. IEEE, vol. 93, no. 6, pp. 1102–1118, Jun. 2005.

DELOACH, S.A. **Analysis and Design Using MaSE and AgentToll.** Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2001) Miami University, Oxford, Ohio Mar, 2001.

DELOACH, S.A.; WOOD, M. F.; SPARKMAN, C. H. **Multiagent System Engineering.** The International Journal Of Software Engineering and Knowledge Engineering. 11(3), p. 231-258. 2001.

DMEC, Site do departamento de Engenharia mecânica da Universidade Federal de Minas Gerais. www.demec.ufmg.br/grupos/usinagem/CADCAM.htm - Acessado em 20 de dezembro de 2009.

DJASSEMI, M.; SENA J.A. **The Paperless Factory: A Review of Issues and Technologies.** Internacional Journal of Computer Science and Network Security (IJCSNS) Vol. 6 n. 12 p. 185-191. 2006.

DURAN, J. M. **A Qualidade Desde o Projeto: Novos Passos para o Planejamento da Qualidade em Produtos e Serviços.** Editora Pioneira, São Paulo-SP. 2004.

ELIPSE, Elipse software: <http://www.iconics.com/products/genesis32.asp> - Acessado em Dezembro de 2009.

ERICKSON, K. T. **Programmable logic controllers**. IEEE Potentials, vol. 15, pp. 14–17, Feb./Mar. 1996.

FERRARINI, L.; VEBER, C.; LUDER, A.; PESCHKE, A.; KALOGERAS, A.; GIALELIS, J.; RODE, J.; WUNSCH, D.; CHAPURLAT, V. **Control Architecture for Reconfigurable Manufacturing Systems: the PABADIS’PROMISE approach**. 11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA’06) Prague, Czechia, Proceedings, pp. 545-552, 2006.

FERREIRA, S. L. C.; GIRARDI, R. **Arquitetura de Software Baseada em Agentes: do nível global ao detalhado**. Revista Eletrônica de Iniciação Científica da SBC, 2002.

FIPA, Foundation for Intelligent Physical Agents. **Fipa2000 agent specification**. Disponível em: <http://www.fipa.org>.

FIPA-OS - Plataforma FIPA para desenvolvimento de sistema multiagente. disponível em: <http://fipa-os.sourceforge.net> – Acessado em dezembro de 2009.

FOWLER, M. **UML Essencial: um breve guia para a linguagem padrão de modelagem de objetos**. Ed. Bookman, 3 ed. Porto Alegre-RS. 2005.

GARCIA, A.; LUCENA, C.; COWAN, D. **Agents in Object- Oriented Software Engineering**. In: Software: Practice & Experience, Elsevier, 34 (5), April 2004, pp. 489-521.

GARCIA-OJEDA, J.C.; DELOACH, S.A.; ROBY-OYENAMR, W.H.; VALENZUELA, J. **O-MaSE: A Customizable Approach to Developing Multiagent Development Processes**. Lecture Notes in Computer Science, Issue no. 4957, pp. 1-15, 2008.

GIORGINI, P.; HENDERSON-SELLERS, B. **Agent-Oriented Methodologies: An Introduction**. In: Agent-Oriented Methodologies. Idea Group Publishing, S. 1-19. 2005.

GIRARDI, R. **Engenharia de Software Baseada em Agentes**. IV Congresso Brasileiro de Ciência da Computação – CBCOMP. Itajaí, SC. ed. 39. 2004.

GIRET, A.; BOTTI, V. **Engineering Holonic Manufacturing Systems**. Computers in Industry, vol. 60, p. 428-440 mar, 2009.

GÓMEZ-SANZ, J.; PAVÓN, J. **Methodology for Developing Multi-Agent Systems**. Journal of Universal Computer Science, Vol 10. n.4 p. 359-374. 2004.

GREENWOOD, D.; LYELL, M.; MALLYA, A. ; SUGURI, H. **The IEEE FIPA Approach to Integrating Software Agents and web Services**. In AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems, pages 1-7, New York, USA, 2007.

GUNASEKARAN, A. **Implementation of Computer-Integrated Manufacturing: a Survey of Integration and Adaptability Issues**. Int. J. Computer Integrated Manufacturing, Vol 10, nº 1-4, 266-280, 1997.

GUTIEREZ, R. M. V.; PAN, S.S.K. **Complexo Eletrônico: Automação do Controle Industrial**. BNDES Setorial, Rio de Janeiro, nº 28, p. 189-232, set., 2008

HORLING, B.; LESSER, V. **A survey os multi-agent organizational paradigms. The Knowledge Engineering Review**. Cambridge University Press, Vol. 19:4, p. 281-316, 2005.

ICONICS, Site da Iconics: <http://www.iconics.com/products/genesis32.asp> - Acessado em Dezembro de 2009.

IGLESIAS, C.A.; Garijo, M. **The Agent-Oriented Methodology MAS-CommomKADS**. In Henderson-sellers, B & Giorgini, P (Eds), Agent-Oriented Methodology; pp. 46-78. London, 2005.

JACK - Plataforma de desenvolvimento de sistema multiagentes JACK releases notes. Disponível em: http://www.aosgrp.com/documentation/jack/jack_release_notes.html - Acessado em dezembro de 2009.

JENNINGS, N. R. **Agent-Oriented Software Engineering**. In: Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World System Engineering (MAAMAW-99), Anais Springer-Verlag, v.1647, p.1–7. Heidelberg, Germany, 1999.

JENNINGS, N. R.; BUSSMANN, S. **Agent-Based Control Systems**. IEEE Control Systems Magazine, V. 23(3), p. 61-74, 2003.

JENNINGS, N. R.; WOOLDRIDGE, M. J. **Applying Agent Technology**. Int. Journal of Applied Artificial Intelligence, 9 (4) 351-369. 1996.

KUMAR, K. D.; KARUNAMOORTHY, L.; ROTH, H.; MIRANLINEE, T. T. **Computers in manufacturing: towards successful implementation of integrated automation systems**. Technovation, Vol. 25, No. 5; pp. 477-488. 2004.

LEITÃO, P.; RESTIVO, F.J. **Implementation of a Holonic Control System in a Flexible Manufacturing System**. IEEE Transactions on Systems, MAN, And Cybernetics: Applications and Review, Vol.38, No. 5, 2008.

LIMA, R.M. **Sistemas Distribuídos de Produção em Ambiente de Produção Simultâneas**. Tese de doutorado submetida a Universidade do Minho, Braga, Portugal, 2003.

LIND, J. **Insues in Agent-Oriented Software Engineering**. AOOSE 2000 on Agent-Oriented Software Engineering, p. 45-58, 2001.

MACKAY, S.; WRIGHT, E.; REYNDERS, J.; PARK, J. **Practical Industrial Data Networks: Design, Installation and troubleshooting**. Newnes, 1 ed. IDC Technologies, ISBN 07506 5807X, 2004.

MADEJSKI, J. **Survey of the agent-based approach to intelligent manufacturing**. Journal of Achievements in Materials and Manufacturing Engineering, Vol. 21, No. 1; pp. 67-70, 2007.

MANESIS, S.; AKANTZIOTIS. **Automated synthesis of ladder automation circuits based on state-diagrams.** Advances Engineering Software issues 4, vol 36 p.225–233 , 2005.

MARICK, V.; MCFARLANE D. **Industrial Adoption of the Agent-based Technologies.** To be published in IEEE Intelligent Systems, 2005.

MARICK, V.; VRBA, P.; HALL, K. H.; MATURANA, F. P. **Rockwell automation agents for manufacturing.** presented at AAMAS, Utrecht, NL, 2007.

MECATRONICA Artigo publicado no site da revista Mecatrônica Atual em 22 de Dezembro de 2008. Disponível em <http://www.mecatronicaatual.com.br/secoes/leitura/324>. Acessado em Dezembro de 2009.

MONOSTORI, J. L.; VÁNCZA, S. R.; KUMARA, T. **Agent-based systems for manufacturing.** Annals of the CIRP, Vol. 55, No. 2; pp. 672-720, 2006.

MOON, Y.B. **Enterprise Resource Planning (ERP).** Int. J Management and Enterprise Development, vol. 4, No. 3, 2007.

MOREL, G.; VALCKENAERS, P.; FAURE, M. J.; PEREIRA C.; DIEDRICH, C. **Survey paper on manufacturing plant control challenges and issues.** 16th IFAC World Congress in Prague, 2007.

MOURA, R.; GUEDES, L. **Modelagem de aplicações de automação industrial usando diagramas uml e reuso de componentes.** VIII Simpósio Brasileiro de Automação Inteligente SBAI, 2007.

MOYNE, J.R.; TILBURY, D.M. **The Emergence of Industrial Control Networks for Manufacturing Control, Diagnostics, and Safety Data.** Proceedings of the IEEE. Vol.25, No. 1, Jan, 2007.

ODELL, J.; PARUNAK, V.D.; BAUER, H.B. **Extending UML for agents**. In G. Wagner, Y. Lesperance & E. Yu (Eds.), Proceedings of Agent-Oriented information Systems Workshop (pp. 3-17). 17th National Conference on Artificial Intelligence, Austin, TX, 2000.

OMG, Object Management Group. **Agent Platform Special**. Interest Group Agent Technology – Green Paper, version 1.0. P.9. September, 2000.

PADGHAN, L.; WINIKO, M. **Prometheus: A pragmatic methodology for engineering intelligent agents**. In J. Debehm, B. Henderson-Sellers, N. Jennings, J. Odell (Eds.), Proceedings of the OOPSLA 2002 Workshop on Agent-Oriented Methodologies, Seattle, Washington; pp. 97-108, 2002.

PADGHAN, L.; THANGARAJAH, J.; WINIKO, M. AUML Protocols and Code Generation in the Prometheus Design Tool. AAMAS'07, Honolulu, Hawaii, USA. May, 2007.

PEREZ, D.H. **Decentralized Coordination of Autonomous AGVs in Flexible Manufacturing Systems**. IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3674–3679, 2008.

PFLEEGER, S.L. **Engenharia de Software: Teoria e Prática**. Pearson Prentice Hall, São Paulo – SP, 2004.

PROUTY, R. **The Application of a Modern Control System in a Cement Manufacturing Facility**. IEEE Cement Industry Technical Conference. Conference Record. pp.268-274. April, 2006.

PRESSMAN, R. S. **Engenharia de Software**. Mc Graw Hill, 5 ed. Rio de Janeiro – RJ. 2002.

QUEIROZ, M. H.; CURY, J.E.R. **Controle Supervisório Modular de Sistemas de Manufatura**. Revista Controle & Automação. Vol 13 n°2, p. 123-133 , agosto, 2002.

QINGUI, D. **On relation of manufacturing system, manufacturing mode and manufacturing technology.** IEEE- International Technology and Innovation Conference, Vol. 1; pp. 889-893, 2006.

RAMADGE, P. J.; WONHAN, W.M. **The Control Of Discrete Event Systems.** Proceedings IEEE, Special Issues on Discrete Event Dynamic Systems, Vol. 77, nº 1, p. 81-98. 1989.

RATIONAL, **Metodologia RUP.** Disponível em <http://www.rational.com> Acessada em Novembro de 2009.

REZENDE, D. A. **Engenharia de Software e Sistema de Informações.** Editora Brasport, Ed. 3. Rio de Janeiro – RJ. 2005.

RMIT Agents Group. **Prometheus e PDT (Prometheus Design Tool).** Disponível em: <http://www.cs.rmit.edu.au/agents/pdt> - Acessado em Janeiro de 2010.

RUSSEL, S.; NORVIG P. **Inteligência Artificial: uma abordagem moderna.** Editora Campus, São Paulo-SP, 2004.

SAJJA, P. S. **Multi-Agent System for Knowledge-Based Access to Distributed Databases.** Interdisciplinary Journal of Information, Knowledge, and Management Volume 3, p. 1-9, 2008.

SAMAD, T.; MCLAUGHLIN, P.; LU, J. **System Architecture for Process Automation: Review and Trends.** Journal of Process Control. n.17, p.191-201, 2006.

SENSORES, **Artigo on-line sobre sensores.** Disponível em: <http://sensores.tk> – Acessado em Dezembro de 2009.

SEO, Y. **Automated Design of Manufacturing Systems Through Virtual Manufacturing Modeling Technique.** KORUS'99 Proceedings of the 3rd Russian-Korean International Symposium on Science and Technology, pp. 293-296. June, 1999.

SHANAKAR, K. G. **Control of Boiler Operation using PLC-SCADA**. Proceedings of the Internacional MultiConference of Engineers and Computer Scientist, IMECS 2008. Hong-Kong, March 2008.

SHEN, W.; HAO, Q.; YOON, H.J.; NORRIE, D.H. **Applications of Agent-Based Systems in Intelligent Manufacturing: An Update Review**. Advanced Engineering Informatics, 20 p. 415-431, 2006.

SILVA, C.T.L.; TEDESCO, P.C.A.R.; CASTRO, J.F.B.; SILVA R.C. **Comparing Agent-Oriented Methodologies Using the NFR Approach**. In Proceedings of the Third International Workshop on Software Engineering for Large-Scale Multi-Agent Systems, SELMAS'04. Vol. 1, pp: 1-9. 2004.

SILVA, H.F.; HENRIQUE, M. H.; LOPES, L.C.O.; GOMES, L.R. **Controle Preditivo Robusto Baseado em Inequações Matriciais Lineares Aplicado a Máquinas Síncronas**. Ver. Ciênc. Exatas, Taubaté, V. 13, No.1. p. 121-129, 2007.

SLACK, N.; CHABERS, S., JOHNSTON, R. **Administração da produção**. 2 ed. Editora Atlas. São Paulo SP, 2008.

SOARES, M. S. **Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software**. 2009.

SOUZA, R. B.; MEDEIROS, A. D.; NASCIMENTO J. M.; MAITELLI, A. L.; GOMES, H. P. **SISAL – Um sistema supervisório para elevação artificial de petróleo**. Rio Oil & Gas Expo and Conference, Rio de Janeiro, 2006.

SOUZA, F. C. **Desenvolvimento de metodologia de aplicação de redes de Petri para automação de sistemas industriais com controladores lógicos programáveis (CLP)**. Dissertação de Mestrado, Universidade de São Paulo – USP 146p. São Paulo, SP. 2006.

SOMMERVILLE, I. **Software Engineering**, Ed. 6th, Addison- Wesley, 2000.

STARUML. Disponível em <http://staruml.sourceforge.net/en/>. Acesso em: novembro de 2009.

TRAMBOULIDIS, K. **IEC61499 in Factory Automaton**. In International Conference on Industrial Electronics, Technology and Automaton, 2005.

TISSERANT, E.; BESSARD, L.; SOUSA, M. **An Open Source IEC 61131-3 Integrated Development Environment**. INDIN 2007, 2007.

TOMMILA, T.; VENTA, O.; KOSKINEN, K. **Next generation industrial automation—needs and opportunities**. Automation Technology Review. 2001.

VMODEL, Metodologia em V. disponível em: <http://www.v-modell.iabg.de> – Acessada em 12/2009

WANG, L.; TAN, K.C. **Modern Industrial Automation Software Design: Principles and real-world examples**. IEEE Publication, 2006

WEYNS, D.; SCUMACHER, M.; RICCI, A.; HOLVOET, T. **Environments in Multiagent Systems**. The KnowledgeEngineering Review, vol. 20, p. 127-141, 2005.

WOLF, W. **Computer as Components – Principles of Embedded Computing System Design**. Academic Press. 1 ed. 2001.

WONDERWARE, Site da WonderWare: <http://www.wonderware.com> - acessado em dezembro de 2009.

WOODRIDGE, M.; JENNINGS, N. R. **Intelligent Agents: Theory and Practice**. The Knowledge Engineering Review (1995), Citeseer, 2009.

XU, X.W.; NEWMAN, S.T. **CNC machine tools more open, interoperable and ntelligent—a review of the technologies**,.Comput. Ind. Vol. 57 (2), P. 141–152, 2006.

ZAMBONELLI, F.; JENNINGS, N. R.; WOOLDRIDGE, M. **Developing Multiagent Systems: The Gaia Methodology.** ACM Transactions on Software Engineering and Methodology, Vol. 12, No. 3; pp. 417-470. 2003.

ZAMBONELLI, F.; OMICINI, A. **Challenges and Research Directions in Agent-Oriented Software Engineering.** Autonomous Agent and Multi-Agents System. Vol 9. P. 253-283. 2004.