Vanderlei Luis Conrad Weber

Derivação de Sistemas Multiagentes industriais através do *framework* O-MaSE

Dissertação submetida ao Programa de Pós-Graduação em Engenharia de Automação e Sistemas para obtenção do Grau de Mestre.
Orientador: Prof. Ricardo José Rabelo, Dr.
Coorientador: Prof. Jomi Fred Hübner, Dr.

Florianópolis

2013

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Weber, Vanderlei Luis Conrad
Derivação de Sistemas Multiagentes industriais através do framework O-MaSE / Vanderlei Luis Conrad Weber ; orientador, Ricardo José Rabelo ; coorientador, Jomi Fred Hübner. - Florianópolis, SC, 2013.
93 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Inclui referências

1. Engenharia de Automação e Sistemas. 2. Sistemas Multiagentes. 3. Ontologia. 4. Metedologia de derivação para Sistemas Multiagentes. 5. Sistemas Industriais. I. Rabelo, Ricardo José. II. Hübner, Jomi Fred . III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. IV. Título.

Vanderlei Luis Conrad Weber

Derivação de Sistemas Multiagentes industriais através do *framework* O-MaSE

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Florianópolis, 11 de Dezembro de 2013.

_____
Prof. Jomi Fred Hübner, Dr.
Coordenador do Curso
Universidade Federal de Santa Catarina

_____
Prof.ª Ricardo Jose Rabelo, Dr.
Orientador
Universidade Federal de Santa Catarina

Banca Examinadora:

_____
Prof.ª Marcelo Ricardo Stemmer, Phd.
Presidente
Universidade Federal de Santa Catarina

_____

Prof.ª Carlos Barros Montez, Dr.
Universidade Federal de Santa Catarina

_____

Prof. Luis Otavio Campos Alvares, Phd.
Universidade Federal de Santa Catarina

Dedico essa dissertação aos meus pais que tanto admiro e que possibilitaram que eu cumprisse essa tarefa.
À minha esposa e a toda minha família.

AGRADECIMENTOS

Agradeço aos meus pais por serem essas pessoas maravilhosas que sempre me apoiaram e aconselharam em cada etapa da minha vida.

Agradeço a Shirlei pela paciência nas horas mais difíceis e pelo amor e carinho no decorrer desses anos.

Ao meu orientador pelo apoio e contribuição nesse trabalho.

Aos meus amigos/irmãos que nunca excitaram em ajudar e apoiar.

RESUMO

O problema de derivação de sistemas multiagentes industriais envolve complexos aspectos. Uma derivação consiste basicamente em gerar uma instância de sistema multiagente a partir de um modelo ou arquitetura genérica, com posteriores adaptações para um certo caso particular. No caso específico, este caso particular corresponde a um dado chão-de-fábrica, a ser gerenciado pelo sistema multiagente derivado.

A complexidade deste problema reside principalmente na natureza do cenário de chão-de-fábrica, que agrega inúmeros aspectos que devem ser consideradas no momento da modelagem do sistema. A presença de uma estrutura heterogênea de sistemas legados e equipamentos deve ser muito bem analisada e descrita para que o resultado final seja coerente com as características, layout e funcionamento do chão de fábrica em questão.

A abordagem de solução utilizada neste trabalho para fazer frente a essa complexidade foi a de executar uma derivação com o apoio de uma metodologia especialmente desenvolvida para tal. Esta metodologia é composta por etapas criadas a partir da análise e escolha dos meta-modelos do *framework* O-MaSE, que comporão a metodologia propriamente dita. Além disto, para suportar os necessários requisitos, duas etapas específicas de descrição de cenários industriais foram criadas em relação ao modelo original do O-MaSE. Uma para especificar o chão de fábrica e outra para representação do conhecimento deste, o que é feito através de uma ontologia.

O resultado obtido pela derivação elaborada e pela adaptação feita da O-MASE é representado pela geração de códigos-base de agentes que representam virtualmente o cenário. Com a utilização do *framework* O-MaSE foi possível conceber e formalizar esta metodologia via diagramas UML e outras documentações auxiliares referentes à especificação dos equipamentos.

Palavras-chave: Sistemas multiagentes industriais, metodologia, ontologia, derivação.

ABSTRACT

The problem of derivation of industrial multi-agent systems comprises complex aspects. A derivation basically consists in generate an instance of a multi-agent system from a generic model or architecture with further adaptations for particular cases. In the specific case, this corresponds to a given shop floor, to be further managed by the generated multi-agent system.

The complexity of this problem is mainly concentrate in the intrinsic nature of shop floors, which has a number of aspects that should be taken in to account in the systems modeling phase. The existence of a very heterogeneous legacy systems' structure and industrial equipments should also be very carefully analyzed in order to provide final results (the derived system) coherent with the target shop floor's characteristics, layout and operating rules.

The adopted approach in this work to face such complexity is grounded on the use of a methodology specifically devoted to help in a derivation. This methodology is composed of some steps that were created from the analysis and selection of meta-models of O-MaSE framework. Besides that, two additional steps had to be created (compared to the original framework) to cope with industrial / shop-floor requirements. One step aimed to specify the shop-floor itself and the second step to model the involved knowledge, which in turn is supported by a proper ontology.

The result achieved by the proposed derivation process and adaptations in the O-MaSE framework is represented by the generation of agents' basic code involved the current scenario. By means of O-MaSE it was possible to conceive and to formalize the proposed methodology via UML diagrams and other supporting documentation related to the industrial equipments' specification.

Keywords: Multi-agent industrial systems, methodology, ontology, derivation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

SMA – Sistema Multiagente.
O-MaSE – Organization-based Multi-agent System Engineering.
ADACOR – ADAptive holonic COntrol aRquiteture
FIPA-ACL – Foundation for Intelligent Physical Agent – Agent Communication Language
UML – Unified Modeling Language
AUML – Agent Unified Modeling Language
MASE – Multi-agent Systems Engineering
IDEF – Integrated DEFinition
AGV – Automated Guided Vehicle
CPS – Cyber Physical Systems
MRP – Material Requirement
HMS - Holonic Manufacturing System
SADT - Structured Analisys and Design Technique

SUMÁRIO

1. INTRODUÇÃO

A automatização de chão de fábrica industrial é um processo complexo que tem como objetivo minimizar os custos operacionais e maximizar o lucro na produção. A complexidade da produção industrial reside nas exigências de qualidade dos produtos manufaturados pelos clientes finais. Para viabilizar uma produção industrial otimizada do chão de fábrica, faz-se necessária a elaboração de estratégias de negócio que desenvolvam uma logística de produção planejada para satisfazer a demanda e as restrições dos sistemas de controle.

A integração de sistemas e equipamentos de chão de fábrica, ou seja, robôs, tornos, fresas, veículos autônomos de transportes (AGVs), entre outros, é alvo de pesquisa desde o início do processo de automação industrial (LEE, 2008). Com a evolução das técnicas de produção e dos equipamentos de chão de fábrica, as exigências no setor industrial, tais como agilidade na produção, diversidade de produtos, flexibilidade no fluxo de trabalho, robustez na troca de equipamentos, etc., têm se tornado um desafio para os desenvolvedores de sistemas industriais. Tal desafio consiste no desenvolvimento de sistemas que sejam mais resilientes de tal forma que o sistema de produção seja interrompido o mínimo possível.

Por se tratar de um ambiente composto de uma ampla diversidade de componentes, diferentes protocolos de comunicação e fluxos de produção, o cenário industrial tem como demanda sistemas de informação que sejam adequados às necessidades das indústrias modernas. A padronização de um processo de desenvolvimento de software e aceitação de novos paradigmas na indústria podem ser caracterizados como desafios relevantes no processo de adaptação de tais sistemas.

As fortes mudanças nos paradigmas das empresas de manufatura consequentes da globalização da economia requerem destas a máxima flexibilidade possível dos sistemas (CAMARINHA-MATOS et al., 1997), o que pode ser alcançado utilizando um SMA (Sistema Multiagente). Como a flexibilidade resulta em aumento da agilidade é necessário um sistema de supervisão com um nível de inteligência maior, que por sua vez deve ser (naturalmente) distribuído, o que justifica ainda mais o uso do paradigma de SMA.

Derivar um sistema ou um cenário consiste em traduzir ou transformar todas as funcionalidades, restrições e características presentes no meio para um sistema computacional a partir de um modelo ou arquitetura de referência. Este sistema precisa atender todos os requisitos funcionais e não funcionais presentes no meio. O conceito de derivação de Sistemas Multiagentes (SMA) para chão de fábrica tem sido aplicado em tal contexto com o objetivo de

contemplar as necessidades dos sistemas de informação para ambientes industriais. Esse processo pode ser feito através da virtualização dos componentes da fábrica ou dos sistemas de controle nas camadas operacionais (KARNOUSKOS, 2011).

As dificuldades envolvidas no processo de derivação residem na complexidade inerente à virtualização e representação dos componentes de um chão de fábrica, aplicação de controles e utilização de sistemas computacionais adequadamente. Tal tarefa necessita de informações acerca do cenário representado e do seu fluxo, os quais nem sempre estão disponíveis no instante da derivação. Diante disto, é extremamente relevante considerar a dimensão do chão de fábrica pretendido de forma a verificar a efetiva viabilidade da derivação do sistema e do seu posterior funcionamento.

O estudo apresentado neste documento concentra-se na problemática da derivação de sistemas de chão de fábrica. Foi desenvolvida uma metodologia que analisa os objetivos gerais de produção de uma dada indústria em um contexto semântico com o intuito de obter um SMA que represente tal cenário adequadamente. Para avaliar a proposta, foi derivado um SMA representativo utilizando o conceito de ontologia para representar os elementos do chão de fábrica. O código dos agentes foi gerado com o auxílio de uma metodologia e ferramentas auxiliares.

O desenvolvimento de software se torna mais robusto quando apoiado por uma metodologia que descreva detalhadamente cada etapa do processo de desenvolvimento. Por esse motivo, buscou-se customizar uma metodologia já existente de desenvolvimento de SMA, adaptando-a as características presentes no cenário específico de chão de fábrica.

O sistema computacional de derivação desenvolvido busca implementar as etapas descritas na metodologia, gerando um arcabouço, código básico (esqueleto) de um SMA e sua documentação composta de diagramas UML desenvolvidos na modelagem.

1.1. MOTIVAÇÃO

A complexidade desse tipo de cenário e a crescente evolução e modernização dos sistemas de manufatura requerem de uma forma intrínseca mais inteligência em seu processo de controle e adaptação dos sistemas, proporcionando a possibilidade de adoção de uma tecnologia não procedural e tradicional. A abordagem de SMA é considerada uma das alternativas para suportar esse tipo de implementação.

SMA é uma abordagem utilizada por JENNINGS; CORERA E LARESGOITI (1995) para ambientes industriais e existe muita pesquisa nesse campo, porém poucos trabalhos oferecem um suporte metodológico específico

para derivação de SMAs industriais. Usualmente o usuário-desenvolvedor seleciona uma dada plataforma genérica de SMAs e, ele, conduz todo processo de especificação e implementação de cada um dos agentes envolvidos para o cenário em questão. Isso faz com que o SMA criado (número de agentes, arquitetura, topologia geral da organização, protocolos de comunicação, papéis dos agentes etc) tenham potenciais inconsistências frente ao que se deseja/é necessário, além de tornar o processo de derivação mais lento.

A questão principal de pesquisa desse trabalho se resume em: Como criar SMAs industriais de uma forma que, ao final, este seja coerente com a planta do chão de fábrica, seja completo em relação aos equipamentos envolvidos, seja robusto e interoperável em relação às comunicações envolvidas, flexível no sentido de permitir que um usuário possa introduzir mudanças gerais no sistema e na planta, sem que para isso o sistemas/SMA tenha que ser refeito.

1.2. OBJETIVOS

Os objetivos desse trabalho estão divididos em objetivos gerais e específicos como apresentados a seguir.

1.2.1. Objetivos Gerais

Desenvolvimento de uma metodologia de derivação de SMAs para aplicações de chão-de-fábrica, acompanhado de um ambiente de criação de tais sistemas.

1.2.2. Objetivos Específicos

- Modelar computacionalmente e utilizar uma ontologia de sistema de manufatura.
- Aplicação de uma metodologia de referência para criação de SMAs, adaptando-a a um cenário industrial.

1.3. CONTRIBUIÇÃO CIENTÍFICA

Quando se considera o desenvolvimento de sistemas de controle industriais, é importante que este desenvolvimento seja orientado por uma metodologia, assim como o desenvolvimento de qualquer sistema. Porém, poucos trabalhos aplicam metodologias específicas para o desenvolvimento de SMAs industriais. Observando-se essa deficiência, buscou-se aperfeiçoar o processo de desenvolvimento, customizando uma metodologia já existente que

visa o desenvolvimento de SMAs tradicionais, utilizando-se de uma ontologia para representação do conhecimento como parte integrante da metodologia, tornando-a específica para um cenário de chão de fábrica.

Optou-se por usar uma ontologia genérica que está baseada em uma taxionomia de componentes de manufatura, proporcionando à metodologia a padronização de nomenclaturas e classes de representação de cenários industriais.

A maior contribuição deste trabalho está na integração de um framework de desenvolvimento de SMAs já consolidado (O-MaSE) e utilizado por diversos projetos de software com uma ontologia voltada para ambientes de manufatura (ADACOR), consolidados na forma de uma metodologia. Outra contribuição se concentra na geração do arcabouço do SMA disponibilizado nos códigos gerados, suporte à comunicação FIPA-ACL, métodos específicos para implementação dos comportamentos dos agentes e biblioteca de suporte de gerenciamento do JADE, aspectos esses abarcados e mantidos do O-MaSE.

1.4. ORGANIZAÇÃO DO TRABALHO

Esta dissertação está estruturada em sete capítulos, os quais são introduzidos da seguinte forma.

Este capítulo 1 abrange uma breve apresentação do cenário industrial ao qual o trabalho se relaciona. A motivação dessa pesquisa, seus objetivos e contribuição científica agregada.

No capítulo 2 é apresentado o referencial teórico básico do trabalho e no capítulo 3 é feita a revisão da literatura sobre sistemas multiagentes e metodologias de suporte.

No capítulo 4 é apresentada a metodologia desenvolvida de suporte à derivação de SMAs industriais, em que também é apresentada a ontologia base deste trabalho.

No capítulo 5 é apresentado o Sistema de Derivação e sua aplicação em um determinado cenário.

No capítulo 6 são apresentadas a análise, avaliações, dificuldades e soluções geradas a partir deste trabalho.

No capítulo 7 apresenta as conclusões da dissertação, sintetizando os resultados alcançados e as perspectivas de trabalhos futuros.

2. REFERENCIAL TEÓRICO

Este capítulo apresenta as teorias de base para entendimento dos conceitos de sistemas multiagentes, metodologias direcionadas para desenvolvimento de sistemas, e ontologia.

2.1. SISTEMAS MULTIAGENTES (SMAs)

Os SMAs são uma estratégia da Inteligência Artificial Distribuída (IAD), de onde se derivou o conceito SMA, com a finalidade de decompor um sistema grande e complexo em problemas menores, módulos (JENNINGS, 1994; SILVEIRA, 2006). Um agente é um sistema computacional que está situado num ambiente dinâmico e é capaz de exibir autonomia e comportamento inteligente. Um agente pode estar inserido em um ambiente composto de diversos agentes que se comunicam entre si formando uma comunidade de agentes, um SMA (MONOSTORI; VÁNCZA; KUMARA, 2006).

Quando se trabalha com situações reais, lidar com as proporções do problema pode ser um grande desafio. Esse problema pode ser dividido em diversos subproblemas mais simples. Utilizando a abordagem de agentes, cada agente pode ser responsável por um subproblema e juntos chegarem a solução deste.

2.1.1. Definição de Agente

Um agente é definido como sendo um sistema de software que se comunica e coopera com outros sistemas de software (agentes ou não) para resolver um problema ou uma tarefa complexa que está além da capacidade de resolução por um único software (agente ou não) de um componente independente deste sistema. Essa abordagem também é conhecida como "dividir para conquistar", princípio adotado para resolução de problemas complexos. Quando vários agentes são utilizados para resolver um problema complexo, pode-se dizer que esse sistema se trata de um SMA.

Definição apoiada por SHEN, HAO, JOONG E NORRIE (1998), diz que um agente é um sistema computacional situado em algum ambiente e que é capaz de ações autônomas neste ambiente a fim de atender seus objetivos de design. Um agente autônomo deve ser capaz de agir sem a intervenção direta de seres humanos ou outros agentes e deve ter controle sobre suas próprias ações e estados internos.

Segundo CAMARINHA-MATOS; BARATA E FLORES (1997), um agente é caracterizado por um comportamento autônomo que inclui as capacidades de comunicação, negociação e cooperação. Essas características do agente permitem o trabalho em conjunto com seus parceiros, tendo como meta, alcançar um objetivo comum, salientando que um agente puramente reativo, como os agentes que representam os sensores, a autonomia está caracterizada em puramente realizar uma execução pré-definida, assim como ler os dados de um determinado sensor e comunicar com o agente responsável pela coordenação do seguimento de produção.

Um agrupamento de agentes, como mencionado no início deste subcapítulo, é considerado um SMA, onde cada agente deve possuir algumas características como (JENNINGS E WOOLDRIDGE, 1998):

- Reatividade – Consegue responder ao ambiente em um tempo adequado.
- Pró-atividade – Faz o que pode para tentar atingir os objetivos aos quais foi projetado.
- Sociabilidade – Interage com os demais agentes do SMA para atingir os objetivos.

2.1.2. Agentes Industriais e suas características

Segundo WOOLDRIDGE e JENNINGS (1995); e JENNINGS (1994), o comportamento de um agente deve possuir as seguintes características:

- Semiautônomo – certo grau de autonomia, porém dependendo de outros agentes para concluir suas tarefas;
- Interação – interação com outros agentes visando a solução de um problema;
- Independente – conhecimento suficiente para concluir algumas tarefas;
- Cooperante – coopera com a organização, não possui comportamento destrutivo ou egoísta, busca o bem comum e solução do problema;
- Benévolo – interesse sempre em cooperar;
- Honesto – não oculta e fornece informações sempre verdadeiras, interagem com outro agente;
- Responsável – ao assumir um compromisso, aplica todos os esforços ao seu alcance para cumprir uma tarefa.

Ao observar as características de um ambiente industrial é possível fazer um paralelo com SMA, pelo fato de que cada componente do chão de

fábrica pode ser representado por um ou mais agentes. Assim, o uso de multiagentes é considerado uma adequada escolha de programação de software que irá representar o cenário.

As características mais importantes que justificam a utilização de SMAs em um ambiente industrial são:

- Complexidade
- Ambiente distribuído
- Integração

Segundo MONOSTORI, VÁNCZA e KUMARA (2006), o conceito de SMA industrial está mais ligado à integração. A integração promovida pela adoção de multiagentes possibilita continuidade operacional de um determinado setor da empresa, caso haja algum colapso de conectividade (tolerância a falhas).

Segundo OLIVEIRA (2005) um sistema de manufatura ágil, que consiga atender as necessidades de mercado, como mencionado no início deste trabalho, precisa atingir todos os setores do chão de fábrica de forma efetiva, ou seja, integração a nível de processos, software e hardware.

A integração é muito mais do que um problema de interconexão de componentes físicos e aplicações de software. Utilizando métodos e ferramentas, é possível explorar esta propriedade para atingir com maior facilidade os objetivos da empresa.

A comunicação utilizada para gerenciamento do sistema tradicional é um aspecto que precisa ser trabalhado com cuidado neste cenário. Quando o assunto é integrar, a complexidade de comunicação entre os vários níveis do chão de fábrica é uma questão que deve ser muito bem analisada.

Em JENNINGS (1994) e RABELO (1997) são abordadas características como cooperação, execução assíncrona, rapidez na execução e obtenção de resultados (execução distribuída de processo). Estas características agregam robustez ao sistema.

Outra importante questão que deve ser tratada por um SMA industrial são os sistemas legados, adotando um paradigma de cliente-servidor entre agente e controlador físico. O agente deve encapsular esse sistema físico em uma forma de máquina virtual abstrata no ambiente multiagente. A ideia é integrar as funcionalidades da máquina abstrata às reais ações executadas pelo controlador físico. Resumidamente, implementar uma imagem de cada controlador físico em uma máquina abstrata que por sua vez é representada por um agente (CAMARINHA-MATOS et al., 1997).

2.2. SMAS INDUSTRIAIS

Sistemas inteligentes direcionados a ambientes industriais são alvo de pesquisa há mais de duas décadas. Porém, a aplicação e estudo da prática dessas tecnologias têm se restringe muito ainda ao meio acadêmico.

Um dos fatores que dificultam o conhecimento da real evolução da aplicação de tecnologias multiagentes e de sistemas inteligentes no processo industrial, especificamente multiagentes, é pelo fato que as industrias adotam sistemas particulares e fechados, inviabilizando a divulgação da tecnologia utilizada em determinados chãos de fábrica, ou seja, protecionismos e segredos industriais, que proporcionam uma competição de mercado.

A evolução dos sistemas de software direcionados ao cenário fabril que utilizam a tecnologia de sistemas multiagentes será abordado em ordem cronológica, demostrando o crescimento, os benefícios e as conquistas alcançadas por estes.

De acordo com RABELO e CAMARINHA-MATOS (1994), o escalonamento da manufatura (*scheduling*) é a atividade responsável por atribuir tarefas dos recursos de manufatura da produção em um intervalo específico de tempo. A produção é um cenário dinâmico, em que novos produtos são produzidos e novos equipamentos estão em constante atualização, o trabalho abrange o agendamento de tais tarefas de acordo com a estrutura organizacional dos equipamentos, dando um suporte a negociação e flexibilidade de produção. Um protótipo chamado HOLOS foi desenvolvido utilizando Prolog e linguagem AIX em uma plataforma IBM RISC 6000.

Em WYSK e SMITH (1995) é exposta a preocupação com a parte de controle de chão de fábrica que é resolvido com estratégias e planos de controle de ações funcionais. O trabalho desenvolveu um sistema formal de controle funcional para a parte discreta dos sistemas de manufatura.

Em CAMARINHA-MATOS; BARATA e FLORES (1997) são abordados soluções de integração de software e hardware para sistemas de supervisão e manutenção de softwares para controladores de robôs industriais de soldagem. Neste trabalho são utilizados arquiteturas multiagentes para preencher os requisitos do chão de fábrica, onde se concluiu que o sistema proposto de supervisão multiagente tem um bom desempenho apesar de necessitar de mais testes que comprovem sua eficiência.

LEITÃO e RESTIVO (2009) propõem uma metodologia de desenvolvimento ADACOR que compreende um conjunto de passos para identificar os *holons* presentes no chão de fábrica. Os holons remetem ao

conceito introduzido por (RABELO, 1997), em que todo equipamento ou processo pode ser interpretado e modelado como um agente e consórcios destes para a realização de atividades conjuntas. Esses holons devem possuir papéis, responsabilidades, comportamentos e interação com outros holons, obtendo então uma estrutura holônica que represente o chão de fábrica.

Em BORGO e LEITÃO (2004) foi abordado que uma ontologia tem um papel muito importante na construção de sistemas e aplicações reutilizáveis e adaptáveis com destaque na área de controle de chão de fábrica. O resultado obtido mostrou as vantagens do uso de ontologia, com destaque nas questões relacionadas à comunicação e partilha de informações, tornando a comunicação realizada entre os sistemas mais completa, com maior detalhamento das informações.

Em OLIVEIRA (2005) foi desenvolvida a arquitetura CoBASA, criada para apoiar as adaptações rápidas às mudanças de arquitetura de controle de chão de fábrica. Sua arquitetura é definida em módulos de software, agentes. Estes agentes assumem contratos criados para atender determinada demanda. Caso houver alguma alteração no meio, basta reconfigurar os contratos estabelecidos entre os agentes.

PESCHKE; LUDER; KUHNLE (2005) desenvolveram uma nova abordagem para sistemas industriais para melhorar a flexibilidade na produção. Como forma de representar a comunicação e representação dos agentes e com o intuito de integrar globalmente os diversos níveis, foi empregada uma ontologia. O projeto é baseado na metodologia GAIA. Foi observado um aumento na flexibilidade e adaptabilidade da estrutura do sistema de controle melhorando assim a eficiência na produção e a possibilidade de integração.

LÜDER (2007) apresenta a evolução da arquitetura de desenvolvimento de sistemas multiagentes baseada em GAIA, chamada Pabadis Promise. O trabalho demonstra uma visão mais detalhada em relação às funcionalidades e distribuição das responsabilidades aos agentes, o que proporciona a possibilidade de adaptação a diferentes necessidades do sistema de produção. O conceito de atribuição de papéis aos agentes, proposta pela metodologia GAIA, mostra-se eficiente, definindo claramente as funcionalidades e responsabilidades atribuídas a cada agente.

SHEN, HAO, JOONG e NORRIE (1998) trazem como destaque o uso da tecnologia de agentes como um paradigma promissor para sistemas de chão de fábrica da nova geração de sistemas de controle de chão de fábrica. Problemas referentes ao uso de sistemas multiagentes, como encapsulamento, organização de agentes, coordenação, negociação, dinâmica de processo, aprendizagem, otimização, segurança e padrões utilizados entre agentes são discutidos. Os resultados obtidos destacaram que a falta de padrões no

desenvolvimento de soluções de sistemas ainda é escassa, tendo em vista que diversas tecnologias são propostas, como RFID (Radio-Frequency IDentification), Sistemas de ERP e MRP, Web semântica, Grid Computing, em que todas necessitam estabelecer um padrão de integração para que o uso de sistemas multiagentes possam ter os resultados esperados.

Em CÂNDIDO e BARATA (2007) se descreve uma forma de agentificação de chão de fábrica que propõe maior agilidade e adaptabilidade à nova visão, cada vez mais volátil das indústrias. A implementação desenvolvida transforma cada componente de chão de fábrica em um agente, aumentando a capacidade de adaptação e interação com o ambiente. Uma ontologia é utilizada para assegurar uma troca de mensagens confiável. A abordagem também mostra como adaptar sistemas legados. Como resultado foi obtida a modularização dos componentes agregando a eles inteligência, dando suporte a rápida atualização e reconfiguração no processo de produção. A adaptação dos sistemas legados foi realizada utilizando um invólucro que encapsula os detalhes de comunicação.

LOSS (2012) propõe uma abordagem para facilitar a comunicação entre sistemas industriais e dispositivos de chão de fábrica. O ambiente proposto é aberto, utiliza padrões e uma integração transparentes como CORBA, KQML, MMS e XML em todos os níveis de comunicação, ou seja, proporciona uma comunicação de alto nível desde os equipamentos de chão de fábrica até os sistemas de gerenciamento de produção.

PEREIRA at al., (2012) abordaram os desafios para se desenvolver SMAs para o ambiente industrial focando na integração dos agentes com equipamentos físicos. Neste trabalho, foi desenvolvido um SMA utilizando a plataforma JADE para um sistema smart grid elétrico. O resultado obtido evidenciou que há uma complexidade maior para o desenvolvimento de sistemas multiagentes para o ambiente industrial quando comparado com soluções tradicionais, principalmente devido às limitações providas pelos equipamentos presentes no chão de fábrica.

2.3. AMBIENTE INDUSTRIAL

Um ambiente industrial de manufatura pode ser dividido sistematicamente de acordo com a Figura 2.1 e sua definição de manufatura, segundo FRANCO (2003):

"*Manufatura é a transformação de matéria-prima, em seus diversos estados, em produtos finais, para serem disponibilizados para o consumidor final. Os produtos são gerados por uma combinação de trabalho manual, máquinas, ferramentas especiais e energia, sendo, então, disponibilizados para o consumidor final*".

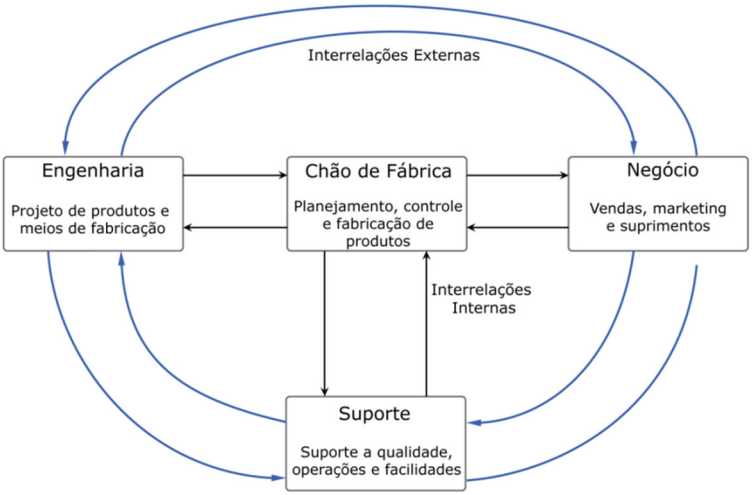

Figura 2.1. Sistema de manufatura

O chão de fábrica, que é objeto de análise desse trabalho, é caracterizado pela presença de inúmeros componentes que trabalham cooperativamente para atingir os objetivos de produção da empresa, podendo ser, por exemplo, beneficiamento de matéria-prima. Devido à diversidade de componentes existentes em um chão de fábrica, fica intrínseca a complexidade do sistema de controle e gerenciamento do mesmo.

Os componentes desse ambiente podem ser classificados como físicos e lógicos.

Componentes físicos:

- Máquinas (tornos, fresadoras, etc.);
- Sensores (componentes de leitura do meio);
- Atuadores (robôs, braços mecânicos, etc.);
- Computadores (servidores, micro computadores, etc.);
- Transportadores (AGVs, esteiras, etc.);

Componentes lógicos:

- Programas de controle (sistema de gerenciamento da produção);
- Protocolos de comunicação;
- Bases de Dados (Banco de dados, Data Warehouse, etc.).

Para integrar os componentes presentes nesse ambiente, é necessário identificar as dependências existentes entre eles, o fluxo de trabalho e as tarefas que cada um deve desempenhar na organização (LEITÃO; RESTIVO, 2009).

Deve-se considerar que esse tipo de ambiente está sempre suscetível à quebra de equipamentos ou reposição por equipamentos diferentes ou mais modernos, o que pode acarretar em alteração ou adaptação de outros componentes de produção já existentes no meio. Outro problema é interrupção do funcionamento de algum módulo do sistema gerencial ou substituição de algum módulo.

Outro fator importante, que aumenta a complexidade do sistema, está relacionado ao controle do sistema que gerencia a produção, pois envolve uma grande diversidade de equipamentos heterogêneos e grande volume de troca de mensagens entre equipamentos, sensores, atuadores e os próprios programas de gerenciamento da produção.

A integração desses componentes também é um fator determinante nesse ambiente, pois devem trabalhar de uma forma cooperativa, a fim de alcançarem o objetivo final de produção.

Algumas das principais características de um chão de fábrica são:

- Natureza distribuída;
- Sistemas heterogêneos;
- Variedade de equipamentos e;
- Protocolos de comunicação particulares.

Levando em consideração as principais características previamente mencionadas, a integração resulta em uma arquitetura de controle hierárquica rígida cuja complexidade estrutural cresce rapidamente de acordo com o tamanho do escopo do sistema (MONOSTORI; VÁNCZA; KUMARA, 2006). Mudanças no chão de fábrica, assim como alterações na demanda do mercado causam instabilidade no ambiente que devem ser considerados no processo de desenvolvimento, a volatilidade das condições de mercado desfavorecem a rigidez da arquitetura hierárquica e consequentemente a programação, mesmo em sistemas computacionais sofisticados.

A Figura 2.2 representa um ambiente industrial através de uma planta fabril, ilustrando alguns componentes mencionados no início desse capítulo.

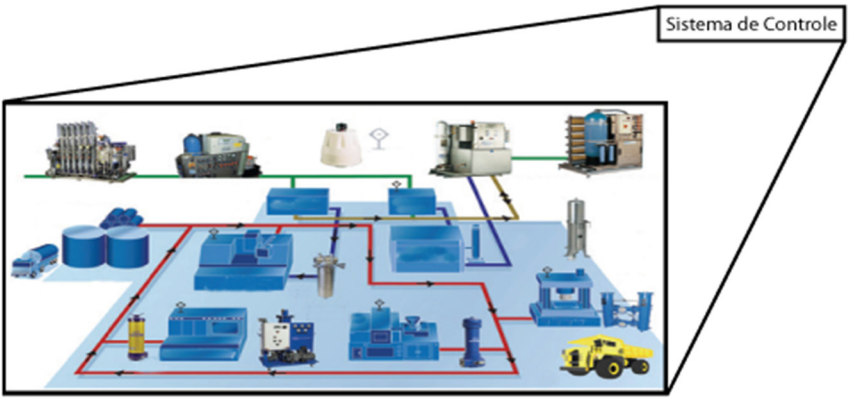

Figura 2.2. Chão de fábrica

Requisitos que o sistema de manufatura deve atender, segundo SHEN, HAO, JOONG e NORRIE (1998):

- Integração completa de programas heterogêneos e sistemas de hardware dentro da empresa.
- Arquitetura de sistema aberto para acomodar novos subsistemas ou substituir os existentes.
- Comunicação eficiente, eficaz e cooperação entre departamentos dentro da empresa e entre empresas.
- Incorporar fatores humanos no sistema de manufatura.
- Resposta rápida às mudanças de ordem externa e mudanças inesperadas no ambiente de produção.
- Tolerância a falhas, tanto em nível de sistema, quanto em nível de subsistema, de forma a detectar e recuperar o sistema de falhas, minimizando os seus impactos sobre o ambiente de trabalho.

Outras questões presentes na realidade atual da manufatura são (CÂNDIDO; BARATA, 2007):

- O tempo para concepção e instalação do sistema é muito longo;
- A fase de reengenharia é complexa e demorada;
- Aplicações extremamente complexas e hierárquicas;
- Escalabilidade;
- Tolerância a faltas;
- Incompatibilidade entre equipamentos de diversos fornecedores e sistemas legados;
- Chão de fábrica isolado de ambientes de alto nível de negócio.

2.5. VIRTUALIZAÇÃO DE SISTEMAS

A integração entre componentes físicos e sistemas computacionais não é uma técnica nova (LEE, 2008). Sistemas embarcados que possibilitam controle e manipulação de informações de equipamentos estão presentes há bastante tempo e formam uma linha de pesquisa muito importante. Com os avanços nas áreas de comunicação, rede, circuitos integrados e equipamentos, o conceito de Cyber-Physical System (CPS) tem ganhado força e bons resultados na implementação de soluções de virtualização (LEE, 2008). A virtualização dos recursos de chão de fábrica (hardware) pode ser um exemplo desse tipo de solução (KARNOUSKOS, 2011).

Juntamente ao conceito de CPS, ou seja, virtualização de equipamentos, a solução de disponibilizar as funcionalidades e características dos componentes industriais pode ser feito via servidor de serviços, em que os agentes que representam os equipamentos no SMA possam interagir com o servidor de serviços, que por sua vez se comunica com os componentes físicos do ambiente industrial.

A construção do SMA que represente o ambiente industrial também precisa se preocupar com a disposição dos agentes e o papel que cada agente deve possuir dentro da organização. Esses papéis representam as restrições comportamentais dos agentes, que se referem a compromissos assumidos dentro do grupo SMA entre agentes e com relação a execução de tarefas (HÜBNER, 2003). Fazendo um paralelo com um agente que representa uma máquina, esse possui um papel que se compromete com outro agente do SMA a executar uma determinada tarefa e a ele é atribuído um papel que representa uma tarefa de controle da máquina. Esse nível hierárquico existente entre os agentes de um SMA é característica de uma organização, onde agentes assumem papéis e compromissos. A Figura 2.3 ilustra um tipo de disposição organizacional que pode ser aplicado ao chão de fábrica, em que o ambiente se refere ao chão de fábrica e duas organizações representam duas células de manufatura capazes de se comunicar entre si de uma forma hierárquica. Os compromissos assumidos pelos agentes também podem ser observados nesta Figura.

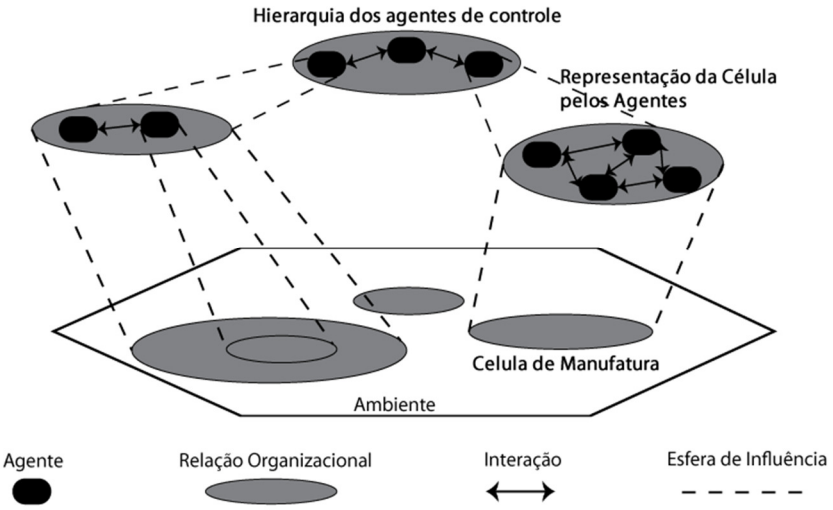

Figura 2.3. Esquema SMA interagindo com o ambiente.

3. REVISÃO DA LITERATURA

Este capítulo tem o intuito de apresentar a evolução das pesquisas relacionadas a metodologias e ontologias.

Primeiramente será apresentado três metodologias e na sequência duas ontologias com suas respectivas introduções.

3.1. METODOLOGIAS

Uma metodologia é definida como uma sequência de etapas, seguidas rigorosa e sistematicamente ao decorrer do desenvolvimento do projeto. Ela é responsável por conduzir a progressão do projeto, servindo como uma ferramenta para orientar o seu desenvolvimento.

PRESSMAN e ROGER (1995) define que metodologia é um conjunto de atividades, ações e tarefas realizadas quando algum artefato de software deve ser criado. Ela determina o relacionamento existente entre os processos de software e o relacionamento entre cada atividade, ação ou tarefa.

As etapas de um projeto de desenvolvimento de software bem elaborado devem seguir uma metodologia como um meio de assegurar confiabilidade no produto desenvolvido e corretude nas suas funcionalidades.

O uso de metodologias para o desenvolvimento de softwares foi utilizado desde o início desta atividade, ou seja, uma sequência de passos, etapas, naturalmente é seguida afim de se obter um software que atenda as necessidades.

Como exemplo pode-se mencionar o desenvolvimento de software. Este desenvolvimento não é iniciado escrevendo-se um código de programação. O cenário deve ser muito bem estudado e analisado anteriormente, a fim de obter-se o maior número possível de informações do cenário e requisitos. A construção da estrutura do projeto não pode ser iniciada sem que haja informações consistentes e corretas em relação aos requisitos do sistema. A modelagem dos requisitos precisa de parâmetros do projeto para que se possa desenvolver um produto viável e útil, trazendo benefícios para quem for utilizar o sistema. Após todas essas sequências de passos/etapas, pode ser inicializado o processo de programação, mesmo que essas etapas possam ocorrer concomitantemente e de forma faseada.

3.2. METODOLOGIAS PARA DESENVOLVIMENTO DE SISTEMAS MULTIAGENTES

Atualmente existem diversas metodologias para desenvolvimento de SMAs, cada uma com suas características e limitações. Algumas características são comuns, por exemplo:

- Em algum momento os agentes são definidos.
- O protocolo de comunicação é definido.
- Qual o papel que cada agente deve exercer.

Essas qualidades podem ser observadas nas metodologias apresentadas a seguir, que foram utilizadas como referências para este trabalho.

3.2.1. Gaia

Primeira metodologia feita com o intuito de guiar o processo de desenvolvimento de SMAs, proposta por (WOOLDRIDGE at al., 2000). A primeira versão desta metodologia é constituída de cinco etapas: Modelo de Papéis, Modelo de Interações, Modelo de Agentes, Modelo de Serviços e Modelo de Habilidades.

Embora esta metodologia tenha trazido uma proposta interessante de sistematização do desenvolvimento de SMAs, a mesma não incorpora as fases de especificação e implementação do sistema. Desta forma, os modelos de papéis e de interação dos agentes são definidos ainda na fase de análise, enquanto os demais são definidos apenas na fase de Design.

3.2.2. MaSE

A metodologia MaSE, criada em 2004, tem como objetivo modelar sistemas SMAs heterogêneos usando basicamente modelos gráficos baseados nos padrões de modelagem UML para descrever os tipos de agentes do sistema e as interfaces entre os agentes. Nesta, o SMA é visto como uma abstração do paradigma de orientação a objetos e os agentes são vistos como objetos especializados (DELOACH, 2004).

MaSE é uma metodologia ampla, que contempla o processo de desenvolvimento de SMAs desde a especificação e análise de requisitos do sistema até a implementação. Tem como uma das características gerar muitos agentes para representação dos cenários.

3.2.3. HOLOS

A metodologia HOLOS é composta por um conjunto de procedimentos e considerações que devem ser seguidos visando a implantação de um sistema de escalonamento para uma empresa, que sugere a existência de um sistema informático de apoio a derivação (RABELO, 1997).

Ela é totalmente voltada para desenvolvimento de SMAs para escalonamento de atividades do chão de fábrica.

A metodologia HOLOS é constituída de cinco fases sequencias, em que cada fase possui passos que devem ser seguidos para completar determinada fase. As etapas da metodologia são (RABELO, 1997):

- Analisar e Preparar, dividida em dois passos: Analisar a empresa e preparar a informação, e Integrar;
- Instanciar, dividida em três passos: Caracterizar o sistema, Configurar a arquitetura do sistema, e Instanciar os agentes;
- Validar, que se resume ao passo de Validação do sistema derivado;
- Executar, que se resume ao passo de Executar, lançando os agentes derivados;
- Avaliar, dividida em três passos: Analisar o desempenho, Reconfigurar a arquitetura, e Reconfigurar a arquitetura genérica.

Embora seja uma metodologia de derivação de sistemas multiagentes particulares específicos direcionada para chão de fábrica, ela faz uso de protocolos proprietários para comunicação entre os agentes (embora use o padrão MAP/MMS para comunicação com os controladores dos recursos de manufatura), não se apoia em nenhuma outra metodologia ou framework para SMAs, e não faz uso de ontologias de suporte.

3.3. ONTOLOGIA

Segundo GRUBER (1995), uma ontologia é definida como sendo a especificação explícita da conceituação de um determinado domínio em estudo, conceito utilizado pela filosofia, em que Ontology é a explicação sistemática da existência. Basicamente, no âmbito da pesquisa em questão, uma ontologia consiste dos conceitos, relações, suas definições, propriedades e restrições existentes no domínio em questão.

Uma ontologia tem a função de classificar os objetos do meio, ou seja, representar a “taxionomia” de um determinado cenário que está sendo estudado (STUDER et al, 1998). Este processo pode envolver um número considerável de profissionais multidisciplinares para que não haja ambiguidade de modelagem e interpretação.

De acordo com OLIVEIRA (2005), ontologia proporciona o comum entendimento dos conceitos, servindo como uma ferramenta importante na integração de diversas áreas e domínios. A padronização dos conceitos facilita a troca de informações e conhecimentos, sendo de particular ajuda quando da comunicação entre sistemas heterogêneos.

Em um sistema multiagente industrial, normalmente, cada agente representa um equipamento ou um processo de gerenciamento, portanto cada agente tem uma visão específica do domínio. Uma ontologia tem a função de integrar essas perspectivas distintas agrupando os conjuntos de conceitos, conhecimentos, relações e restrições, para então formar uma ontologia comum, em que os agentes possam ter uma visão ampla do domínio (FALASCONI et al, 1996).

De forma geral, uma ontologia pode ser desenvolvida para diversos fins e atingir os seguintes propósitos:

- No desenvolvimento de uma ontologia, as pessoas envolvidas enfrentam o desafio de explicar seu entendimento sobre o domínio, o que facilita o entendimento da área de conhecimento, que futuramente pode ser analisado por outros profissionais.
- Geralmente existem diversas interpretações sobre os conceitos envolvidos em uma área de conhecimento. Diferentes especialistas com diferentes entendimentos sobre determinado domínio levam a um problema de comunicação. A ontologia busca um consenso sobre os significados, ajudando a chegar a uma melhor compreensão destes.
- Uma ontologia sobre determinada área de conhecimento deve permitir que uma pessoa que queira aprender mais sobre o domínio não precise recorrer a um especialista, mas pode também aprender sobre o domínio estudando a ontologia desenvolvida, que detém o conhecimento geral e de consenso, promovendo o aprendizado sobre essa área de conhecimento específica.

3.3.1. Estrutura de uma Ontologia.

A estrutura de uma ontologia é composta da seguinte forma:

- Classes: As classes representam um conjunto que contém indivíduos. Descrição formal dos requisitos que possa representar um indivíduo.
- Instâncias: As instâncias representam os objetos do domínio, nos quais se tem interesse em descrever.

- Atributos: Atributos representam as características que uma instância possui.
- Propriedades: As propriedades representam a relação binária existente entre indivíduos, que também podem ser identificados como *slots*.

Dentre as ontologias estudadas, duas tiveram destaque por serem direcionadas para o ambiente fabril e possuírem uma taxionomia que permite representar um grande número de cenários de chão de fábrica.

3.3.2. Ontologia MASON

A proposta da ontologia MASON (MAnufacturing's Semantics ONtology), consiste em três conceitos principais: entidades, operações e recursos. Utiliza a ferramenta Protégé para representar a ontologia.

O principal objetivo dessa ontologia é proporcionar compreensão compartilhada e comum do domínio de chão de fábrica, ou seja, uma semântica comum do ambiente de manufatura.

De acordo com KOMMA (2012) as ligações entre os objetos que representam um sistema de manufatura devem consistir de um sistema de informação, regras e um dicionário comum, que possibilita representar os relacionamentos e o que cada instância representa no cenário.

3.3.3. Ontologia ADACOR

A ontologia de controle de manufatura ADACOR está baseada na taxionomia dos componentes de manufatura, contribuindo para o formalismo e compreensão do problema de controle de manufatura (CASAIS et al., 2011). Ela é representada por um diagrama UML de classes.

A ontologia ADACOR faz parte de uma arquitetura de desenvolvimento de sistemas multiagentes para o setor de manufatura. A arquitetura ADACOR é baseada no paradigma Holonic Manufacturing System (HMS), voltado para problema de controle distribuído de manufatura (LEITÃO; RESTIVO, 2008).

A estrutura padrão da ontologia ADACOR é representada pela Figura 3.1, em que é observada a taxinomia utilizada para representar os equipamentos presente em um ambiente industrial (BORGO; STEFANO; LEITÃO; PAULO, 2004).

Como se verá no próximo capítulo, escolheu-se utilizar a ontologia ADACOR no presente trabalho pelo fato de que seus atributos seguem melhor

a padronização dos conceitos industriais e a facilidade de compreensão de sua estrutura.

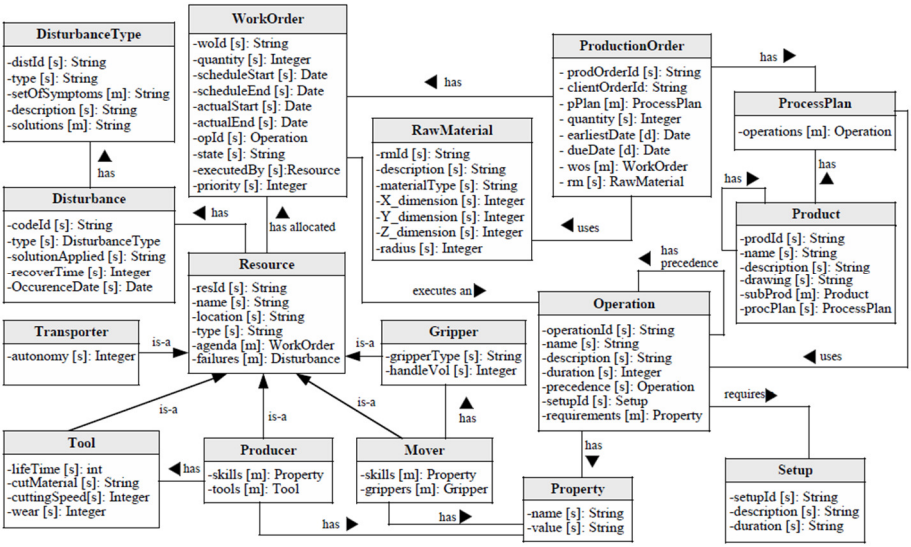

Figura 3.1. Estrutura da ontologia ADACOR

3.4 SUMÁRIO

Os conceitos-base de metodologias e ontologias foram abordados. É observado os benefícios que uma metodologia bem estruturada tem o potencial de proporcionar no momento de desenvolver um sistema multiagente para um cenário específico como um chão de fábrica e como uma ontologia direcionada para ambientes fabrís pode auxiliar no processo de integração de um ambiente heterogêneo.

No próximo capítulo será apresentada a metodologia proposta.

4. A METODOLOGIA DE DERIVAÇÃO

No decorrer deste trabalho buscou-se apresentar a complexidade de um cenário de chão de fábrica levando em consideração os estados dos processos, a organização formada pelos equipamentos (máquinas, robôs, transportadores, etc), o fluxo de informações, entre outros aspectos. Também foram abordadas metodologias clássicas de modelagem de sistemas multiagentes e, por fim, foi apresentada uma ontologia genérica a qual será adotada como base para a representação do conhecimento no contexto deste trabalho.

Observou-se que há a necessidade de uma metodologia que seja específica para ambientes industriais e que utilize uma tecnologia de representação do conhecimento flexível, adaptável e de fácil reutilização, tendo em vista o nível de complexidade presente nos ambientes industriais e as metodologias de modelagem de sistemas multiagentes existentes.

O desenvolvimento de sistemas baseados em uma metodologia oferece uma confiabilidade melhor, considerando que ela ajuda como um guia e sistematização do processo assim como permite envolver e explicitar todos os recursos e aspectos necessários numa derivação.

Como referência de desenvolvimento desta metodologia, várias metodologias e frameworks foram analisados, cada uma com suas características específicas. Porém, a O-MaSE (*Organization-based Multiagent System Engineering*), proposta por DELOACH (2004), fornece uma estrutura bem definida de meta-dados proporcionando uma ferramenta que permite construir metodologias customizadas para cenários específicos. Assim, julgou-se a O-MaSE como a melhor opção de apoio ao processo de derivação. Ela é derivada da metodologia MaSE.

A solução desenvolvida neste trabalho de dissertação procurou atender aos seguintes requisitos básicos:

- Possuir uma ontologia de sistemas de manufatura como base;
- Possibilitar a inclusão de novos componentes ao sistema, sem que para isso ele tenha que ser reescrito;
- O código gerado pelo "Sistema de Derivação" deve fornecer suporte para que todos os agentes sejam adaptados à realidade do cenário que está sendo modelado.

O objetivo específico deste trabalho é construir um sistema computacional que, a partir dos requisitos do cenário de chão de fábrica, da planta e das especificações dos equipamentos e fluxo de trabalho, possa gerar o "esqueleto" do SMA.

A Figura 4.1 ilustra o trabalho e deve ser basicamente interpretada da seguinte forma: dado um determinado chão de fábrica, construa um (esqueleto

de) SMA (i.e. os seus agentes, sua hierarquia de controle e protocolos de comunicação) que represente o cenário.

A ideia da dissertação é que essa construção seja gráfico-interativa, guiada por um sistema computacional (um ambiente com várias ferramentas, como Eclipse, o *framework* O-MaSE, Jade, Protégé, entre outras) e guiada por uma metodologia, que por sua vez utilize uma ontologia (genérica) de sistemas de manufatura em uma das etapas. Esta ontologia deverá ser modelada de acordo com o cenário, a fim de auxiliar na definição dos agentes que serão gerados.

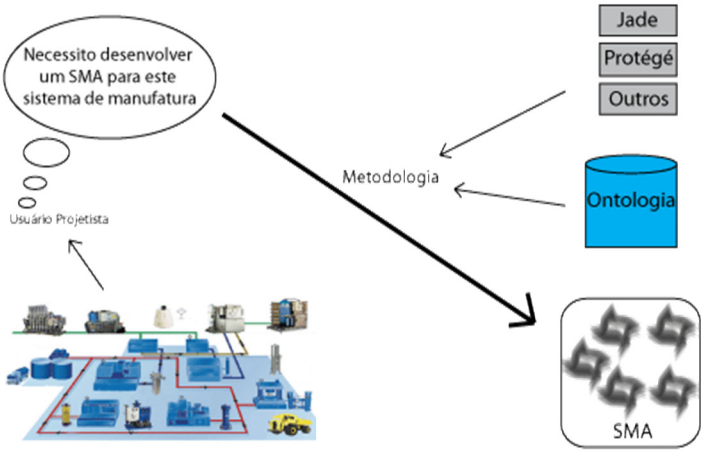

Figura 4.1. Requisitos Gerais.

4.1. FRAMEWORK O-MASE

No contexto O-MaSE, diagramas UML de classe, estado e sequência são, por exemplo, utilizados para representar as etapas de desenvolvimento do projeto da metodologia proposta neste trabalho.

O *framework* O-MaSE tem como principal objetivo permitir que o projetista construa uma metodologia customizada orientada a agentes, baseada em um conjunto de fragmentos de métodos. Os fragmentos de métodos definem como um conjunto de produtos de análise e projeto podem ser criados e usados (DELOACH, 2007) e (GARCIA-OJEDA et al, 2008). A Figura 4.2 ilustra os fragmentos de métodos do framework.

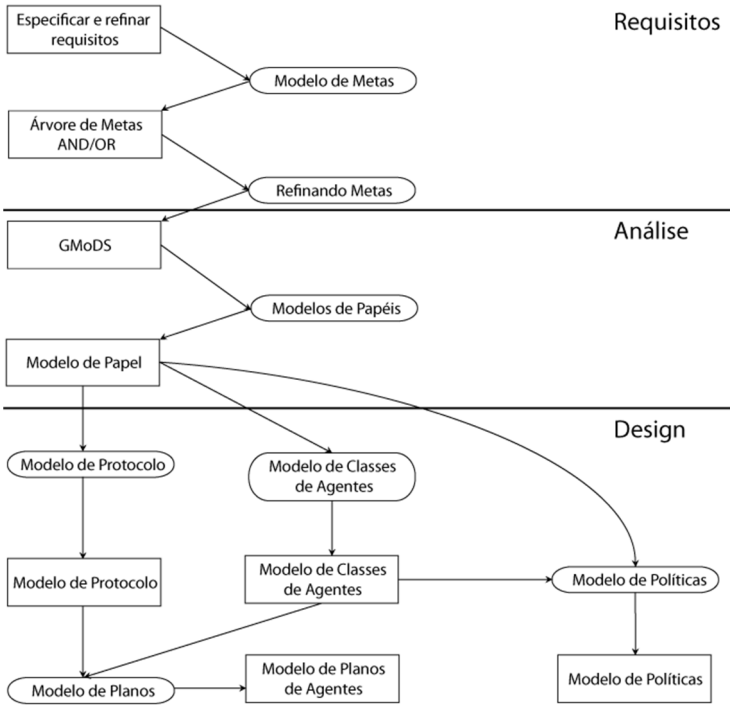

Figura 4.2. Sequência de etapas em uma modelagem SMA tradicional (GARCIA-OJEDA et al, 2008).

A Figura 4.2 é um exemplo de metodologia construída utilizando o *framework* O-MaSE, sendo ela composta de oito etapas e devendo seguir a sequência indicada pelas setas. Cada etapa deste exemplo gera um modelo, que pode ser representado graficamente no O-MaSE através de diagramas baseados em UML.

Para adaptar a estrutura do O-MaSE aos objetivos deste trabalho, algumas mudanças foram adotadas no exemplo da Figura 4.2. Etapas foram retiradas e outras incluídas através *framework* O-MaSE para que o cenário de chão de fábrica fosse melhor representado. A Figura 4.3 ilustra a extensão do modelo anteriormente apresentado com as devidas alterações.

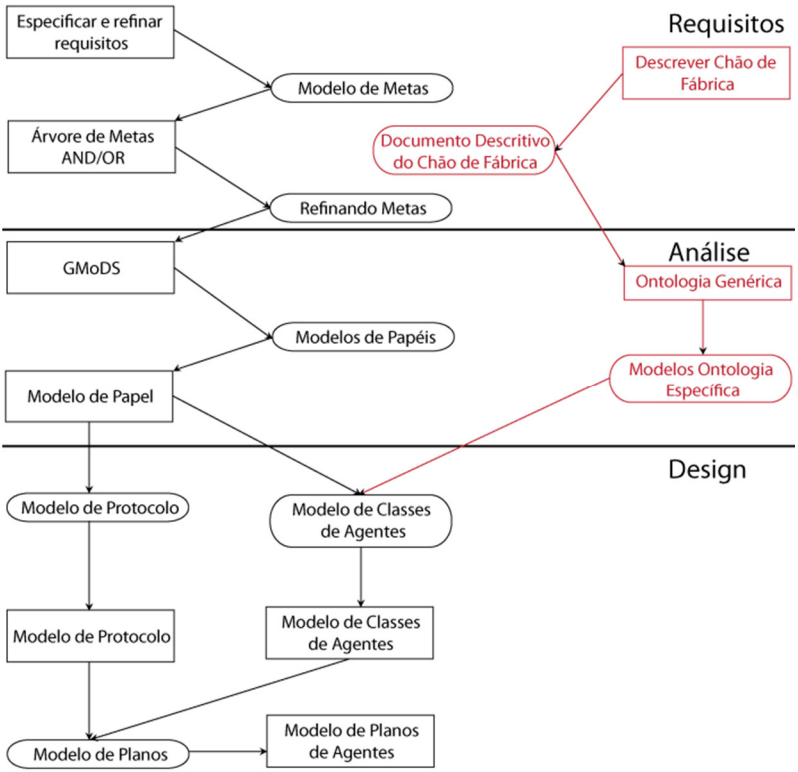

Figura 4.3. Estrutura da Metodologia proposta.

A Figura 4.3 representa a estrutura da metodologia proposta neste trabalho. Ela está relacionada ao exemplo da Figura 4.2, e foram incluídas duas novas etapas e uma etapa foi abstraída. As etapas na Figura 4.3 são detalhadas no decorrer deste capítulo.

4.2. O PROCESSO DE DERIVAÇÃO.

A derivação do sistema passa por etapas, onde se descreve cada método com o intuito de representar da melhor forma possível um determinado cenário industrial. Nesta proposta, um processo de derivação de SMAs industriais é executado em cinco etapas:

1. Descrever chão de fábrica;
2. Criar ontologia;
3. Especificar e refinar requisitos;
4. Decompor em metas, o que é dividido em outras duas subetapas;
   4.1. Modelar Metas;
   4.2. Refinar Metas;
5. Criar estrutura responsável por unir as especificações das etapas 1,2,3 e 4. Esta etapa é subdividida em quatro subetapas:
   5.1. Definir Papéis;
   5.2. Definir Agentes;
   5.3. Definir Protocolos;
   5.4. Definir Planos.

As etapas 1 e 2 e as etapas 3 e 4 podem ser executadas paralelamente, pois são responsáveis pela especificação dos requisitos do sistema e estrutura de representação do conhecimento a cerca de todo chão de fábrica. Ainda, são responsáveis pela análise, cobrindo a composição do chão de fábrica, relacionamento entre os equipamentos e hierarquia de produção do cenário que está sendo modelado.

A metodologia IDEF0 (*Integration Definition for Function Modeling*) (*apud* LEAL et al, 2008) foi utilizada para dar suporte à formalização dos passos adotados no processo de derivação. Ela faz parte do conjunto de padrões da ISO/IEC/IEEE 31320-1:2012, normalmente aplicada em modelagens de sistema e permite descrever, através de diagramas, o modelo funcional do sistema que se pretende analisar ou implementar. Ela é baseada na estrutura SADT (*Structured Analisys and Design Technique*) desenvolvida por (DICKOVER et al., 1977).

Basicamente, cada atividade é expressa através de uma caixa (diagrama), ao redor da qual há os chamados ICOMs (*Input Control Output Mechanism*). Setas indicam a finalidade ou procedência de uma determinada informação. A lateral esquerda representa as entradas (inputs), a lateral direita as saídas (outputs), a parte superior as restrições e habilitadores, e a parte inferior os recursos necessários e atores envolvidos para executar a tarefa.

O processo de mais alto nível é rotulado como A0 (Activity 0), que é posteriormente desdobrado em subníveis.

As caixas retangulares na Figura 4.4 representam cada tarefa ou subtarefa, representando a proposta da metodologia.

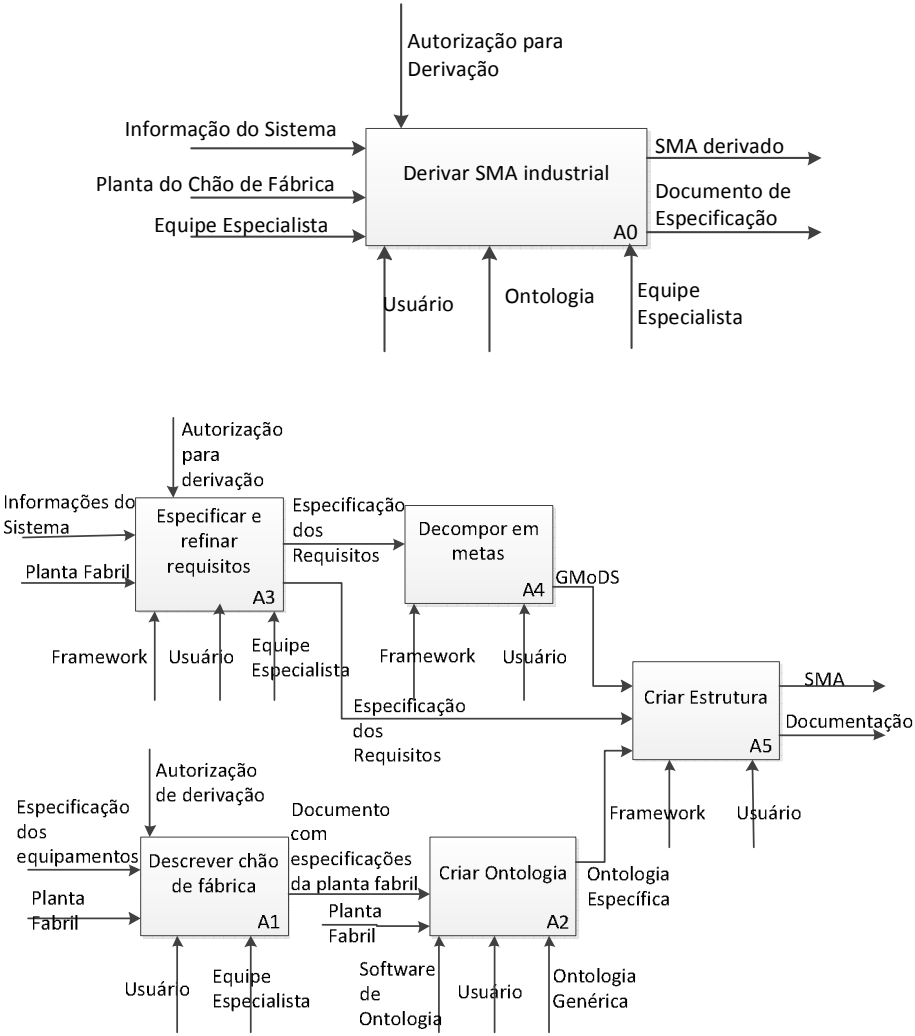

Figura 4.4. Diagrama do processo de derivação

A seguir, as etapas da metodologia serão detalhadas.

4.3. DETALHAMENTO DAS ETAPAS

Antes de iniciar o detalhamento das etapas, alguns conceitos utilizados no processo de modelagem devem ser estabelecidos. São eles:

- Usuário – É o utilizador responsável por definir as características do sistema e também o que deverá fazer a modelagem e análise das entradas e recursos de cada etapa.
- Planta fabril – Se refere ao conhecimento e definição sobre os componentes da planta fabril em questão (máquinas, robôs, transportadores, etc.), incluindo o relacionamento existente entre eles.
- Sistema de derivação – Sistema que fornece suporte ao utilizador derivador no momento da derivação de um cenário particular. Esse sistema deve seguir as etapas modeladas pelo modelo conceitual, guiando o utilizador derivador na fase de implementação. Neste trabalho, o sistema de derivação se refere ao *framework* O-MaSE, adaptado para o tipo de cenário que se propõe trabalhar.
- Ontologia genérica – Ontologia padrão, baseada na ADACOR.
- Equipe de especialistas – Profissionais que detêm o conhecimento acerca dos processos do chão de fábrica e dos equipamentos que compõem o cenário.
- GMoDS – *Goals Model Description System* introduz o relacionamento de precedência e "gatilhos" existentes entre as metas ("goals"). Para representar o GMoDS é utilizada uma árvore AND/OR. A árvore AND/OR é um diagrama UML que apresenta as metas do sistema. Quando uma folha de uma árvore precisa adquirir outra meta para atingir um objetivo, o relacionamento existente entre a meta *pai* e a meta *filho* é AND, ou seja, o *pai* precisa daquela meta para atingir determinado objetivo. Quando o relacionamento é alternativo, ou seja, o *pai* pode precisar da meta *filho* o relacionamento é do tipo OR.
- AUML – *Agent Unified Modeling Language*: diagrama que representa os atos de comunicação entre os agentes e a sequência que eles ocorrem.

4.1.1. Etapa A1 - Descrever chão de fábrica

Esta etapa tem como objetivo descrever o fluxo de trabalho e os equipamentos existentes no cenário de chão de fábrica. A Figura 4.5 é um segmento da Figura 4.4 e representa a etapa A1.

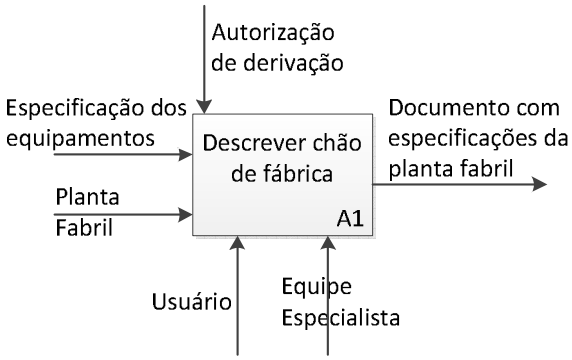

Figura 4.5. Etapa A1 – Descrever chão de fábrica

- Entrada: layout do chão de fábrica e a especificação dos equipamentos levando em consideração os relacionamentos existentes entre os mesmos.
- Restrições: Essa etapa depende da autorização de derivação do chão de fábrica para iniciar o processo de derivação.
- Recursos: O utilizador derivador, juntamente com uma equipe de especialistas.
- Saída: Um documento descritivo do chão de fábrica é gerado especificando todas as características que o cenário possui (máquinas, ferramentas, transportadores, etc), seus relacionamentos e a hierarquia existente entre os equipamentos, até mesmo entre os ciclos de trabalho.

O profissional que executa essa etapa deve possuir um bom conhecimento a respeito do Domínio da aplicação, identificando todos os equipamentos e suas respectivas funcionalidades e relacionamentos.

4.1.2. Etapa A2 - Definir ontologia

Esta etapa tem como objetivo transformar a ontologia genérica ADACOR em uma ontologia específica que represente os equipamentos do chão de fábrica em questão, suas características e seus relacionamentos. A Figura 4.6 é um segmento da Figura 4.4 e representa a etapa A2.

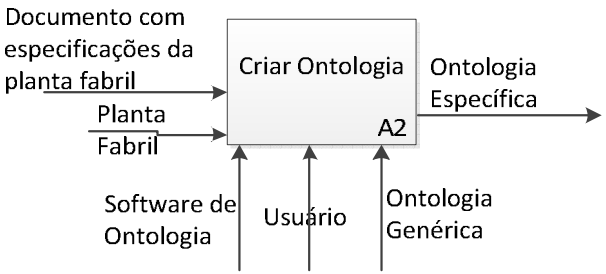

Figura 4.6. Etapa A2 – Definir Ontologia

Entrada: Planta fabril e o documento de descrição do chão de fábrica, gerado na etapa A1.

Recursos: Ontologia genérica de chão de fábrica, usuário, ferramenta de modelagem de ontologias, por exemplo, o Protégé, que é a que foi usada neste trabalho.

Saída: Ontologia específica que represente o cenário de chão de fábrica, para o qual se deseja derivar o SMA.

Com a especificação de todos os componentes realizada na etapa A1, a ontologia genérica é modelada de acordo com as características do cenário. Dessa forma, uma máquina ou uma esteira, por exemplo, se tornam instâncias de “tool” e “transporter” respectivamente na ontologia genérica. Caso algum componente do chão de fábrica não se encaixe na ontologia genérica, uma nova classe deve ser criada para representar esta instância.

O profissional que executa essa etapa deve possuir um bom conhecimento a respeito do Domínio, compreender e analisar o documento que especifica os componentes, atributos e relacionamentos, para, então, alterar, adaptar e modelar a ontologia genérica, tornando-a uma ontologia específica para adaptar ao cenário que está sendo modelado. Como a representação do conhecimento é feita utilizando a ferramenta Protégé, é necessário que esse profissional domine essa ferramenta de modelagem.

A ferramenta Protégé foi utilizada para análise e adaptação da ontologia genérica. A Figura 4.7. ilustra uma modelagem da ontologia desenvolvida no Protégé.

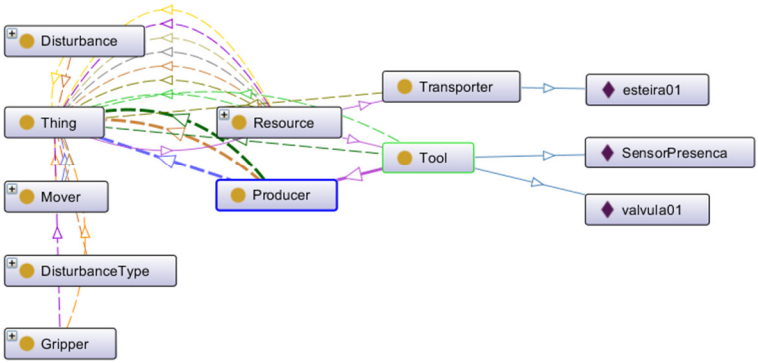

Figura 4.7. Ontologia genérica com instâncias de 3 (três) equipamentos.

4.1.3. Etapa A3 - Especificar e refinar requisitos

Esta etapa tem como objetivo analisar e especificar os requisitos que o sistema deve atender. Quanto mais rico e claro for o detalhamento dos requisitos, mais coerente tenderá a ser o resultado final da derivação. A Figura 4.8 é um segmento da Figura 4.4 e representa a etapa A3 da metodologia.

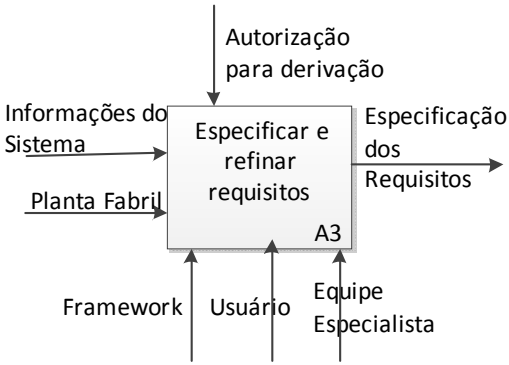

Figura 4.8. Etapa A3 - Especificar e refinar requisitos

A etapa A3 corresponde tem as seguintes propriedades:

Entrada: Informações referentes ao sistema e a planta fabril.

Restrições: Depende da autorização de derivação do chão de fábrica para iniciar o processo de derivação.

Recursos: Utilizador derivador, apoiado pelo sistema de derivação juntamente com uma equipe de especialistas deve analisar as informações do cenário e definir os objetivos que o sistema deve atingir.

Saída: Um documento descritivo do sistema é gerado, especificando todos os requisitos que o sistema deve possuir, assim como os objetivos que o mesmo deve atingir.

Esse documento deve conter todos os requisitos funcionais e não funcionais do cenário.

O profissional que executa essa etapa deve possuir um bom conhecimento a respeito do Domínio, identificando todos os requisitos que o sistema deve atender.

4.1.4. Etapa A4 - Decompor em metas

Esta etapa tem como objetivo traduzir o documento gerado na etapa A3 em uma árvore AND/OR, traçando todas as metas que o sistema deve atingir. São consideradas metas as funções globais da organização, ou seja, os requisitos do sistema. A Figura 4.9 é um segmento da Figura 4.4 e representa a etapa A4 da metodologia.

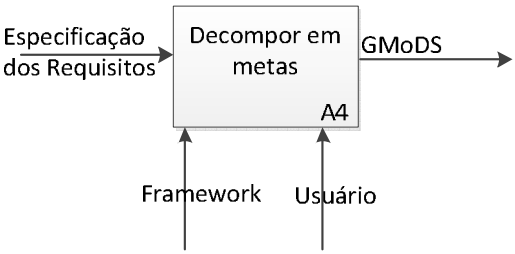

Figura 4.9. Etapa A4 – Decompor em metas

A etapa A4 corresponde tem seguintes propriedades:

- Entrada: Documento descritivo do sistema gerado na etapa A3.
- Recursos: Usuário, apoiado pelo Sistema de Derivação.

- Saída: É gerado um diagrama definido como árvore AND/OR, que representa as metas que o sistema deve atender com todas as restrições e dependências entre metas.

O profissional que executa essa etapa deve possuir um bom conhecimento a respeito do Sistema de Derivação, das técnicas de decomposição AND/OR e identificar os atributos, gatilhos e sequência de metas, respeitando o documento de especificação do sistema, produzido na etapa A3.

Esta etapa é dividida em outras duas subetapas, como representado na Figura 4.9:

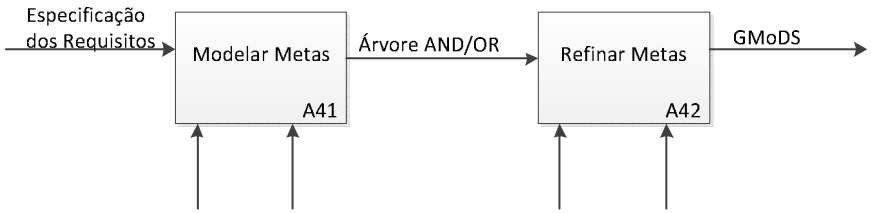

Figura 4.9. Sub-etapas A4

4.1.4.1. Sub-Etapa A41 - Modelar Metas

Esta subetapa tem como objetivo criar um diagrama a partir das especificações do documento gerado na etapa A3.

- Entrada: Documento descritivo do sistema gerado na etapa A3.
- Recursos: Usuário, apoiado pelo sistema de derivação.
- Saída: É gerado um diagrama AND/OR.

4.1.4.2. Sub-Etapa A42 - Refinar Metas

Esta subetapa tem como objetivo refinar a árvore AND/OR, adicionando as relações entre metas.

- Entrada: Documento descritivo do sistema gerado na etapa A3 e a árvore AND/OR da subetapa A41.
- Recursos: Usuário, apoiado pelo sistema de derivação.
- Saída: Árvore AND/OR com todas as relações de precedência ou dependências entre metas.

A estrutura da árvore consiste em organizar as metas em folhas, em que as folhas-filho são conectadas às folhas-pai através de uma relação de objetivos que o pai precisa (AND) adquirir e uma relação que o pai pode (OR) adquirir.

Esta etapa completa a árvore AND/OR, incluindo os tipos de relacionamento existentes entre metas dispostas horizontalmente na estrutura. Esse relacionamento define quando uma meta precede outra ou quando uma meta dispara outra meta através de um evento. A Figura 4.10 a seguir ilustra o diagrama da árvore.

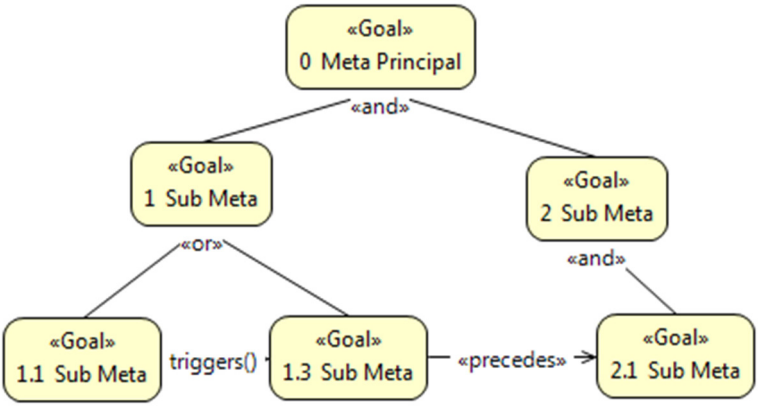

Figura 4.11. Diagrama AND/OR com relacionamentos de precedência e dependência.

4.1.5. Etapa A5 - Criar Estrutura

Esta etapa tem como objetivo unir as especificações do sistema e do chão de fábrica das etapas anteriores a fim de gerar um SMA inicial que represente o cenário observado.

As etapas A2 e A4 dependem das etapas A1 e A3 respectivamente, assim como a etapa A5 depende das etapas A2 e A4, podendo assim definir que a etapa A5 realiza a unificação entre especificações do sistema e representação do conhecimento.

A Figura 4.12 é um segmento da Figura 4.4 e representa a etapa A5 da metodologia.

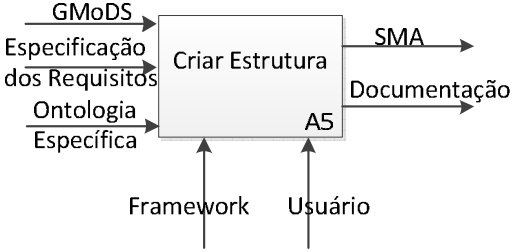

Figura 4.12. Etapa A5 – Definir Estrutura

A etapa A5 tem as seguintes propriedades:

- Entrada: Ontologia do chão de fábrica, documento de especificação dos requisitos do sistema e a árvore AND/OR.
- Recursos: Utilizador derivador e sistema de derivação.
- Saída: Documentação de especificação de todo sistema e os códigos dos agentes, "esqueletos dos agentes" que constitui o SMA derivado inicial.

Nesse ponto todas as especificações do sistema já foram feitas: restrições, requisitos, descrição dos componentes do chão de fábrica e representação acerca dos componentes do chão de fábrica por ontologia estão prontos. Para atingir o objetivo final dessa etapa faz-se necessário vincular todas as especificações produzidas pelas etapas anteriores e utilizá-las no desenvolvimento do SMA. A Figura 4.13 ilustra como esse processo deve ser interpretado.

Pode-se ler a Figura 4.12 da seguinte forma: a partir das instâncias criadas na análise do chão de fábrica, auxiliado por uma ontologia genérica de manufatura, as etapas A1 e A2 e as especificações dos requisitos do sistema através de uma árvore de metas, e as etapas A3 e A4, criam-se os agentes que, por sua vez, agregam papéis que deverão atender às metas do sistema. Para representar a comunicação e a troca de informações entre os agentes cria-se o diagrama de sequência. Por fim, estrutura-se um plano que os agentes devem seguir para que o processo seja completo.

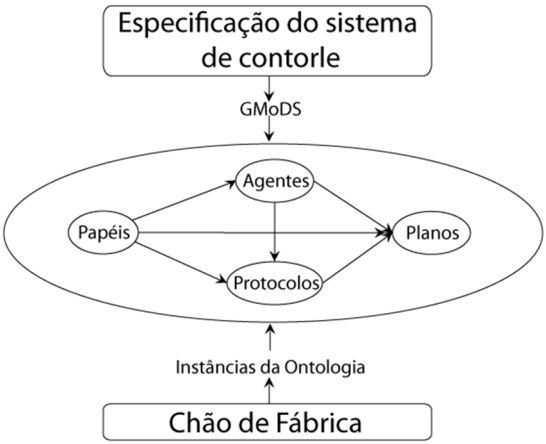

Figura 4.13. Processo de derivação de papéis, agentes, protocolos e planos.

Por ser uma etapa complexa, esta etapa é derivada em 4 subetapas, como visto na Figura 4.14 seguinte.

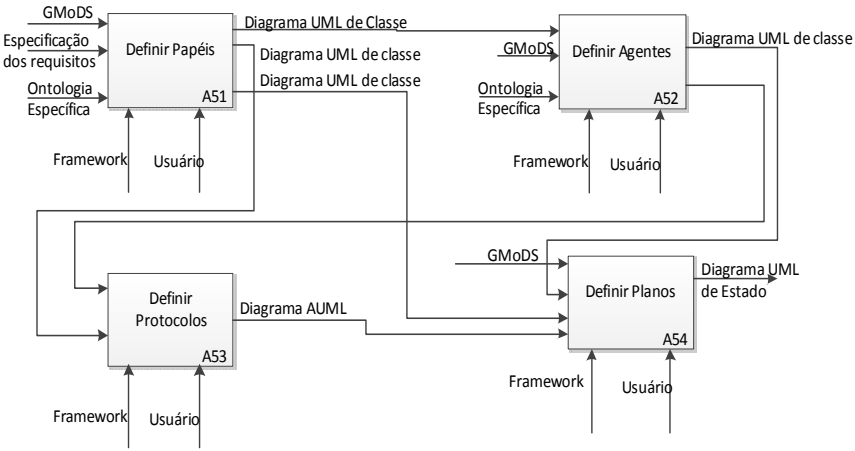

Figura 4.14. Diagrama IDEF0 da etapa A5 - Criar Estrutura.

4.1.5.1. Subetapa A51 - Definir papéis

Geralmente cada meta disposta na árvore de metas AND/OR adquire um papel no diagrama UML desta etapa; porém, um papel pode representar múltiplas metas ou ainda muitos papéis são necessários para atingir uma meta.

A Figura 4.15 é um segmento da Figura 4.12 e representa a etapa A51.

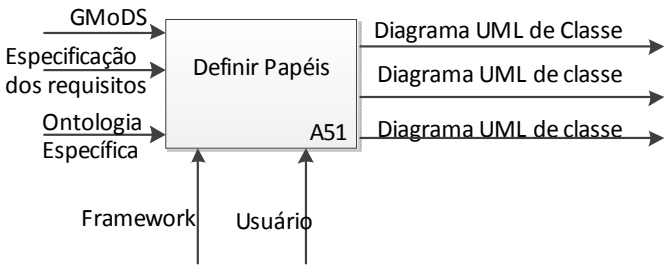

Figura 4.15. Etapa A51 – Definir Papéis

A subetapa A51 tem as seguintes propriedades:

- Entrada: Diagrama AND/OR, ontologia específica e documento de análise de requisitos.
- Recursos: Usuário e sistema de derivação.
- Saída: Diagrama UML relacionando os vínculos entre papéis ("role") e entre papéis e metas ("goal"). A cada instância da ontologia atribui-se um papel. Este papel se relacionará com outros papéis modelados na análise de requisitos das etapas A3 e A4.

O profissional que executa essa etapa precisa interpretar e integrar todas as etapas anteriores de especificação e representação do conhecimento acerca do chão de fábrica para poder definir quais papéis devem ser criados, a fim de satisfazer os objetivos identificados.

A Figura 4.16 ilustra a modelagem UML dos papéis.

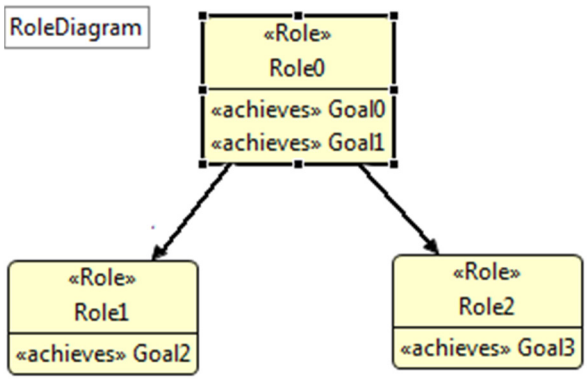

Figura 4.16. Diagrama UML representando os papéis.

4.1.5.2. Subetapa A52 - Definir agentes

Esta subetapa tem como objetivo criar os agentes adequados para desempenhar/executar os papéis e representar os componentes do chão de fábrica, definidos como instâncias na etapa de A2.

A Figura 4.17 é um segmento da Figura 4.12 e representa a etapa A52.

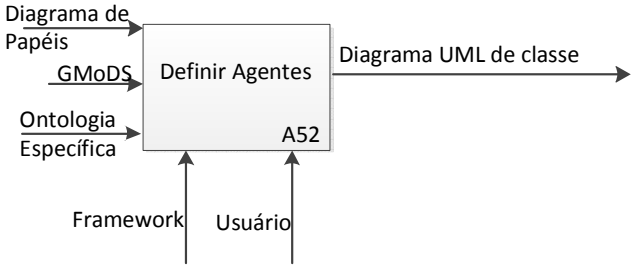

Figura 4.17. Etapa A52 – Definir Agentes

A etapa A52 tem as seguintes propriedades:

- Entrada: Diagrama UML de papéis, subetapa A51, ontologia específica, etapa A2 e documento de análise de requisitos, etapa A3.
- Recursos: Usuário e sistema de derivação.
- Saída: Estrutura organizacional dos agentes representada por um diagrama UML que inclui os relacionamentos entre agentes e os vínculos dos agentes com os papéis.

O profissional que executa essa etapa precisa interpretar e integrar todas as etapas anteriores de especificação e representação do conhecimento acerca do chão de fábrica para poder definir quais agentes devem ser criados e quais papéis esses agentes irão executar dentro da organização.

Toda instância criada na ontologia deve ser interpretada como um agente nessa etapa. A este agente será atribuído os papéis que melhor se adaptem às funções desempenhadas dentro da organização.

A classe de um agente é um *template* de um tipo de agente no sistema, analogamente aos diagramas UML de classe. As setas representam as interações entre agentes. A Figura 4.18 ilustra a modelagem UML dos agentes.

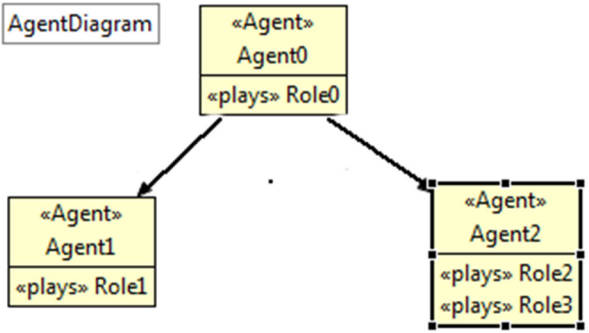

Figura 4.18. Diagrama UML representando os agentes.

4.1.5.3. Subetapa A53 - Definir protocolos

Nessa etapa somente as interações entre agentes são identificadas. Os tipos de informação que os agentes irão transmitir são definidos. A Figura 4.19 é um segmento da Figura 4.12 e representa a etapa A53.

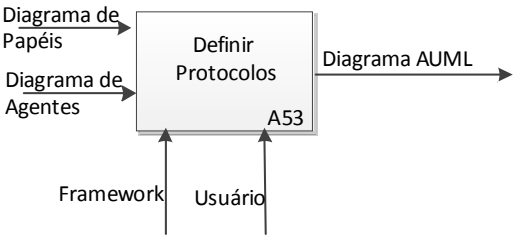

Figura 4.19. Etapa A53 – Definir Protocolos

A etapa A53 tem as seguintes propriedades:

- Entrada: Diagrama de papéis e agentes.
- Recursos: Utilizador derivador e Sistema de Derivação.
- Saída: Diagrama AUML que representa os protocolos de comunicação estabelecida entre os agentes.

O profissional que executa essa etapa precisa interpretar os diagramas UML de papéis e agentes, sendo responsável por gerar os diferentes protocolos que devem existir entre os agentes.

Um diagrama AUML é usado para representar os protocolos de comunicação, que estabelecem a ordem das mensagens enviadas entre os agentes. A Figura 4.20 ilustra o diagrama de protocolos AUML.

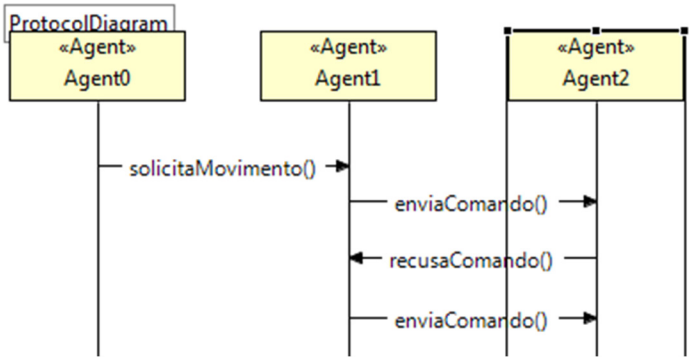

Figura 4.20. Diagrama AUML representando os protocolos de comunicação.

4.1.5.4. Subetapa A54 - Definir planos

O planejamento captura as ações e protocolos usados pelo agente para atingir os objetivos. A Figura 4.21 é um segmento da Figura 4.12 e representa a etapa A54.

A etapa A54 tem as seguintes propriedades:

- Entrada: Diagrama de papéis, agentes, protocolos e o GMoDS.
- Recursos: Usuário e Sistema de Derivação.
- Saída: Diagrama UML de estados, que define a hierarquia com que os processos de produção do chão de fábrica são sequenciados.

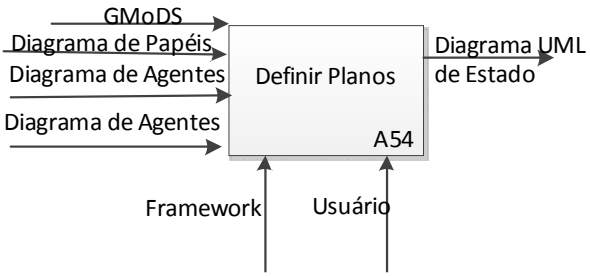

Figura 4.21. Etapa A54 – Definir Planos

O profissional que executa essa etapa é responsável por desenhar a sequência de planos/estados que os agentes devem seguir/atingir a fim de satisfazer os objetivos do sistema.

Esse processo é representado por um diagrama UML de transição de estados. As setas representam mensagens transmitidas entre estados. A Figura 4.22 ilustra o diagrama UML de estado.

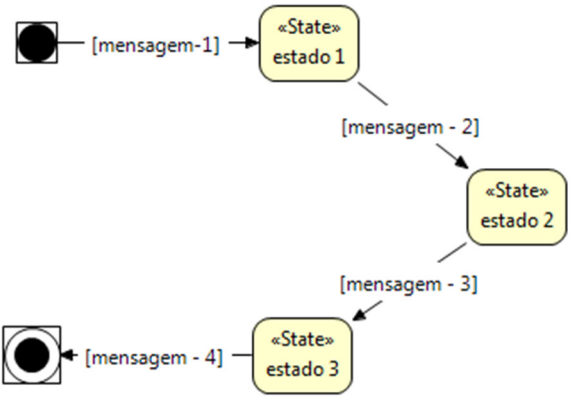

Figura 4.22. Diagrama UML representando os estados.

Os profissionais responsáveis pelas etapas Definir papéis, Definir agentes, Definir Protocolos e Definir Planos precisam ter conhecimento acerca de UML para modelagem.

5. Protótipo desenvolvido

O presente capítulo tem como objetivo descrever o trabalho de implementação idealizado nesta dissertação.

A implementação concentra-se em adaptar os meta-modelos do *framework* O-MaSE às etapas da metodologia proposta neste trabalho. A descrição do *framework* foi abordada no item 4.1.

5.1. TECNOLOGIAS UTILIZADAS

Utilizaram-se as seguintes ferramentas para desenvolver esse projeto:

- Eclipse Ganymede Platform, Versão 3.4.0
- agentTool 1.1, Eclipse Plugin
- agentTool Process Editor (APE) O-MaseML Method Library
- Protégé, Versão 4.1.0
- Jade

Além da versão Ganymede 3.4.0 do Eclipse, as versões Galileo 3.5 e Europa 3.3 podem ser utilizadas. Porém, nesta última, é necessário instalar a ferramenta gráfica Graphical Editing Framework (GEF), possibilitando a utilização dos recursos gráficos de modelagem disponíveis pelo plugin agentTool.

Para implementar as etapas da metodologia proposta neste trabalho, utilizou-se a APE O-MaSE, mencionada em 4.1, que fornece as estruturas/meta-modelos básicos do framework O-MaSE e que serviram como um guia contendo todas as etapas da metodologia O-MaSE. A APE (*agentTool Process Editor*) é uma biblioteca que permite implementar os meta-modelos do framework O-MaSE utilizando uma interface gráfica que se aproxima da modelagem UML.

5.2. CRIANDO ETAPAS NO FRAMEWORK O-MASE

Para criar as etapas da metodologia é necessário instalar os plugins agentTool e criar um projeto baseado na APE O-MaSE mencionada anteriormente.

Cada etapa da metodologia O-MaSE é um fragmento/meta-modelo. A proposta utiliza os seguintes fragmentos: Especificação dos Requisitos, Modelo de Metas, Modelo de Papéis, Modelo de Agentes, Modelo de Protocolos e Modelo de Planos. Acrescentando ainda as etapas de Descrição de Chão de Fábrica e Modelo de Ontologia, o que caracteriza esta proposta como sendo específica para cenários de chão de fábrica.

A Figura 5.1 ilustra as etapas da metodologia implementadas no *framework* O-MaSE. As etapas são desenvolvidas utilizando documentos de especificação, ontologia, diagramas UML e diagramas AUML.

Como resultado final da implementação, após realizar toda modelagem do cenário, é gerado o código dos agentes, ou seja, o "esqueleto" dos agentes, que posteriormente devem ser complementados, programando as ações, métodos e comportamentos específicos de cada agente.

Na Figura 5.1 também pode ser observado que as etapas foram divididas em níveis de abstração, que podem ser classificadas na engenharia de software como Especificação, Análise e Projeto. Cada etapa da metodologia é identificada com um *Index*. A sequência das etapas é definida no campo *Predecessors*, mostrando as dependências de cada etapa.

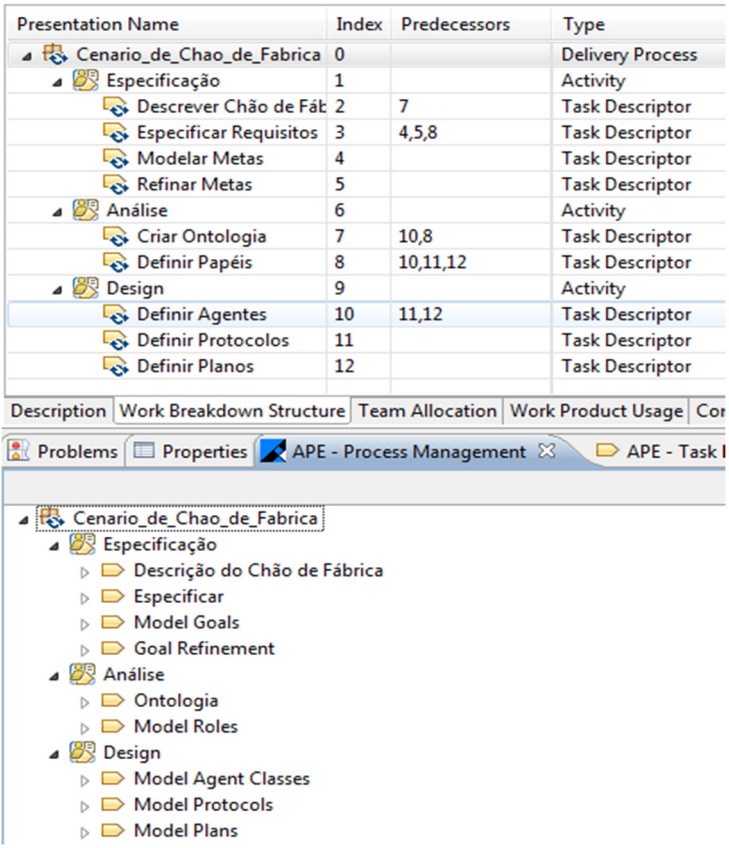

Figura 5.1. Etapas implementadas no Framework O-MaSE

O *framework* define as Especificações, Análise e Projeto como *Activity*. As etapas da metodologia proposta são definidas como *Task Descriptor*. As etapas da metodologia que possuem uma representação nativa no *framework*, como por exemplo, Definir Agentes, possuem um modelo UML que pode ser modelado automaticamente acessando a *Task* com um duplo click, como observado na Figura 5.2.

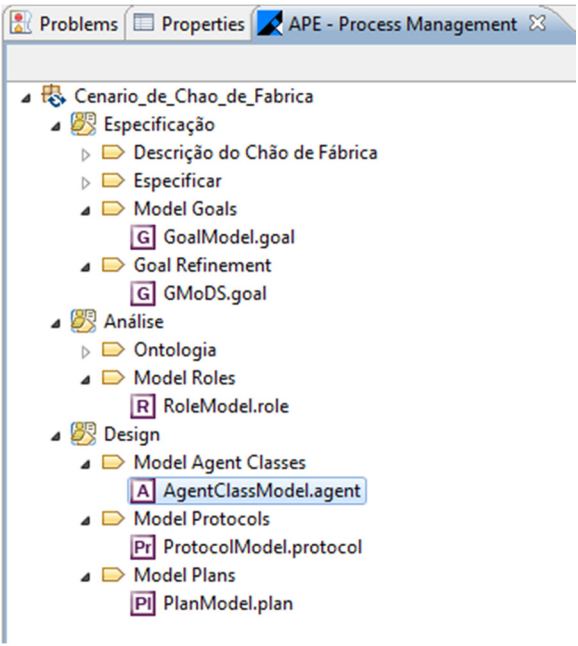

Figura 5.2. Acessando o modelo UML de modelagem de Agentes.

Ao acessar o modelo de definição de agentes é necessário modelar os agentes que comporão a estrutura de agentes de acordo com as especificações modeladas nas etapas anteriores. Um exemplo da modelagem de agentes pode ser observado na Figura 4.17.

5.3. ETAPAS NÃO ESPECIFICADAS NO O-MASE

Estas etapas foram desenvolvidas especificamente com intuído de abranger um cenário particular: o cenário de chão de fábrica. Estas etapas buscam aperfeiçoar a análise e modelagem, visto que o cenário é considerado

de natureza complexa, o que torna a presente metodologia específica e particular.

As etapas que não possuem representação nativa no O-MaSE são:

- Descrição de Chão de Fábrica: Corresponde a elaboração de um documento similar ao documento de Especificação dos Requisitos, com o diferencial de ser direcionado exclusivamente para fornecer informações mais abrangentes em relação ao cenário que será utilizado.
- Criar Ontologia: O documento de Descrição do Chão de Fábrica fornece informações que possibilita construir uma Ontologia que representa o cenário que está se utilizado. Com a Ontologia ADACOR servindo como base, é possível estabelecer conceitos e termos padronizados já utilizados por outros projetos, abrindo oportunidades de integração com outros sistemas que utilizem essa ontologia de manufatura.

5.4. CENÁRIO DE CHÃO DE FÁBRICA

Diversos cenários foram analisados no decorrer dos estudos dessa dissertação. Um cenário simples foi escolhido para ilustrar a utilização da metodologia.

5.4.1. Descrição do cenário

A descrição do cenário pode ser, nesta metodologia, representada pelas etapas "Descrever Chão de Fábrica" e "Especificar Requisitos".

A descrição dos requisitos consta de um documento que reúne as informações do chão de fábrica através do conhecimento dos profissionais envolvidos no chão de fábrica, planta fabril e especificação dos equipamentos (*datasheet*). O resultado desta análise pode ser observado na tabela 5.1.

Descrição do chão de fábrica:
- Esteira: Transportar os vasilhames sobre a esteira até o ponto de abertura da válvula. Especificação: • Altura, largura, comprimento, velocidade, sensor de presença. - Válvula: Acionar válvula após posicionamento de vasilhame ser concluído. Especificação: • Tempo de abertura, volume mínimo, volume máximo, volume desejado.

Tabela 5.1. Descrição de Chão de Fábrica (Exemplo parcial)

A especificação dos requisitos deve seguir um padrão adotado pela engenharia de software. A análise sobre determinado cenário é divida em funções que o sistema deve atender ou realizar. Estes aspectos são fragmentados em requisitos funcionais e não-funcionais. Para produzir o documento de análise de requisitos é preciso possuir o conhecimento do processo de produção, objetivos da fábrica e composição do cenário.

O documento que representa uma análise dos requisitos pode ser observado na tabela 5.2, em que cada função é descrita e restrições são atribuídas a elas.

O sistema deve gerenciar o preenchimento de líquido em uma esteira. Sempre há vasilhames disponíveis na esteira.				
Nome do Requisito: F1 – Preencher vasilhame com líquido **Descrição:** Sistema deve mover esteira, posicionando os vasilhames sob a válvula de liberação de líquido, posteriormente deve acionar a válvula de liberação do líquido. **Requisitos não Funcionais:**				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF1.1 - Abre válvula	Válvula de liberação de líquido somente abre se tiver vasilhame sob ela.	Segurança		

Tabela 5.2. Análise de requisitos (Exemplo parcial)

5.4.2. Modelagem do Cenário

A modelagem do cenário é constituída em transferir todos os conhecimentos adquiridos na Descrição do cenário em modelos UML, AUML e Ontologia seguindo a sequência definida na metodologia.

O primeiro passo da metodologia, Descrição do Chão de Fábrica, descrito em 5.4.1, é pré-requisito para a segunda etapa, Criar Ontologia. Como mencionando em 5.3, a ontologia ADACOR servirá como base para gerar uma metodologia específica pra o cenário que está sendo analisado. A Figura 5.3 ilustra um diagrama da ontologia ADACOR adaptado ao cenário em questão.

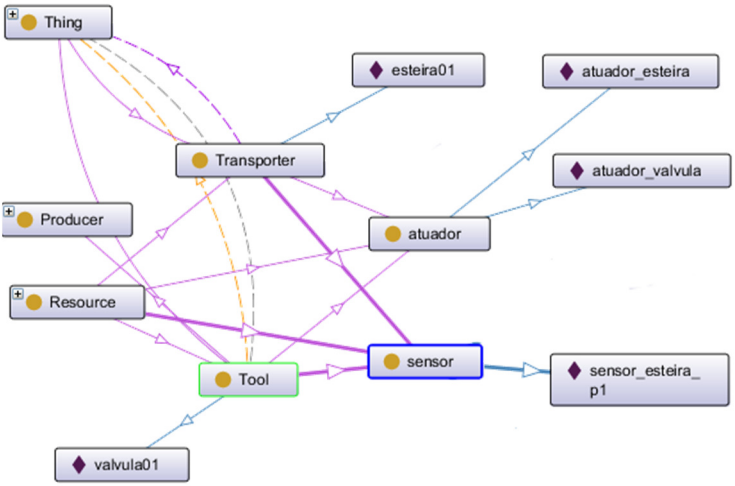

Figura 5.3. Ontologia ADACOR adaptada

Na sequência da modelagem, os requisitos de sistema são analisados e classificados. Está etapa está descrita no item 5.4.1. Seguindo os passos da metodologia, é preciso definir as metas e refiná-las. Como esta etapa é dividida em duas sub-etapas, duas Figuras foram utilizadas pra descrevê-la. A Figura 5.4 representando a sub-etapa Definir Metas exige habilidade do desenvolvedor em converter os requisitos funcionais em metas. A Figura 5.5 ilustra o refinamento das metas de acordo com o relacionamento e dependências que uma meta tem com outra.

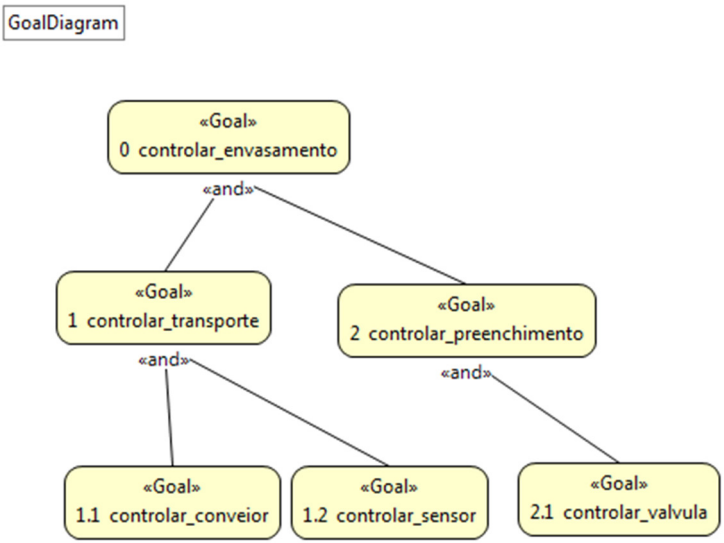

Figura 5.4. Definindo as Metas

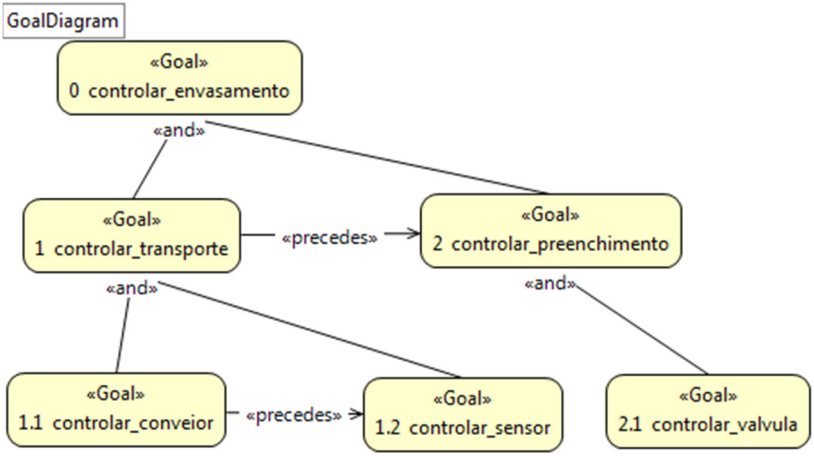

Figura 5.5. Refinando as Metas

A próxima etapa, Definir Estrutura, reúne quatro sub-etapas que dependem das especificações e modelagens realizadas até o momento. A primeira sub-etapa é responsável por Definir os Papéis que cada agente irá executar, representada pela Figura 5.6.

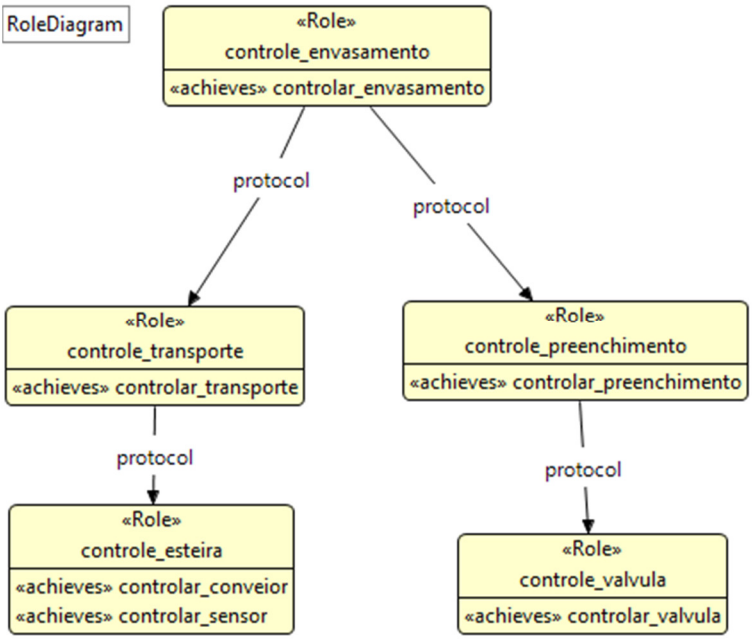

Figura 5.6. Definindo os Papéis

A segunda sub-etapa, Definir Agentes, define os agentes que serão gerados a partir da análise dos papéis e define um agente para cada instância gerada na ontologia (Figura 5.7).

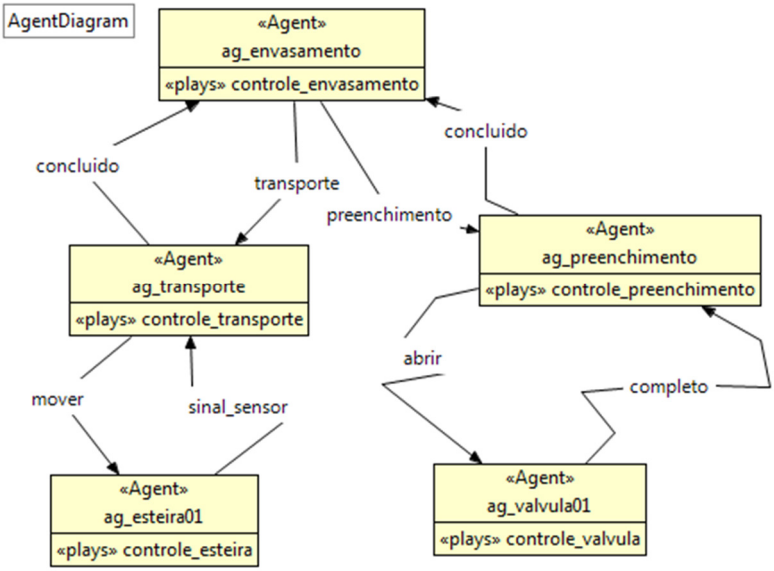

Figura 5.7. Definindo Agentes

A Definição de Protocolos, terceira sub-etapa, é modelada observando os agentes e as metas que devem ser atingidas, definindo assim quais agentes precisam trocar mensagens entre si, como pode ser observado na Figura 5.8.

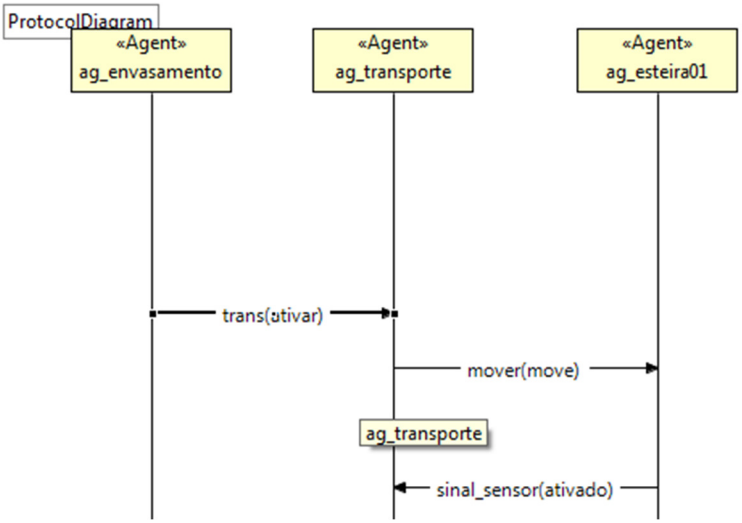

Figura 5.8. Definindo Protocolos (Exemplo parcial)

Por último são definidos os Planos, representados por um diagrama de interação, Figura 5.9, estabelece os estados que o sistema deve atingir para que o ciclo de funcionamento/processo seja bem sucedido.

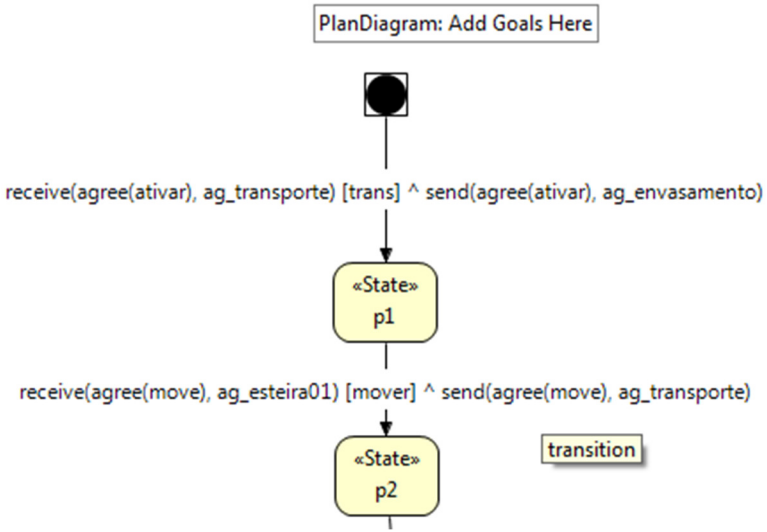

Figura 5.9. Definindo Planos (Exemplo Parcial)

Concluída a etapa de Definir planos é possível gerar os códigos-fonte na linguagem Java dos agentes que foram modelados. Além dos agentes, toda uma estrutura é gerada para que, quando o sistema for inicializado, o gerenciador (JADE) seja inicializado para dar inicio ao processo de geração das instâncias dos agentes no sistema. O Jade é responsável por gerenciar a organização, disponibilização das páginas de registros de serviços e de registro de agentes.

5.5. GERAÇÃO DOS ESQUELETOS DOS AGENTES

Com a conclusão da modelagem do cenário, é possível gerar o código-base dos agentes. Para isso é preciso selecionar o projeto e acessar o menu do eclipse: agentTool3->Code Generation->JADE-based Code Generation. Será criado um novo projeto, com a estrutura dos arquivos na linguagem java como pode ser observado na Figura 5.10.

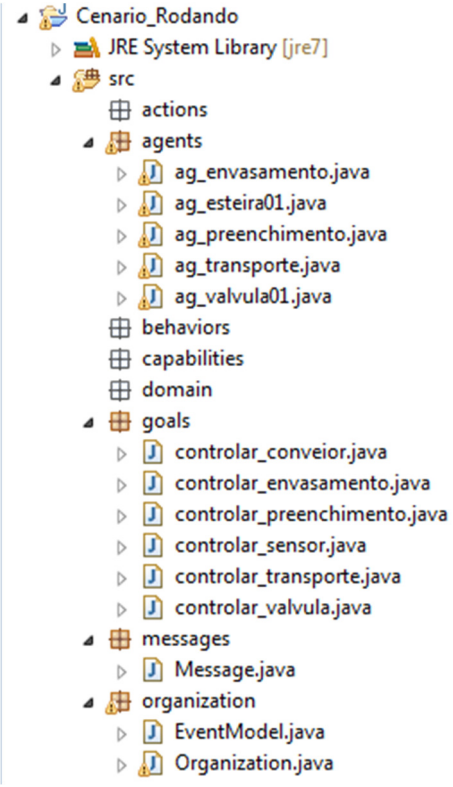

Figura 5.10. Estrutura dos arquivos Java.

Esta estrutura possui as características modeladas no framework O-MaSE utilizando apenas documentos de análise de requisitos, ontologia e diagramas UML.

5.5.1. Análise dos códigos gerados

Os códigos gerados representam o esqueleto-base de um programa escrito na linguagem Java que representa um agente, com suas estruturas de código características de agentes, comportamentos e protocolos de comunicação, como pode ser observado na Figura 5.11.

```
package agents;

import messages.Message;
import behaviors.*;
import jade.core.Agent;
import jade.core.behaviours.Behaviour;
import jade.lang.acl.*;
import jade.core.AID;

public class ag_preenchimento extends Agent {
  //NON-Reason
  private static final long serialVersionUID = 1;
  protected void setup() {
    System.out.println("Agent "+getLocalName()+" started.");
    //ADD GENERAL BEHAVIOUR THEN ADD SPECIFIC BEHAVIOURS
    addBehaviour(new MyGlobalBehaviour(this));
  }

private class MyGlobalBehaviour extends Behaviour{
    //NON-Reason
      private Agent a;
      private static final long serialVersionUID = 1;

      private Message envioMensagem;

      MyGlobalBehaviour(Agent a){
        this.a=a;
        this.envioMensagem = new Message();
        }
      public void action(){
          ACLMessage msgEnvio;
          ACLMessage msg = myAgent.receive();
```

Figura 5.11. Estrutura agente, esqueleto padrão

Na Figura 5.11 é possível observar que o agente importa bibliotecas Jade. Este mesmo código está presente em todos os agentes gerados, o que possibilita que o Jade possa gerenciar e ter acesso aos agentes.

5.5.2. Adaptando o código gerado

Como a estrutura gerada possui suporte de comunicação, é possível adaptar os agentes introduzindo códigos de envio de mensagem entre os agentes, permitindo enviar conteúdos informativos, requisições ou comandos de execução. Um exemplo de envio e recebimento de mensagens pode ser observado na Figura 5.12.

```
public void action(){
  ACLMessage msg = myAgent.receive();
  if(this.startWebservice == 1)
  {
    this.startWebservice += 1;
    this.consultaWebservice();
  }
  if(msg!=null){
  String content = msg.getContent();
  System.out.println(content);
  if(msg.getPerformative() == ACLMessage.INFORM)
  {
    System.out.println("ag_envasamento recebe: "+msg.getContent());

    if(msg.getSender().getLocalName().compareTo("ag_transporte") == 0)
    {
      if(content.compareTo("ok")==0)
      {
        msgEnvio = envioMensagem.construtorMsg("ag_preenchimento",
                   "AutorizadoPreenchimento", ACLMessage.REQUEST);
      }
      else
      {
        msgEnvio = envioMensagem.construtorMsg("ag_transporte",
                   "tenteNovamente", ACLMessage.REQUEST);
      }
        send(msgEnvio);
      }
```

Figura 5.12. Código esqueleto alterado.

Como existe a flexibilidade de poder alterar o código-fonte e as estruturas de execução já estarem preparadas, com a presença de bibliotecas, referências dentre instâncias e protocolos padrões de comunicação, construiu-se um cenário simples para exemplificar a troca de mensagem da estrutura modelada anteriormente.

A estrutura de agentes é composta pelos agentes: ag_envasamento, ag_transporte, ag_esteira01, ag_preenchimento, ag_valvula01.

O código de todos estes agentes foi alterado para que fosse possível simular uma situação real de funcionamento do processo de envasamento. O código da Figura 5.12 é referente a programação do ag_envasamento, em que é possível observar a validação de recebimento de uma mensagem no formato FIPA-ACL. Caso exista uma mensagem, e essa mensagem seja um *Inform,* do protocolo FIPA-ACL, do ag_transporte, um *Request* é enviado ao ag_preenchimento requisitando o preenchimento do vasilhame.

A sequência de troca de mensagens e solicitação de ações segue os seguintes passos:

1. Autorização de produção do setor administrativo da indústria ao ag_envasamento;
2. Envio de requisição de movimentação da esteira ao ag_transporte;
3. Envio de requisição de movimentação de esteira ao ag_esteira01;
4. Esteira movida, ag_esteira01;
5. Sensor detectou presença de vasilhame sob a válvula, ag_esteira01;
6. Confirmação de posicionamento de vasilhame ao ag_transporte;
7. Confirmação de posicionamento de vasilhame ao ag_envasamento;
8. Envio de requisição de preenchimento ao ag_preenchimento;
9. Envio de requisição de abertura da válvula ao ag_valvula01;
10. Preenchimento Concluído, ag_valvula01;
11. Confirmação de preenchimento ao ag_preenchimento;
12. Confirmação de preenchimento ao ag_envasamento;
13. Operação concluída, ag_envasamento.

O resultado da execução do SMA modificado para atender a sequência proposta é apresentado na Figura 5.13. Todos os passos programados, que representa a sequência completa de envasamento de um vasilhame, podem ser visualizados na Figura 5.13.

```
INFO: ----------------------------------------
Agent container Main-Container@Vanderlei-PC is ready.
--------------------------------------------
Agent ag_preenchimento started.
Agent ag_envasamento started.
Inicilização -- Tem ordem de serviço.
Organization organization started.
Agent ag_esteira01 started.
Agent ag_valvula01 started.
Agent ag_transporte started.
Recebi a seguinte mensagem:PermissaoParaMoverConveior
Recebi a msgMoveConveior
Movendo Esteira
Passou no Sensor
ag_transporte recebe: ok
ag_envasamento recebe: ok
Recebi a msgAutorizadoPreenchimento
Recebi a msgAbrirValvula
Preenchimento concluido.
ag_preenchimento recebe: ok
ag_envasamento recebe: ok
Operacao concluida....
```

Figura 5.13. Resultado obtidos após implementação do cenário.

A interface JADE observada da Figura 5.14 apresenta os agentes que foram instanciados após a geração do código pelo *framework* O-MaSE e posterior programação dos comportamentos utilizados para validar a metodologia.

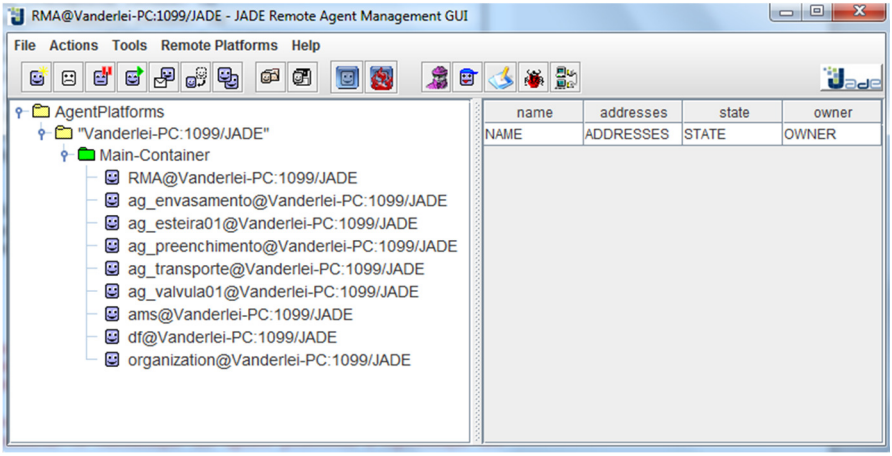

Figura 5.14. Interface JADE

A biblioteca Jade também é gerada pelo framework O-MaSE o que facilita a instanciação e gerenciamento dos agentes.

A modelagem apresentada neste capítulo demonstra como a metodologia proposta é usada na derivação de um SMA hipotético, em ambiente controlado. A modelagem do cenário, apesar de simples, serve para demostrar as características de funcionamento de um cenário industrial, derivado utilizando a metodologia proposta.

6. DISCUSSÃO E AVALIAÇÃO DOS RESULTADOS

O ambiente de chão de fábrica possui características que as metodologias tradicionais de derivação de sistemas ainda não conseguem contemplar de uma forma satisfatória. Isso se deve ao fato de que esse cenário possui propriedades específicas e particulares que não se enquadram nos modelos estudados.

Neste trabalho foi proposta uma metodologia usando o conceito de ontologia com uma etapa específica para descreve adequadamente o cenário. Com isto, as características podem ser representadas utilizando-se padrões específicos de modelagem para construção de um modelo apropriado para o cenário e de fácil entendimento para especialistas.

Como o principal objetivo do *framework* O-MaSE é apoiar a criação de novas metodologias, ele se mostrou adequado para o proposito deste trabalho, contendo modelos robustos e de fácil adaptação às necessidades do desenvolvedor.

Definir quais modelos devem compor a nova metodologia não é uma tarefa fácil, pois algumas vezes somente se observa que está faltando um método ou que algum método não se encaixa no momento de modelar um cenário, o que torna o processo de definição dos métodos lento.

Os próprios métodos disponíveis na O-MaSE para modelagem não possuem uma definição clara de seu significado e importância, sendo assim é preciso realizar diversos estudos para estabelecer se aquele método é realmente essencial e satisfaz a necessidade a minha metodologia.

Após análise dos meta-modelos disponíveis no *framework*, foram selecionados aqueles que melhor representam o processo de modelagem e criadas duas novas etapas com a finalidade de tornar esta metodologia específica. Estas tiveram o intuito de modelar esse tipo de ambiente com maior clareza e representatividade no processo de derivação.

A etapa “Descrever chão de fábrica” possibilita expor textualmente os detalhes do processo de produção de determinado cenário de chão de fábrica, o que possibilita maior clareza do processo de produção no momento da análise da composição e relacionamento entre equipamentos.

Na etapa “Criar Ontologia”, a ontologia ADACOR é utilizada como base, pois utiliza uma taxionomia padrão de chão de fábrica, possibilitando maior integração com outros sistemas e compreensão pelos profissionais da área. Como a Ontologia ADACOR é alvo de muitas pesquisas e implementações, utilizado em diversos projetos, optou-se por utilizá-la como base nesta etapa.

Para representar determinados equipamentos que não possuem uma representação padrão na ontologia foi adicionando uma representação que

contempla as características do mesmo no cenário que está sendo modelado. Na modelagem, a representação dos equipamentos presentes no chão de fábrica é feita através de instâncias das classes da ontologia base ADACOR.

As duas etapas mencionadas anteriormente contribuíram na criação de uma metodologia específica e particular para derivação de sistemas industriais. Além de proporcionar um grau elevado de descrição do cenário de chão de fábrica, isso torna a leitura da modelagem mais intuitiva e completa, caso seja necessário futuras intervenções.

Ao final do processo de modelagem é obtido um arcabouço de código Java, com suporte à troca de mensagens no padrão FIPA-ACL, estrutura organizacional, estrutura de definição de papéis e estrutura de comportamentos. Os códigos gerados oferecem estruturas que posteriormente serão alteradas/adaptadas para programar as propriedades específicas de cada agente de acordo com a função dos mesmos dentro do ambiente.

A possibilidade de incorporar novos componentes a um sistema cujo código já tenha sido gerado e o sistema estando em operação é possível através do registro do agente na organização. O Jade é responsável por gerenciar esse registro. Como a inclusão de novo agente na organização impacta em remodelagem do mesmo, etapa por etapa, buscando identificar o impacto causado nos relacionamentos entre os agentes que estão rodando, o Jade torna essa inclusão transparente através do uso de registro de agentes e suas características e papéis dentro na organização.

Como o código gerado é uma "casca" com suporte a implementações diversificadas a fim de poder representar todas as funcionalidades de um equipamento, os novos agentes devem ter a capacidade e estrutura mínima de comunicação, quando não com suas funcionalidades já definidas.

A metodologia proposta neste trabalho, além de descrever o cenário de chão de fábrica com maior riqueza de detalhes, deixou as etapas mais enxutas e simples de modelar, com recursos gráficos e possibilidade de geração dos códigos base.

Para avaliar e verificar o resultado/código/estrutura gerado pelo *framework* O-MaSE, uma simulação para o cenário de uma fábrica de envasamento de líquidos foi desenvolvida da seguinte forma:

- Modelagem do cenário de acordo com as etapas propostas pela metodologia;
- Geração dos arcabouços dos agentes;
- Implementação das funcionalidades de cada equipamento nos agentes, programando os agentes de controle, trocas de mensagens entre os agentes e demais ajustes;
- Execução e simulação do sistema.

Os esqueletos gerados foram adaptados com êxito, traduzindo os requisitos funcionais, requisitos não-funcionais e especificação do ambiente industrial. A customização dos agentes possibilitou a representação dos comportamentos, ações e comunicação entre os agentes utilizando o protocolo de comunicação gerado pelo *framework* O-MaSE.

O comportamento do sistema durante a execução foi dentro do esperado, realizando todas as funções programadas na adaptação dos códigos. As mensagens trocadas entre os agentes usaram o protocolo FIPA-ACL corretamente. Os agentes executaram seus papéis de acordo com as funcionalidades estabelecidas na programação e na ordem estabelecida pelo fluxo do processo de produção.

7. CONCLUSÕES

Uma metodologia de derivação de sistemas multiagentes (SMA) industriais foi desenvolvida utilizando o meta-modelo do *framework* O-MaSE.

Apesar da complexidade existente na derivação de um SMA, seja ele industrial ou não, o *framework* O-MaSE possibilita modelar um cenário complexo de forma visual, utilizando diagramas UML de fácil compreensão.

Todas as metodologias para desenvolvimento de sistema multiagentes pesquisadas possuem uma etapa em que os agentes são claramente definidos. No entanto, não são plenamente voltados para um chão de fábrica.

A primeira etapa da metodologia proposta, a qual contém a descrição do chão de fábrica, não está presente nas metodologias estudadas. A ontologia, segunda etapa da metodologia proposta, é utilizada por outras metodologias no processo de modelagem de sistemas de chão de fábrica. Porém, não é parte integrante da mesma, ou seja, não representa uma etapa bem definida, como na metodologia proposta. A ontologia específica, baseada na ontologia ADACOR e gerada neste trabalho, inclui padrões de representação de equipamentos de cenários industriais, predefinidos na ontologia-base.

Ao final da modelagem, foi gerada a estrutura organizacional dos agentes, na qual cada agente está preparado para ser programado de acordo com as funcionalidades e especificações de sua representação no chão de fábrica, assim como agentes abstratos de controle da sequência de produção dos produtos gerados pelo chão de fábrica.

O framework O-MaSE, apesar de não ser muito simples de manipular e compreender, se mostrou adequado para o objetivo proposto, de "montar" uma metodologia e depois gerar SMAs.

7.1. Limitações

A definição dos agentes modelados na ontologia não é feita de forma automatizada, sendo preciso analisar a ontologia e criar de forma manual cada agente.

A representação dos protocolos de comunicação industriais foi estudada, porém a complexidade inerente a eles impossibilitou sua implementação neste trabalho.

7.2. Trabalhos futuros

Como sugestão para trabalhos futuros se propõe:

- Integrar os agentes gerados com os equipamentos físicos utilizando técnicas de arquitetura orientada a serviço, especificamente o protocolo *Device Profile Web Service* (DPWS), com o objetivo de suportar uma comunicação "mais direta" com os controladores dos equipamentos industriais e de minimizar os problemas de interoperação com os encapsulamentos usualmente feitos.
- Possibilitar que o agente possa buscar as funcionalidades na forma de serviços, disponibilizados em repositórios distribuídos, maximizando escalabilidade e reuso funcional.
- Implementar um Sistema de Derivação para automatizar o máximo possível assim como assistir o usuário-derivador ao longo do processo de criação de um SMA industrial, amparado pela metodologia desenvolvida.
- Introduzir o conceito de Organização do projeto do SMA, tornando mais bem organizada a arquitetura, topologia e papéis dos agentes dentro de um dado contexto.
- Com base na ontologia, criar "agentes-classe" por tipo de equipamentos industriais de forma a facilitar o processo de derivação.

REFERÊNCIAS BIBLIOGRÁFICAS

BRESCIANI, PAOLO. Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems, v. 8, n. 3, p. 203-236, 2004.

BORGO, S. AND LEITÃO, P., "The Role of Foundational Ontologies in Manufacturing Domain Applications", On the Move to Meaningful Internet Systems: CoopIS, DOA and ODBASE, R. Meersman and Z. Tari (eds.), Lecture Notes in Computer Science, Springer-Verlag Berlin-Heidelberg, vol. 3290, pp. 670-688, 2004.

CAMARINHA-MATOS, LM AND BARATA, J. AND FLORES, L. "Shopfloor Integration and Multiagent based Supervision", Livro: Intelligent Engineering Systems, ,IEEE , pg:457-462, 1997.

CÂNDIDO, G. AND BARATA, J. "A multiagent control system for shop floor assembly", jornal: Holonic and Multi-Agent Systems for Manufacturing, Springer, pg.: 293-302, 2007.

CASAIS, F. AND LEITAO, P. AND RESTIVO, F. Holonic Manufacturing Control: A Practical Implementation, Journal - IFIP Advances in Information and Communication Technology (AICT), p. 33-44, 2011.

DELOACH SA, Wood MF, Base W-patterson AIRF. June 2000 MULTIAGENT SYSTEMS ENGINEERING : THE ANALYSIS PHASE. 2000.

DELOACH SA. Analysis and Design using MaSE and agentTool. Cognitive Science. 2001.

DELOACH, SCOTT A. The mase methodology. In: Methodologies and software engineering for agent systems. Springer US, 2004. p. 107-125.

DICKOVER, Melvin E.; MCGOWAN, Clement L.; ROSS, Douglas T. Software design using: SADT. In: Proceedings of the 1977 annual conference. ACM, 1977. p. 125-133.

EVANS, RICHARD. MESSAGE: Methodology for engineering systems of software agents. EURESCOM, EDIN, p. 0223-0907, 2001.

FALASCONI, S. LANZOLA, G., STEFANELLI, M. Using ontologies in multi-agent systems. In Proceedings of Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW), University of Calgary, Banff, Alberta, Canada, 1996.

GARCIA-OJEDA, Juan C. et al. O-MaSE: a customizable approach to developing multiagent development processes. Springer Berlin Heidelberg, 2008.

GIRET, A. AND BOTTI, V. "Engineering holonic manufacturing systems", journal=Computers in industry, vol. 60, p. 428-440, Elsevier, 2009.

GRUBER, T.R. "Toward principles for the design of ontologies used for knowledge sharing", journal: International journal of human computer studies, vol.: 43, pg.: 907-928, 1995.

HÜBNER, JOMI FRED. Um Modelo de Reorganização de Sistemas Multiagentes. São Paulo, 2003, p.224

JENNINGS, N. - *Cooperation in Industrial Multi-Agent Systems*, World Scientific Series in Computer Science (Vol 43), 1994.

JENNINGS, N.R. AND CORERA, J.M. AND LARESGOITI, I. "Developing industrial multi-agent systems", Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), p. 423-430.1995.

JENNINGS, N.R. AND WOOLDRIDGE, M.J. "Applications of intelligent agents", Springer, 1998.

JENNINGS, N.R., WOOLDRIDGE, M.J. "Applications of intelligent agents", Agent Technology: Foundations, Applications, and Markets, Springer, 1998, pp. 3–28.

KARNOUSKOS, S. "Cyber-physical systems in the SmartGrid", Livro: Industrial Informatics (INDIN), IEEE, pg: 20-23, 2011.

KOMMA, V.; JAIN, P.; MEHTA, N. Agent-based simulation of a shop floor controller using hybrid communication protocols. International Journal of Simulation Modelling, v. 6, n. 4, p. 206-217, 2007.

LEAL, F. AND ALMEIDA, D.A. AND MONTEVECHI, J.A.B. “Uma Proposta de Técnica de Modelagem Conceitual para a Simulação através de Elementos do IDEF”, Simpósio Brasileiro de Pesquisa Operacional, vol. 40, 2008.

LEE, E.A. “Cyber physical systems: Design challenges”, Livro: Object Oriented Real-Time Distributed Computing (ISORC), IEEE, pg: 363-369, 2008.

LEITÃO, P. AND RESTIVO, F.J. Implementation of a holonic control system in a flexible manufacturing system - IEEE, vol 38, p. 699-709, 2008.

LEITÃO, P. AND RESTIVO, F., “Identification of ADACOR Holons for manufacturing control”, Universidade de Porto, Portugal, 2009.

LOSS, LEANDRO. Integração de sistemas multiagente industriais. 2012.

LÜDER, A. The PABADIS’PROMISE-Architecture. Proceedings of Automazione e Strumentazione, v. 94, p. 101, 2007.

MONOSTORI, L. AND VÁNCZA, J. AND KUMARA, S.R.T. “Agent-based systems for manufacturing”, Journal: CIRP Annals-Manufacturing Technology, Elsevier, vol: 55, pg: 697- 720, 2006.

OLIVEIRA, JOSÉ ANTÔNIO BARATA DE. Coalition Based Approach for Shop Floor Agility – A multiagent Approach. Livro, Edições Orion, Ano 2005.

PAVÓN, JUAN; GÓMEZ-SANZ, JORGE J.; FUENTES, RUBÉN. The INGENIAS methodology and tools. Agent-oriented methodologies, v. 9, p. 236-276, 2005.

PEREIRA, ARNALDO; RODRIGUES, NELSON; LEITÃO, PAULO. Deployment of multi-agent systems for industrial applications. In: Emerging Technologies & Factory Automation (ETFA), 2012 IEEE 17th Conference on. IEEE, 2012. p. 1-8.

PESCHKE, JÖRN; LUDER, A.; KUHNLE, H. The PABADIS'PROMISE architecture-a new approach for flexible manufacturing systems. In:Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on. IEEE, 2005. p. 6 pp.-496.

PRESSMAN, ROGER S. Engenharia de software. McGraw Hill Brasil, Livro, 1995.

RABELO, RICARDO J.; CAMARINHA-MATOS, L. M. Negotiation in multi-agent based dynamic scheduling. Robotics and Computer-Integrated Manufacturing, v. 11, n. 4, p. 303-309, 1994.

RABELO, RICARDO JOSÉ. "Um enquadramento para o Desenvolvimento de Sistemas de Escalonamento Ágil da Produção – Uma abordagem Multiagente". Lisboa, 1997.

ROBIN G. QIU, "Towards ontology-driven knowledge synthesis for heterogeneous information systems", Journal: Intelligent Manufacturing, vol.: 17, pg.: 99-109, 2006.

SILVEIRA, R.A. "Introdução a Sistemas Multiagente", Universidade Federal de Santa Catarina (UFSC), 2006.

STUDER, Rudi; BENJAMINS, V. Richard; FENSEL, Dieter. Knowledge engineering: principles and methods. Data & knowledge engineering, v. 25, n. 1, p. 161-197, 1998.

THANGARAJAH, J. AND PADGHAM, L. AND WINIKOFF, M. "Prometheus design tool", Livro: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, p. 127—128, ACM, 2005.

W. SHEN, Q. HAO, H. JOONG, AND D.H. NORRIE, "Applications of agent-based systems in intelligent manufacturing : An updated review q," *Advanced Engineering Informatics*, vol. 20, 2006, pp. 415-431.

WERNECK, V. M. B. Uma Avaliação da Metodologia MAS-CommonKADS. In: Proceedings of the Second Workshop on Software Engineering for Agent-oriented Systems. 2006. p. 13-24.

WINIKOFF M, PADGHAM L. "Chapter 11 THE PROMETHEUS METHODOLOGY", Springer, 2004.

WOOLDRIDGE, M.; JENNINGS, N. - *Agent Theories, Architectures, and Languages: A Survey*, Lectures Notes in AI (890) / Intelligent Agents (eds. M. Wooldridge e N. Jennings), pp.1-39, Springer-Verlag, 1995.

WOOLDRIDGE, M. AND JENNINGS, N.R. AND KINNY, D. , "The Gaia methodology for agent-oriented analysis and design", Journal: Autonomous Agents and Multi-Agent Systems, Springer, vol. 3, p. 285--312,2000.

WYSK, RICHARD A.; SMITH, JEFFREY S. A formal functional characterization of shop floor control. Computers & industrial engineering, v. 28, n. 3, p. 631-643, 1995.