Ambiente de derivação de sistemas multiagentes industrias apoiados por metodologia e ontologia

Vanderlei L. C. Weber

PPGEAS - Programa de Pós-Graduação em Engenharia de Automação e Sistemas -- Universidade Federal de Santa Catarina (UFSC) -- Caixa Postal 476 -- 88040-900 -- Florianópolis -- SC – Brasil

vweber@das.ufsc.br

Resumo. *O estudo apresentado nesse artigo concentra-se em gerar um SMA (Sistema multiagente) que represente um cenário de chão de fábrica, apoiada por uma metodologia,* em que são analisados *os objetivos gerais da indústria em um nível abstrato para chegar a um esqueleto SMA que represente o cenário específico. Para auxiliar esta metodologia, fez-se uso de ontologia e um sistema computacional que segue as etapas modelados pela metodologia.*

1. Introdução

Os modernos sistemas de manufatura vêm requerendo cada vez mais de inteligência dos vários atores envolvidos no planejamento, execução e recuperação de ações, bem como uma grande capacidade de readaptação do seu layout na medida em que novos produtos ou mudanças de processos de inovação são concebidos pela empresa. A abordagem dos sistemas multiagente (SMA) vem sendo considerada uma das mais adequadas para o desenvolvimento de sistemas computacionais para suportar tal adaptação.

Embora haja vários trabalhos em termos de arquiteturas e metodologias para a construção de SMAs, pouco há quando os SMAs são voltados para aplicações industriais, em particular os que envolvem controle de chão-de-fábrica e equipamentos industriais. Uma das razões é a intrínseca complexidade de tal cenário, uma vez que cada equipamento (e seu controlador industrial) é diferente. Existem dezenas de tipos de equipamentos, cada chão-de-fábrica tem um layout e uma arquitetura de controle particular, entre outros aspectos. Paralelamente a isso, há diversas abordagens de "encapsulamento" (*wrapping*) que, numa filosofia de integração *bottom-up*, precisam adaptar-se aos requisitos de integração com SMAs em seus vários níveis de protocolos de comunicação. Dada essa enorme diversidade, é difícil conhecer qual seria a mais adequada arquitetura e conceitos associados para o dado SMA.

Nesse contexto, serão abordados nesse trabalho, como criar SMAs industriais de uma forma que, ao final, este seja coerente com a planta e completo em relação a todos os equipamentos envolvidos, robusto e interoperável em relação às comunicações envolvidas e flexível no sentido de permitir facilmente que um usuário possa introduzir mudanças gerais no sistema e na planta, sem que para isso, o SMA tenha que ser refeito completamente.

A figura 1.1 a seguir ilustra o objetivo dessa pesquisa, ou seja, o desenvolvimento de uma metodologia de construção de SMA para aplicação em chão de fábrica, com auxílio de uma ontologia e um ambiente integrado de criação de tais sistemas, que possa satisfazer as necessidades mencionadas anteriormente.

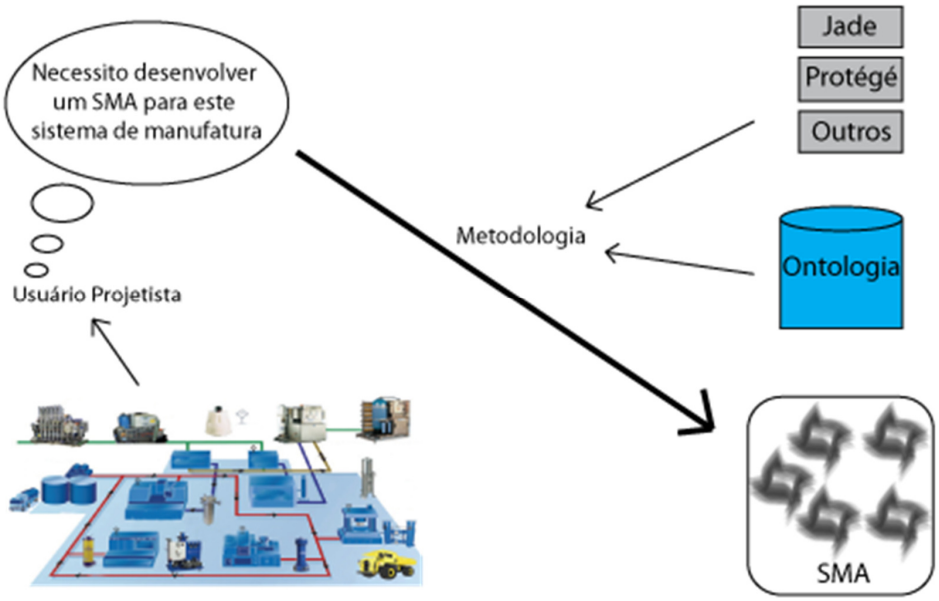

Figura 1.1. Derivação de um chão de fábrica

Na seção 2 é abordado a composição de um cenário industrial e a justificativa para uso de Sistemas multiagentes nesse ambiente; na seção 3, é apresentado uma metodologia e uma ontologia que servirão de base para esse trabalho e um *framework* que auxiliará no desenvolvimento da extensão da metodologia base; na seção 4, a metodologia O-MaSE é estendida, para melhor adequação ao problema de derivação; na seção 5, é demonstrada a utilização do *plugin agenttools* e da ape O-MaSE e na seção 6, as conclusões.

2. Sistemas Multiagentes Industriais

Quando se fala de indústria e planta fabril, normalmente imagina-se algo grande, composto por diversas máquinas, funcionários, transportadores, sistemas de controle, etc[Rabelo 1997]. Apesar da realidade nem sempre ser essa, em que se encontram poucas ou até mesmo uma máquina com algoritmos ótimos e processo de fluxo de produção bem definido, deve-se considerar cenários maiores, cenários que exijam uma característica colaborativa entre os diversos componentes do chão de fábrica, trabalhando em conjunto para atingir o objetivo final de produção. Nesse contexto, visando atender a demanda dos diferentes cenários fabris, independente de seu tamanho e complexidade, é necessário que se projete uma metodologia e um sistema que atenda um tamanho variado de indústrias.

Nesse ambiente característico de colaboração de chão de fábrica, 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). Essa é uma definição apoiada por [Jennings, Wooldridge. 1998], que diz que um agente é um sistema de computador situado em algum ambiente e que é capaz de ações autônomas 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. As características mais importantes e úteis num

ambiente de manufatura baseado em agentes são por exemplo: autonomia, cooperatividade, pró-atividade e adaptabilidade.

Algumas das principais características de um chão de fábrica são:

- natureza distribuída;
- sistemas heterogêneos;
- variedade de equipamentos, e;
- protocolos de comunicação particulares.

Relacionando as principais características de um chão de fábrica com SMAs, pode-se comparar a natureza distribuída com ambiente distribuído, sistemas heterogêneos com complexidade, protocolos de comunicação particulares e variedade de equipamentos com integração, justificando, assim, o uso dessa abordagem, em que os SMAs suprem as características principais encontradas em um ambiente industrial.

Segundo [Monostori L, Váncza J, Kumara 2006], o conceito de multiagente industrial está mais ligado à integração. A integração promovida pela adoção de multiagentes, possibilita continuidade operacional de uma determinada parte local da empresa, caso haja algum colapso de conectividade.

As fortes mudanças nos paradigmas das companhias de manufatura, consequentes da globalização da economia, requerem de uma empresa a máxima flexibilidade possível dos sistemas [Torre, Barata, Torre, Flores 1997], o que pode ser alcançado utilizando um sistema multiagente bem projetado. Como a flexibilidade exige aumento da agilidade, é necessário um sistema de supervisão com um nível de inteligência maior, que por sua vez, deve ser distribuído utilizando-se o paradigma de multiagentes.

Em [Rabelo, R.J. 1998] e [Jennings 1994], o comportamento de um agente industrial deve possuir as seguintes características:

- Semiautônomo – certo grau de autonomia, porém depende de outros agentes para concluir suas tarefas.
- Interação – interação com outros agentes visando solucionar um problema.
- Independente – conhecimento suficiente para concluir algumas tarefas.
- Cooperante – coopera com a organização, não possui comportamento destrutivo ou egoísta, buscando 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.

Um comparativo entre SMA Industrial e SMA Tradicional é apresentado na tabela 2.1. A tabela estabelece uma relação de propriedades de SMAs que normalmente é esperada em um sistema multiagente, segundo [W. Shen, Q. Hao, H. Joong, and D.H. Norrie 2006],[Teixeira, L.F.P. and Miranda, R. and de Ávila, E.S. 2001],[Jennings, N.R. and Corera, J.M. and Laresgoiti, I. 1995].

2.1. Tabela comparativa de SMAs

Propriedades	SMA Industrial	SMA Tradicional
Modularidade	Permite o funcionamento de setores locais da fábrica caso haja problemas de conectividade, garantindo a continuidade da produção.	Permite o funcionamento de subsistemas de uma empresa em caso de falha de um módulo.
Flexibilidade	A estrutura pode ser adaptada para executar uma nova tarefa, caso haja uma mudança na produção, ou ingresso de novos equipamentos.	Muito importante devido a alterações de sistemas e agregações de subsistemas ao SMA.
Robustez	Característica importante em caso de quebra de algum equipamento ou ingresso de novos equipamentos no ambiente fabril. Deve também suportar qualquer tipo de alteração do meio.	Característica importante em caso de implementação de nova tecnologia, o SMA deve estar preparado para qualquer alteração no ambiente.
Integração	Ambiente extremamente heterogêneo que necessita de integração entre sistemas e equipamentos.	Ambiente pode ser heterogêneo, necessita de integração entre os diversos sistemas existentes.
Tolerância a Falhas	Muito importante em SMAs industriais, evitando desperdício de matéria-prima e quebra de equipamentos. Evitar a parada da linha de produção	Sua importância ocorre pela consistência de dados.
Cooperação	Deve ser cooperativo para atingir os objetivos da organização.	Pode ser cooperativo, mas existe organização com a presença de agentes não cooperantes.
Domínio	Heterogêneo com relação a equipamentos, sistemas legados e protocolos de comunicação. Forte necessidade de comunicação e coordenação entre equipamentos, o que justifica sua complexidade de controle.	Presença de sistemas legados sistemas operacionais, bancos de dados e redes de comunicação variados.
Distribuição	Muito característico nesse tipo de domínio.	Pode ser distribuído.

3. Tecnologias de desenvolvimento

Para desenvolver um SMA que atenda as necessidades de chão de fábrica, utilizou-se a metodologia e o *framework* O-MaSE (*Organization-based Multiagent System Engineering)* que modela os fragmentos de métodos propostos pela metodologia através da ferramenta de desenvolvimento Eclipse Ganymede 3.4.2. É necessário também instalar a *api agenttools*, que possibilita construir diagramas UML dentro do Eclipse.

Com o objetivo de especificar e melhor representar o chão de fábrica, fez-se uso de uma ontologia, que permite definir a especificação e conceituação de um determinado domínio, em que basicamente, uma ontologia consiste dos conceitos, relações, suas definições, propriedades e restrições existentes no domínio [Gruber, T.R. 1995].

3.1. Metodologia O-MaSE

A metodologia O-MaSE desenvolvida por [Deloach SA. 2006] é derivado da metodologia MaSE (*Multiagent Systems Engineering*) também desenvolvida por [Deloach SA. 1999]. Alguns problemas em relação à MaSE foram corrigidos, mas o maior avanço concentra-se na possibilidade de customização da O-MaSE, proporcionando a *designers* criarem suas próprias metodologias orientadas a agentes. Ela é baseada em um conjunto de fragmento de métodos que definem um conjunto de produtos de análise e design que podem ser criados e usados dentro do *framework* O-MaSE.

3.2. Ontologia

A arquitetura ADACOR [Borgo, S. and Leit, P. 2004], desenvolvida para representar cenários industriais, possui uma ontologia baseada nas recomendações da FIPA *Ontoloy Service Recommendation*[C. Schlenoff, A. Knutilla, S. Ray 1996], e é utilizada como base genérica no desenvolvimento da ontologia nesse trabalho. A figura 3.1 a seguir representa essa ontologia, utilizando diagrama UML de classes.

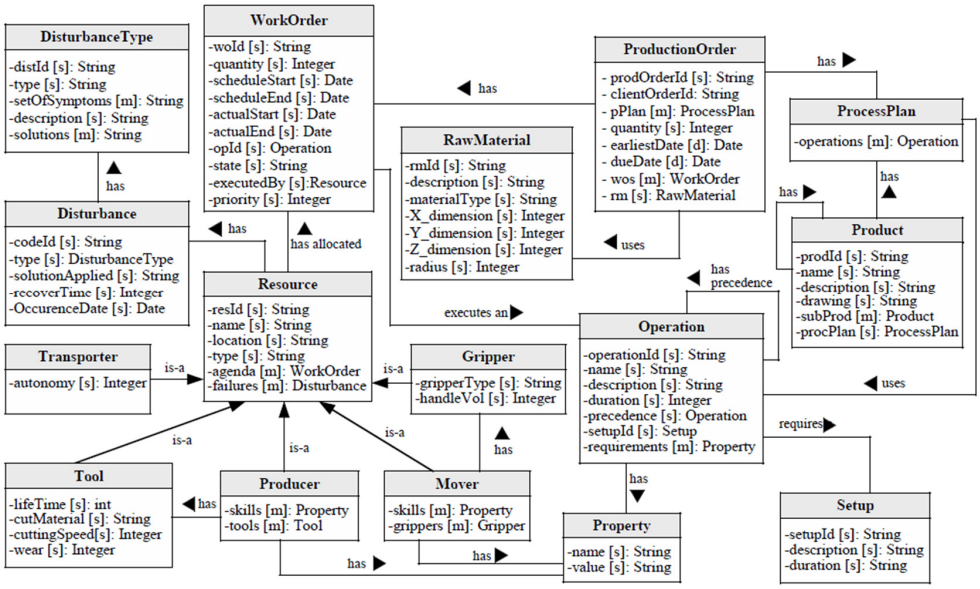

Figura 3.1. Ontologia de manufatura ADACOR

4. Extendendo a metodologia O-MaSE.

Tomando como base a metodologia O-MaSE e os fragmentos de métodos utilizados pelo *framework* O-MaSE, duas novas etapas foram criadas, a fim de melhorar a representação de conhecimento do chão de fábrica, com a finalidade de proporcionar maior integração com o sistema de controle. Essas etapas serão detalhadas em 4.2.

4.1. Construindo uma metodologia usando O-MaSE e Ontologia.

Para construir a metodologia proposta a seguir, foram utilizadas alguns fragmentos de métodos da metodologia O-MaSE e etapas que utilizam conceitos de ontologia. As etapas dessa metodologia são apresentadas da seguinte forma:

1 - Especificar e refinar requisitos;

2 - Decompor em metas, dividida em outras duas subetapas:

2.1 – Modelar Metas e;

2.2 – Refinar Metas;

3 - Descrever chão de fábrica e;

4 - Criar ontologia.

5 - Criar estrutura, é responsável por unir as especificações das etapas A1,A2,A3,A4 e é dividida da seguinte forma:

5.1 - Definir papéis;

5.2 - Definir Agentes;

5.3 - Definir Protocolos;

5.4 - Definir políticas e;

5.5 - Definir Planos.

Para representar a sequência de etapas propostas, é utilizada a metodologia IDEF0 (*Integration Definition for Function Modeling*), que faz parte do conjunto de padrões da família IDEF, normalmente aplicada em modelagens de sistema. Como esse método é uniforme e de iteração limitada para facilitar o processo, foi muito bem aceita para modelagem de processos industriais, gerenciais, executivos e outros. As caixas retangulares, ICOMs (*Input Control Output Mechanism*), observadas na figura 4.1, representam cada tarefa. Setas indicam a finalidade ou procedência de um determinado dado. A lateral esquerda representa as entradas (*inputs*), a lateral direita as saídas (*outputs*), parte superior as restrições e parte inferior os recursos necessário para executar a tarefa. A figura 4.1, ilustra a estrutura IDEF0, com a proposta da metodologia.

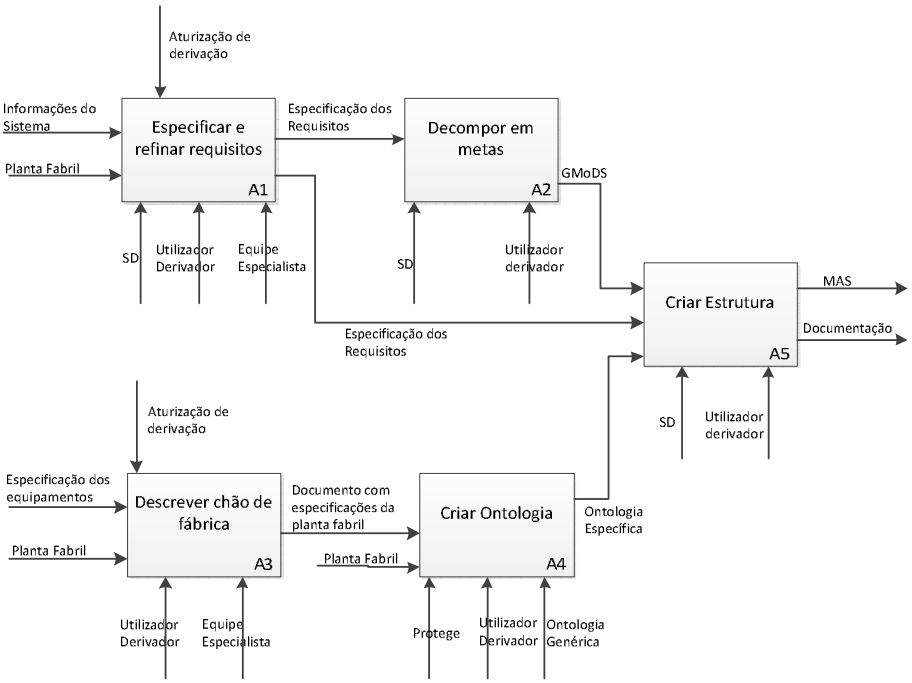

Figura 4.1. Diagrama da metodologia.

Como pode ser observado na figura 4.1, existem dois fluxos de desenvolvimento que podem ser realizados em paralelo. O fluxo iniciado em A1 e o fluxo iniciado em A3. A etapa A5 é responsável por unir os dois fluxos de desenvolvimento.

O cenário de chão de fábrica analisado para aplicar essa metodologia conta com um agv, uma esteira e uma válvula de preenchimento de líquidos. Na descrição das etapas da metodologia, serão detalhadas algumas etapas para melhor compreensão de suas aplicações dentro desse cenário industrial.

4.2. Etapas da metodologia

As etapas ilustradas na figura 4.1, A1, A2 e A5 são derivadas da metodologia O-MaSE e são usadas como fragmentos de métodos no *framework* O-MaSE. As etapas A3 e A4 foram criadas para melhor especificação do domínio, visando integrar os agentes que serão desenvolvidos na etapa A5.

Como A3 e A4 são adaptações a metodologia e são executadas em paralelo com A1 e A2, elas são primeiramente abordadas:

- Descrever chão de fábrica (A3)

  Esta etapa tem como objetivo descrever o relacionamento entre os recursos de produção, suas características e fluxo de produção. Após a derivação ser autorizada têm-se como entrada de dados dessa etapa, as especificações dos equipamentos e a planta fabril do chão de fábrica. Uma equipe de especialistas deve auxiliar o utilizador derivador (usuário projetista, responsável por modelar o cenário), a fim de produzir um documento coerente com o cenário, que sirva como base no desenvolvimento da ontologia específica.

- Definir ontologia (A4)

  A representação do conhecimento acerca do chão de fábrica, utilizando ontologia, facilita sua interpretação pelos agentes que serão criados na etapa seguinte. A ontologia ADACOR é originalmente modelada utilizando um diagrama de classe UML, como ilustrado da figura 3.1. Para melhor representação da ontologia ADACOR, utiliza-se a ferramenta Protégé, especialmente desenvolvida para desenvolvimento de ontologias, ilustrada na figura 4.2.

  Como não é possível apresentar o processo completo de derivação, apresenta-se na ontologia ilustrada na figura 4.2, as instâncias de dois transportadores (agv01 e esteira01) e uma ferramenta (valvula01), de acordo com o cenário analisado. Essas entidades compõem os recursos de chão de fábrica, podendo sofrer perturbações, devem respeitar uma ordem de produção, compostas por operações e, basicamente seguir uma ordem de trabalho.

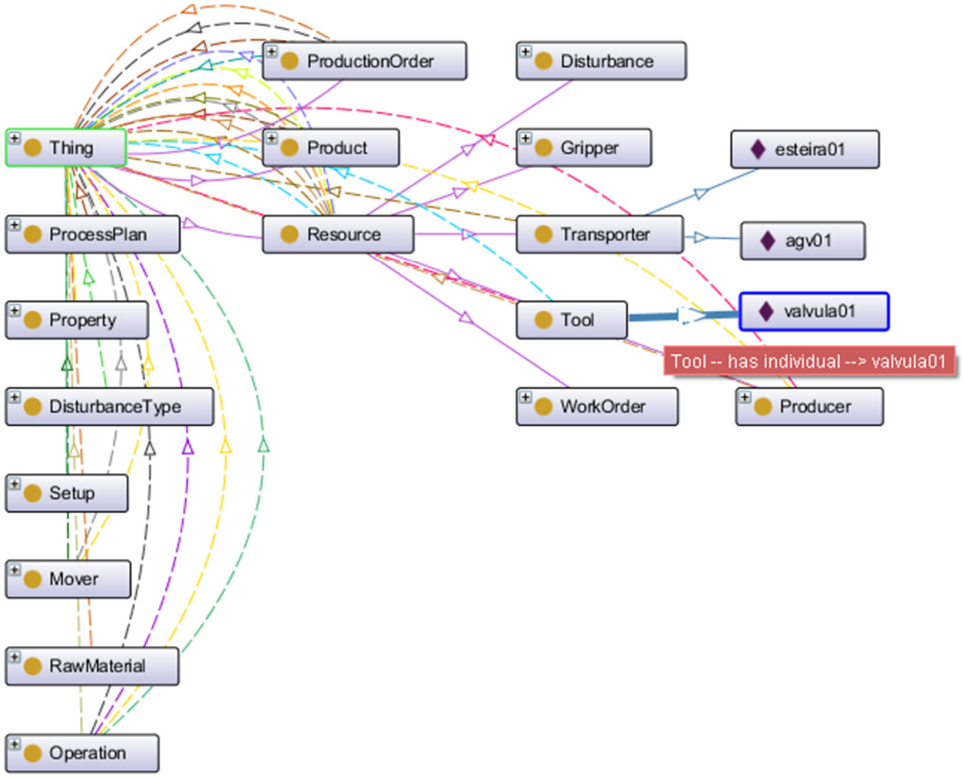

Figura 4.2. Ontologia ADACOR modelada no Protégé, com 3 instâncias do cenário.

As etapas A1 e A2 seguem o modelo especificado pelo *framework* O-MaSE, em que na etapa A1 será gerado um documento que contém as especificações de o quê e como serão realizados os processo do chão de fábrica. Nessa etapa são listados os requisitos funcionais e não funcionais do sistema. Na etapa A2, o documento gerado pela etapa A1 deve ser traduzido para um diagrama de classe UML, gerando a árvore de objetivos GMoDS(*Goal Model for Dynamic Systems*), ou seja, uma árvore AND/OR. Como a árvore AND/OR é representada por um diagrama de classes, cada classe deve ser

interpretada como uma folha da árvore. Nesse sentido, cada requisito é analisado e um conjunto de metas é traçado, afim de satisfazer os requisito impostos pela etapa de especificação.

A etapa A5 é responsável por juntar as informações da árvore AND/OR e da ontologia, gerando um código base, esqueletos dos agentes do SMA. Essa etapa conta com as seguintes subetapas:

- Definir Papéis – A relação entre os objetivos da árvore AND/OR é de cardinalidade n para n, ou seja, um objetivo pode precisar de vários papéis para ser atingido ou um papel pode satisfazer mais de um objetivo. Um diagrama UML é utilizado para ilustrar essa etapa.
- Definir Agentes – Esta subetapa define os agentes e quais os papéis que cada agente exercerá. Cada instância gerada na ontologia deve ser representada por um agente nessa etapa. Um diagrama UML é utilizado para ilustrar essa etapa.
- Definir Protocolos – Para representar as mensagens trocadas entre agentes ou entre agentes e papéis é utilizado um diagrama AUML, definindo os protocolos existentes entre eles.
- Definir Planos – A técnica de planejamento captura as ações e protocolos usados pelo agente para adquirir um objetivo, processo representado por um diagrama UML de transição de estados.
- Definir Políticas - Essa etapa identifica as propriedades dos requisitos do sistema, escrevendo-os em linguagem natural. Todas as restrições são identificadas e formalmente especificadas usando linguagem formal.

A metodologia O-MaSE não menciona o uso de ontologia para representação do domínio. Com o intuito de satisfazer essa deficiência, a etapa A4 instacia as entidades agv01, esteira01 e a valvula01, modeladas e representadas pela ontologia ADACOR, que devem ser convertidas em agentes na subetapa Definir Agentes da etapa A5, como pode ser observado no diagrama UML da figura 4.3.

Como mencionada anteriormente nessa mesma seção, os agentes lógicos de transporte e controlador_preenchimento devem ligar-se com os agentes instanciados na ontologia, controlando as ações desses agentes e se comunicando através de protocolos de comunicação estabelecidos na etapa de Definir Protocolos.

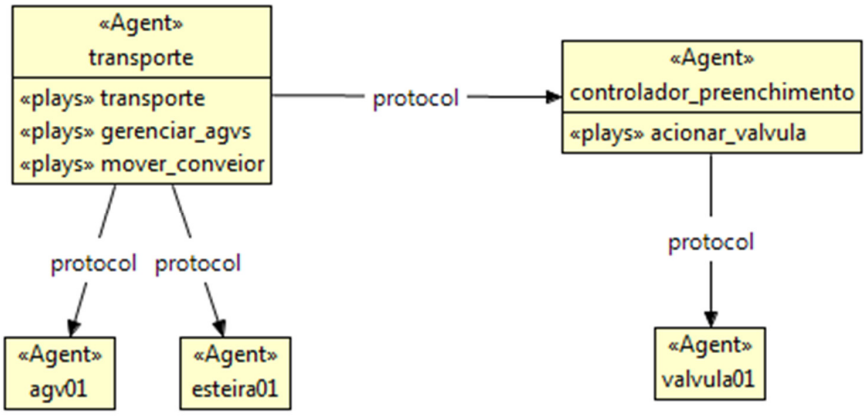

Figura 4.3. Subetapa Definir Agentes

5. Ambiente de desenvolvimento

Para implementar as etapas da metodologia mencionada em 4.1, utilizou-se a APE (Agent Process Editor) O-MaSE e os fragmentos de métodos: Especificação dos Requisitos, Modelo de Objetivos, Modelo de Papéis, Modelo de Agentes, Modelo de Protocolos, Modelo de Planos e Modelo de Políticas. Acrescentando ainda as etapas de Descrição de Chão de Fábrica e Ontologia, ilustrado na figura 5.1.

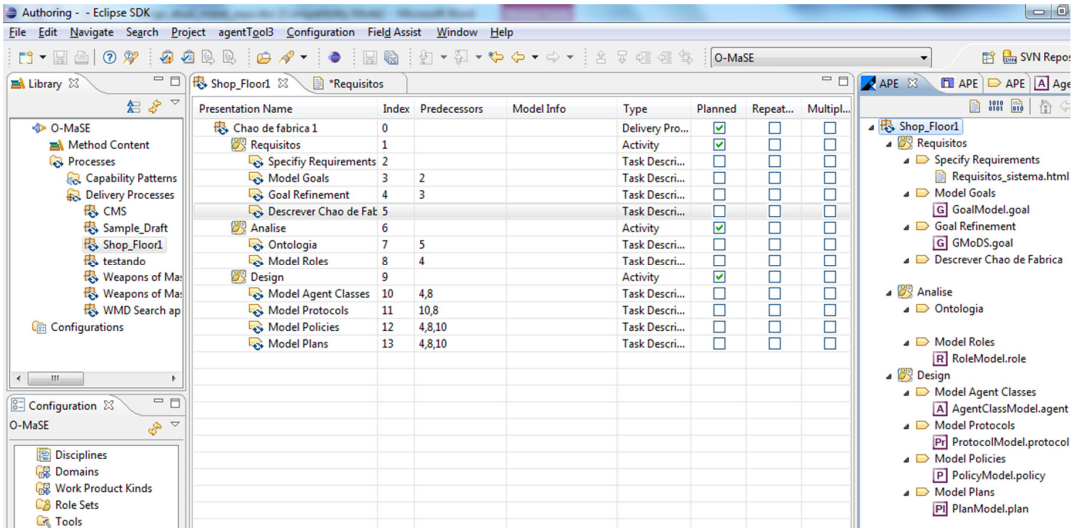

Figure 5.1. Configuração da APE O-MaSE

Cada etapa é desenvolvida usando documentos de especificação, ontologia, diagramas UML e diagramas AUML. Como resultado final da implementação obtém-se o código fonte, ou seja, um "esqueleto" dos agentes que posteriormente deve ser desenvolvida, implementando as ações, métodos e comportamentos específicos de cada agente, na figura 5.2 tem-se uma ideia de como ficaria o código fonte do esqueleto do agente gerado pelo APE O-MaSE. Os comentários no código proporcionam maior compreensão das instruções.

```
public class controlador_preenchimento extends Agent {//Classe do agente
  private static final long serialVersionUID = 1;
  protected void setup() {
    System.out.println("Agent "+getLocalName()+" started.");
    //Adiciona o comportamento geral e suas especificações
    addBehaviour(new MyGlobalBehaviour(this));
  }

private class MyGlobalBehaviour extends Behaviour{//Cria classe de comportamento global para esse agente
          private Agent a;
          private static final long serialVersionUID = 1;
          MyGlobalBehaviour(Agent a){
                this.a=a;
                }
          public void action(){
                  ACLMessage msg = myAgent.receive(); //Captura mensagem
                 if(msg!=null){
                         String content = msg.getContent(); // Captura o conteúdo da mensagem
                         if(content.compareTo("achieveacionar_valvula")==0){
                                //Adiciona um plano comportamental do agente
                                addBehaviour(new acionar_valvulaFSMBehaviour(a)); // Executa o plano de ação
                                }
                         }
                         else{ // Senão fica bloqueado
                                 block();
                         }
        -
```

Figura 5.2. Esqueleto de código do agente controlador de preenchimento

6. Conclusão

Nesse trabalho foi apresentada uma forma de derivação para um ambiente de manufatura, adaptando uma metodologia utilizada para desenvolvimento de SMA, usando ontologia para representar o conhecimento de um chão de fábrica.

Devido a complexidade encontrada neste ambiente e a necessidade de o SMA representar corretamente o sistema, atendendo as premissas estabelecidas neste trabalho, foi utilizada uma ontologia que oferece suporte à representação de características dos equipamentos e processos. Com isso, torna-se viável incrementar, remover ou substituir componentes do cenário, diminuindo a necessidade de reconstrução completa do sistema. Tais características também podem ser observadas durante as etapas da metodologia, nas quais novas características e processos podem ser adotados.

7. Referências

Borgo, S. and Leit, P. (2004) "The Role of Foundational Ontologies in Manufacturing Domain Applications System". In: On the Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE - journal, p. 670-688, Springer.

C. Schlenoff, A. Knutilla, S. Ray, (1996) "Unified Process Specification Language: Requirements for Modelling Process", NIST, Interagency Report 5910, Gaithersburg MD.

Deloach S.A. (1999) "Multiagent systems engineering: A methodology and language for designing agent systems", DTIC Document.

DeLoach S.A. (2006) "Multiagent Systems Engineering of Organization-based Multiagent Systems", 4th International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'05), St. Louis, MO. Springer LNCS Vol 3914, pp 109 - 125.

Gruber, T.R. (1995) "Towards principles for the design of ontologies used for knowledge sharing", Int. J. Human-Computer Studies, v. 43, n. 5/6.

Jennings, N.R. and Corera, J.M. and Laresgoiti, I. (1995) "Developing industrial multi-agent systems", Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), p. 423-430.

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

Monostori L, Váncza J, Kumara SRT (2006). "Agent-Based Systems for Manufacturing". *CIRP Annals - Manufacturing Technology*. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1660277306000053 [Accessed November 2, 2011].

Rabelo, Ricardo José. (1997) "Um enquadramento para o Desenvolvimento de Sistemas de Escalonamento Ágil da Produção – Uma abordagem Multiagente". Dissertação Apresentada Para a Obtenção do Grau de Doutor em Engenharia Eletrônica, Especialidade de Robótica e Manufactura Integrada. Universidade de Nova Lisboa, Faculdade de Ciências e Tecnologia, Lisboa.

Rabelo, R.J. (1998) "Uma Abordagem de Integração Balanceada para Sistemas Multiagente Industriais", Dissertação (Submetida em Concurso Público Para Professor Adjunto)--Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina, Florianópolis.

Shen W, Hao Q, Yoon HJ, Norrie DH. (2006) "Applications of agent-based systems in intelligent manufacturing: An updated review". Advanced Engineering Informatics. 415-431. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1474034606000292. Accessed September 19, 2011.

Teixeira, L.F.P. and Miranda, R. and de Ávila, E.S. (2001) "Engenharia de Sistemas Multiagentes : Uma Investigação sobre o Estado da Arte", Murilo Juchem e Ricardo Melo Bastos.

W. Shen, Q. Hao, H. Joong, and D.H. Norrie (2006) "Applications of agent-based systems in intelligent manufacturing : An updated review," Advanced Engineering Informatics, vol. 20, pp. 415-431.