Mário Lucio Roloff

Uma nova Abordagem para a Implementação de um Sistema Multiagente para a Configuração e o Monitoramento da Produção de Pequenas Séries

Tese submetida ao Programa de Pós-Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina para a obtenção do Grau de Doutor em Engenharia de Automação e Sistemas.
Orientador: Prof. Dr. –Ing. Marcelo Ricardo Stemmer

Florianópolis
2014

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

Roloff, Mário Lucio
Uma nova Abordagem para a Implementação de um Sistema Multiagente para a Configuração e o Monitoramento da Produção de Pequenas Séries / Mário Lucio Roloff ; orientador, Marcelo Ricardo Stemmer - Florianópolis, SC, 2014.
218 p.

Tese (doutorado) - 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. Manufatura. 3. Sistemas Multiagente. 4. JaCaMo. 5. SCADA. I. Stemmer, Marcelo Ricardo. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. III. Título.

Mário Lucio Roloff

Uma nova Abordagem para a Implementação de um Sistema Multiagente para a Configuração e o Monitoramento da Produção de Pequenas Séries

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

Florianópolis, 03 de outubro de 2014.

_____

Prof. Marcelo Ricardo Stemmer, Dr. -Ing.
Orientador

_____

Prof. Rômulo Silva de Oliveira, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia de Automação e Sistemas

Banca Examinadora:

_____

Prof. Jomi Fred Hübner, Dr.
Universidade Federal de Santa Catarina

_____

Prof. Enzo Morosini Frazzon, Dr. –Ing.
Universidade Federal de Santa Catarina

_____

Profa. Luciana Bolan Frigo, Dra.
Universidade Federal de Santa Catarina

_____

Prof. Marcos Marinovic Doro, Dr.
Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

_____

Prof. Carlos Eduardo Pereira, Dr. –Ing.
Universidade Federal do Rio Grande do Sul

Este trabalho é dedicado aos colegas de profissão e familiares.

AGRADECIMENTOS

Ao orientador, Prof. Dr. –Ing. Marcelo Ricardo Stemmer, que aqui encerra a etapa de orientação e formação que se iniciou na graduação em Engenharia de Controle e Automação Industrial.

Ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina onde encontrei o equilíbrio entre a academia e a indústria.

Aos familiares e especialmente a minha esposa Micheli Cristina Starosky Roloff que está presente em todos os momentos, bons ou maus.

A Deus!

Quando a escola progride, tudo progride.
Martinho Lutero (1483-1546)

RESUMO

A Produção de Pequenas Séries (PPS) é um tipo de manufatura caracterizado pela alta diversidade de produtos a serem produzidos associada a tamanhos de lotes reduzidos – possivelmente unitários. Neste sistema de produção as tecnologias empregadas para configuração e monitoramento do processo devem priorizar a produção sem defeitos, inclusive no primeiro item do lote, que pode ser o único. Falhas neste sistema de produção ou defeitos inseridos nos produtos facilmente inviabilizam economicamente todo o lote. Neste contexto, a linha de produção deve ser capaz de efetuar uma troca rápida de configuração para produzir um novo lote em um cenário de vários lotes de tamanho reduzido. É preciso também garantir completo monitoramento da produção do lote, sem falhas, ou quando ocorrer uma falha, ações corretivas devem ser executadas imediatamente. As pesquisas recentes demonstram que sistemas baseados em agentes é uma abordagem promissora para o cenário da PPS. Diante disso, a contribuição desta tese é a apresentação de uma nova Arquitetura de Referência para a implementação de Sistemas Multiagente na configuração e monitoramento da Produção de Pequenas Séries. A abordagem propõe uma Arquitetura de Referência, chamada MAS4SSP, baseada em uma solução unificada, sinérgica e com alto nível de abstração. Para garantir isto, a abordagem emprega como Modelo de Referência o *framework* JaCaMo que segue o paradigma orientado a multiagente (MAOP). A integração com a linha de produção é realizada com o emprego da tecnologia de comunicação *Web Service* que é utilizada pelo Sistema Multiagente (cliente) e por um sistema SCADA baseado em aplicação Web, o ScadaBR (servidor). A interface com o usuário pode ser desenvolvida como um recurso adicional na plataforma JaCaMo, ou pode ser realizada uma integração com sistemas legados de produção (como ERP, PCP, MRP) utilizando também a tecnologia de *Web Service*. Além da Arquitetura de Referência a tese apresenta um Modelo Genérico de Modelagem e Implementação que serve como guia para o desenvolvedor. Esta abordagem foi instanciada em um experimento simulado no contexto de uma PPS de Placas de Circuito Impresso (PCI). Por fim, os resultados e as conclusões sobre a Arquitetura de Referência e o Modelo Genérico de Modelagem e Implementação são apresentados em conjunto com sugestões de trabalhos futuros.

Palavras-chave: Manufatura. JaCaMo. ScadaBR.

ABSTRACT

Small Series Production (SSP) is a type of manufacturing characterized by a high diversity of products to be produced associated with a reduced batch sizes – possibly unitary. In this production system, the technologies employed for process setup and monitoring must prioritize production without defects, including the first item of the lot, which may be the one. Faults in this production system or defects in the product easily become the whole batch economically unfeasible. In this context, the production line should be able to monitor the process and make a quick change of the configuration to produce a new batch in a scenario of several batches with small sizes. The system should also ensure the batch production without failure or when a fault occurs, the corrective actions must be executed immediately. Recent research shows that agent-based systems are a promising approach for the SSP. The contribution of this thesis is the presentation of a new Reference Architecture for the implementation of Multi-Agent Systems to setup and monitoring Small Series Production. The approach proposes a Reference Architecture, called MAS4SSP, based on a unified, synergistic and high level of abstraction solution. To ensure this, the approach employs as a Reference Model the JaCaMo framework that follows the Multi-Agent Oriented Paradigm (MAOP). The integration with the production line is realized with the use of Web Service as communication technology between the Multi-Agent System (client) and a SCADA system – ScadaBR (server). The user interface can be developed as an additional resource in JaCaMo platform or can be an integration system with legacy production systems (such as ERP, MES, MRP) also using Web Service. Beyond the Reference Architecture, this thesis presents a Generic Modeling and Implementation Model which serves as a guide for the developer and that was instantiated in a controlled experiment - a SSP line of Printed Circuit Boards (PCB). Finally, the conclusions and perspectives about the Reference Architecture and the Generic Modeling and Implementation Model are presented together with suggestions for future works.

Keywords: Manufacturing. JaCaMo. ScadaBR.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE GRÁFICOS

LISTA DE ABREVIATURAS E SIGLAS

A&A – Agents & Artifacts
A&E – Agents & Environment
A&O – Agents & Organization
A&A&O – Agents & Artifacts & Organization
API – Application Program Interface
AOI – Automatic Optical Inspection
AOP – Agent-Oriented Programming
AOSE – Agent Oriented Software Engineering
BDI – Belief-Desire-Intention
BRAGECRIM – The Brazilian-German Collaborative Research Initiative on Manufacturing Technology
CAPES – Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CArtAgO – Common ARTifact infraestructure for Agents Open Environments
CNPq – Conselho Nacional de Desenvolvimento Cient. e Tecnológico
COGMET – Cognitive Metrology for Flexible Production of Small Series
DFG – Deutsch Forschungsgemeinschaft
EAI – Enterprise Application Integration
E&O – Environment & Organization
ERP – Enterprise Resource Planning
FINEP – Financiadora de Estudos e Pesquisas
FIPA – Foundation for Intelligent Physical Agents
FMS – Flexible Manufacturing System
HCBA – Holonic Component-Based Architecture
HMS – Holonic Manufacturing System
HTTP – HyperText Transfer Protocol
IDE – Integrated Development Environment
IEEE – Institute of Electrical and Electronics Engineers
IMS – Internacional Manufacturing System
IOP – Interaction Oriented Programming
JADE – Java Agent Development Framework
MAOP – Multi-Agent Oriented Programming
MAS – Multi-Agent System
MAS4SSP – Multi-Agent System for Small Series Production
MAS-ML – Multi-Agent System Modeling Language
OEM – Original Equipment Manufacturer
OOP – Object Oriented Programming
OOP – Organization Oriented Programming
PCB – Printed Circuit Board
PCI – Placa de Circuito Impresso

PCP – Planejamento e Controle da Produção
P&D – Pesquisa e Desenvolvimento
PPS – Produção de Pequenas Séries
RWTH-Aachen – Universidade Técnica de Aachen, Alemanha
S2i – Sistemas Industriais Inteligentes
SCADA – Supervisory Control And Data Acquisition
SE – Sistema Especialista
SMA – Sistema Multiagente
SOAP – Simple Object Access Protocol
UE – União Européia
UFSC – Universidade Federal de Santa Catarina
UML – Unified Modelling Language
XML – eXtensible Markup Language
WSDL – Web Services Description Language

SUMÁRIO

1. INTRODUÇÃO

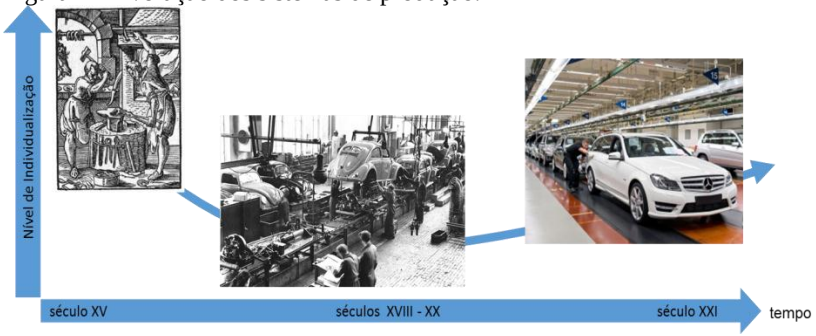
O atual nível de automação do processo já garante alta produtividade, escalabilidade, estabilidade e assegura a qualidade na produção em grande escala (DORO, 2009). Na produção em escala não existe variação de produtos (ou a variabilidade é limitada) e os lotes produzidos possuem várias unidades. Neste cenário de poucas trocas de produto e grande escala, o tempo de troca (*Setup Time*) é pequeno quando comparado com o tempo de ciclo de produção (*Production Lead Time*). Com isso, é possível corrigir uma falha em um ou vários produtos durante o tempo de produção e ainda é possível garantir que o lote não seja inviabilizado economicamente. Contudo, quando o tamanho do lote diminui (podendo ser até unitário) associado a uma diversidade de produtos o tempo de troca passa a ser fundamental no planejamento da produção – um tempo de troca alto pode inviabilizar economicamente o lote - e a ocorrência de uma falha no produto pode significar que todo o lote está com falha, acarretando gastos em retrabalho que também podem inviabilizar economicamente o lote. Diante do posto, o modelo de automação da produção em escala, que prioriza a repetibilidade, deve ser substituído por um modelo que priorize 'fazer certo da primeira vez'. É necessário possibilitar a alteração rápida, automática e autônoma de configurações de software e hardware da linha de produção. Para que isto seja possível, o sistema de produção precisa ser distribuído, autônomo, dinâmico e integrador de diferentes domínios de conhecimento.

Neste contexto surge o conceito de Produção de Pequenas Séries (PPS) que pode possuir diferentes nomenclaturas dependendo do enfoque do autor: pequenos lotes, multiproduto, *small lot, small batch, small run, high mix and low volume, non-repetitive*, entre outros (JADHAV, 2005; LIN *et al.*, 1997). De modo geral, a PPS é voltada a fabricação de uma grande variedade de produtos (comparativamente a produção em massa), para um baixo volume de produção, considerando ainda um curto espaço de tempo de configuração e produção (HITOMI, 1996). É importante ressaltar que não existe um valor máximo fixado – número de unidades do produto – para que a produção seja caracterizada como uma PPS.

A PPS pode ser considerada a origem das tecnologias de produção com o artesão atendendo cada pedido de forma personalizada. Após, seguiu-se o período de industrialização com a produção em massa. Atualmente o mercado retorna ao passado com produtos cada vez mais individualizados, contudo, hoje o mestre artesão é substituído

por equipamentos de alta tecnologia associados tanto ao produto como a produção (Figura 1) (HEIKE *et al.*, 2001; STORCH e LIM, 1999).

Figura 1 – Evolução dos sistemas de produção.



O principal problema com o surgimento das PPS, trata-se da necessidade de introduzir novas abordagens que visam garantir a configuração e o monitoramento da produção e posterior garantia da qualidade do produto final (como também da produção), uma vez que o *setup* fixo e o controle estatístico de falhas (muito utilizado na produção em massa) não são aplicáveis neste contexto de pequenas séries. Devido a baixa escala de produtos a serem produzidos, não é possível garantir o mesmo *setup* dos equipamentos para produtos diversos e também a quantidade necessária para a amostragem de falhas não é aplicável em pequenas séries. Visando ainda a qualidade da produção, o procedimento de controle e inspeção apresentam paradigmas diferentes da produção em massa, principalmente, devido a constante mudança de projetos inseridos na linha de produção (PAVIM, 2011). Dessa forma, torna-se necessária a busca por novas abordagens que, permitam flexibilizar a automação da produção e, ainda, possibilite a rápida adequação da linha de produção aos requisitos de um novo *setup* de produção de acordo com os requisitos de fabricação de um novo (diferente) produto.

As pesquisas recentes (BRECHER, 2012; LEITÃO *et al.*, 2011; LEITÃO *et al.*, 2013; LEITÃO *et al.*, 2013b; PAVIM, 2011) demonstram que uma abordagem baseada em agentes é promissora para o ambiente flexível e altamente reconfigurável requerido pela PPS. O Sistema Multiagente (SMA) é um sistema composto por um conjunto de agentes de software que interagem com o objetivo de alcançar metas individuais ou coletivas (DEMAZEAU, 1995).

> Um agente é algo capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por meio de atuadores. (RUSSEL & NORVIG, 2003).

Neste cenário, este trabalho propõe uma solução integrada, estruturada na forma de uma Arquitetura de Referência, para dar suporte a implementação de um sistema multiagente para configuração e monitoramento da PPS. O propósito geral desse trabalho é apresentar uma nova abordagem que utiliza um *framework* de sistema multiagente, que considera os agentes, o ambiente e a organização de forma integrada e sinérgica, com um *framework* de supervisão e aquisição de dados do processo e uma interface com o usuário específica ou legada.

1.1. PROBLEMÁTICA GERAL

O problema principal em que esta tese de doutorado se baseia é na ausência de uma abordagem de implementação de sistema multiagente que considere os diferentes tipos de entidades existentes em um sistema de produção e proponha um modelo de implementação que integre de forma sinérgica estas entidades e não de uma forma *ad hoc*.

Neste contexto, a pergunta que guia a realização desta pesquisa consiste em: *a abordagem proposta é uma solução técnica para a configuração e monitoramento da produção de pequenas séries?*

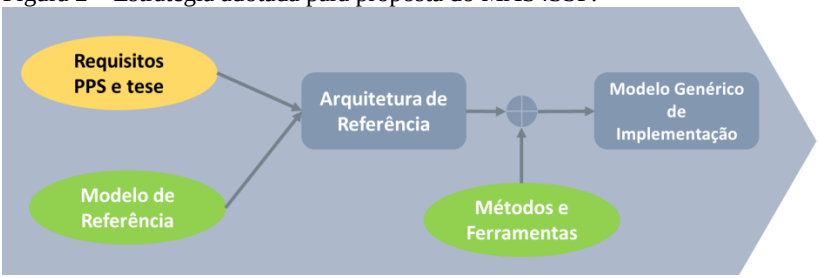
A concepção de uma Arquitetura de Referência é vista como uma forma de apresentar um padrão genérico para um projeto. Esta deve abordar os requisitos para o desenvolvimento de soluções, guiado pelo Modelo de Referência e por requisitios desejáveis de forma a atender as necessidades do trabalho de pesquisa (BASS et., 2003). Desta forma, a Figura 2 apresenta a estratégia adotada nesta pesquisa para propor uma Arquitetura de Referência que após o uso de ferramentas e métodos gera um Modelo Genérico de Modelagem e Implementação de um sistema multiagente para configuração e monitoramento da PPS.

Esta abordagem proposta, nomeada de MAS4SSP (*Multiagent System for Small Series Production*), deriva em uma ferramenta computacional capaz de configurar e monitorar as diferentes entidades presentes nos diversos níveis organizacionais da linha de produção disponiblizando informações sobre a qualidade do produto e o desempenho dos processos.

Ao identificar o problema existente, uma revisão do estado da arte foi realizada para a verificação de possíveis soluções para este

problema. Diversos materiais foram encontrados, alguns com características relevantes para a solução deste problema, e que de fato, foram utilizadas para compor a abordagem proposta (esse assunto é detalhado no capítulo de revisão do estado da arte deste documento). Porém, nada foi encontrado na literatura para resolver o problema aqui trazido em sua totalidade (camadas de abstração especializadas e integradas de forma sinérgica), caracterizando assim, o ineditismo do trabalho desenvolvido nesta tese de doutorado.

Figura 2 – Estratégia adotada para proposta do MAS4SSP.



O desenvolvimento desta tese de doutorado traz contribuições tanto científicas quanto empresariais.

No âmbito científico contribui para a área de Inteligência Artificial, mais precisamente para a área de Sistema Multiagente, onde apresenta uma implementação do Paradigma Orientado a Multiagente em um cenário industrial. Outra área onde essa tese contribui é a de Produção de Pequenas Séries, uma vez que apresenta uma Arquitetura de Referência para a configuração e monitoramento da produção baseada em um sistema multiagente.

No âmbito empresarial esta tese contribui oferecendo uma nova abordagem, integrada, sinérgica e open source para a implementação de sistema multiagente para a configuração e o monitoramento da produção de pequenas séries. Esta abordagem apresenta um nova abordagem para que as empresas se adequem a produção de pequenas séries. Estima-se que o SMA para PPS implementado seguindo esta abordagem melhore a confiança das empresas na tecnologia de SMA. Desta forma, esta tese contribui também para os clientes que adquirem um produto resultante de uma PPS, que agora podem ter o conhecimento da maturidade dos processos na qual o produto foi desenvolvido. Contribui também para que outras empresas de outros modelos de produção possam implementar esta abordagem em estruturas mais completas satisfazendo

as necessidades de toda a cadeia de suprimentos (visto como trabalho futuro).

1.2. OBJETIVOS

Os objetivos de uma pesquisa são elaborados para caracterizar o seu alcance e utilizados para delimitar o seu problema. Os objetivos estabelecidos nesta pesquisa são apresentados na forma de objetivo geral (forma genérica) e objetivos específicos (forma exata) (SILVA e MENEZES, 2005).

1.2.1. Objetivo Geral

Visando responder à pergunta de pesquisa numa perspectiva tecnológica, o objetivo geral desta tese é propor uma nova abordagem para a implementação de sistema multiagente para a configuração e o monitoramento da produção de pequenas séries.

1.2.2. Objetivos Específicos

A partir do objetivo geral desta proposta de tese, os seguintes objetivos específicos são propostos:

- definir o Modelo de Referência para o sistema multiagente;
- conceber uma Arquitetura de Referência para a configuração e o monitoramento da produção de pequenas séries;
- apresentar uma arquitetura de implementação genérica baseada no Modelo de Referência e na Arquitetura de Referência;
- experimentar a proposta em um protótipo computacional associado à Arquitetura de Referência concebida, para servir de instrumento de avaliação da proposta.

1.3. ESCOPO DO TRABALHO

Este trabalho de pesquisa está inserido no âmbito do programa de cooperação Brasil-Alemanha BRAGECRIM (*Brazilian-German Collaborative Research Initiative on Manufacturing Technology*) iniciado em 2009. O objetivo principal do programa BRAGECRIM é *apoiar projetos de pesquisa que prioritariamente resultem no*

desenvolvimento de soluções inovadoras para melhorar a produtividade, a qualidade e a sustentabilidade das indústrias brasileiras e alemãs. Outros objetivos são a troca de conhecimento e a formação de recursos humanos através de missões de trabalho e estudo de pesquisadores e estudantes de ambos os países.

O projeto COGMET (*Cognitive Production Metrology for Flexible Small Series Production*) é um dos projetos de pesquisa do programa BRAGECRIM que conta com a participação das seguintes instituições: CERTI/LabElectron e UFSC/DAS/S2i participam pelo lado brasileiro e o RWTHAachen/WZL pelo lado alemão. O projeto possui como objetivo principal: *o desenvolvimento de uma nova geração de sistemas para metrologia e garantia da qualidade da produção com capacidades adaptativas e um alto grau de percepção cognitiva relacionada ao produto e ao processo* (PFEIFER *et al.*, 2010).

O experimento simulado para avaliação técnica da proposta empregou como cenário a linha de PPS de Placas de Circuito Impresso (PCI) do laboratório-fábrica CERTI/LabElectron localizado em Florianópolis/SC.

1.4. ASPECTOS METODOLÓGICOS

A pesquisa científica é a realização concreta de uma investigação planejada e desenvolvida de acordo com as normas consagradas pela Metodologia Científica (GIL, 2010). Como a metodologia define um conjunto de etapas para vencer na investigação de um fenômeno, para realizar uma pesquisa com rigor científico é preciso proceder a elaboração de um plano de trabalho de ordem planejada, ordenada e conclusiva (CERVO *et al.*, 2006). Sendo a metodologia a base que define os fundamentos para os estudos científicos, esta subseção tem por objetivo enquadrar a presente pesquisa de acordo com a sua Natureza, Abordagem, Objetivos e quanto ao Método da Pesquisa (SILVA e MENEZES, 2005). A Tabela 1 apresenta o enquadramento metodológico deste trabalho de pesquisa.

Tabela 1 – Enquadramento metodológico desta pesquisa.

Critério	Classificação	Justificativa
Natureza	Aplicada	Pela pesquisa prever a aplicação dos resultados deste trabalho em problemas existentes.
Abordagem	Qualitativa	No uso de técnicas na qual geram dados qualitativos.
Objetivos	Exploratória	Pela investigação exploratória de um cenário, levantamento de informações e criação de uma premissa a ser sanada com a tese.
Método	Indutivo	Por considerar que pelas observações individuais, considera-se que a abordagem soluciona um problema existente.

1.5. ESTRUTURA DO DOCUMENTO

A organização deste documento tenta refletir o objetivo essencial de uma Tese de Doutorado, que é o de identificar um problema relevante e o ponto de ineditismo, mostrar o estado em que a ciência encontra-se nesse quesito, introduzir o que se pretende atingir para se resolver o problema (ou contribuir para) e com qual visão metodológica, e descrever o como se pretende atingir o resultado desejado, apontando-se os pontos críticos, recursos e cronograma.

Assim sendo, este documento está dividido em 6 capítulos. O capítulo 1 apresenta o tema de pesquisa, o problema a ser abordado, aos objetivos do trabalho, e a metodologia a ser adotada.

O capítulo 2 traz os conceitos referentes às áreas envolvidas no problema abordado para a concepção da nova abordagem.

Já o capítulo 3 discute o estado da arte na área de sistemas multiagente aplicados na manufatura.

O capítulo 4 detalha a proposta deste trabalho, apresentando o Modelo de Referência, os Requisitos, a Arquitetura de Referência, os Métodos e Ferramentas finalizando com o Modelo Genérico de Modelagem e Implementação da proposta.

No capítulo 5, é detalhada a implementação de um experimento baseado no Modelo Genérico de Modelagem e Implementação. Os resultados do experimento são discutidos.

O capítulo 6 realiza uma análise geral da proposta através da aplicação de questionário e publicação de artigos científicos. Também

discute o atendimento as expectativas do trabalho de pesquisa, as conclusões e sugestões para trabalhos futuros.

Por fim, o documento finaliza com os elementos pós-textuais, referências, anexos e apêndices.

2. CONCEITUAÇÃO TEÓRICA

Os conceitos envolvidos neste trabalho de pesquisa estão centrados em dois temas principais: a Produção de Pequenas Séries e os Sistemas Multiagente. Em primeiro lugar a Produção de Pequenas Séries será caracterizada e os desafios ligados a automatização deste sistema de produção serão descritos. Em seguida é apresentado o Sistema Multiagente delimitando a pesquisa ao Paradigma Orientado a Multiagente.

2.1. PRODUÇÃO DE PEQUENAS SÉRIES

No contexto das indústrias de manufatura de larga escala, como na fabricação de automóveis, o dinamismo está presente nos estágios iniciais de projeto e de prototipagem do produto, já a fase de produção em massa é caracterizada pela relativa estabilidade. Já a Produção de Pequenas Séries, como na fabricação de aviões e navios, é caracterizada por um alto grau de personalização dos produtos (DORO, 2009).

2.1.1. Definição

Apesar de não existir uma definição amplamente reconhecida e abrangente para Produção de Pequenas Séries na literatura, dos trabalhos pesquisados uma ideia geral pode ser extraída: *a Produção de Pequenas Séries se concentra na fabricação de uma **grande variedade** de produtos em um **curto período de tempo** e com um **baixo volume** de produção (possivelmente unitário)* (LEITÃO *et al.*, 2013b; LIN *et al.*, 1997; PYZDEK, 1993; TANG e BARNETT, 1994). Ainda o tempo para o processamento de um lote de produção completo é aleatório e os diferentes produtos possuem diferentes níveis de complexidade, o que provoca constantes mudanças no fluxo de produção.

Assim, as características mais comuns que descrevem o sistema produtivo que neste trabalho é chamado de Produção de Pequenas Séries (PPS) são: **o baixo volume de produção; o curto tempo de produção por lote; a alta variedade de produtos**.

Deve ficar claro que não é possível definir valores exatos para as características apresentadas acima para que a produção seja classificada como de pequeno lote. Na realidade, dependendo do tipo de produto, do ambiente de manufatura considerado, do estágio de fabricação, do grau de controle sobre o processo entre outros fatores, as quantidades a considerar podem variar significativamente.

2.1.2. Características

Normalmente em uma empresa que atua na PPS, o *layout* do chão de fábrica é customizado, pois cada produto compartilha os recursos das operações com outros produtos. Os produtos são encaminhados no chão de fábrica de pequenas séries para as várias máquinas (rota de produção), sendo que os trabalhadores devem ter uma habilidade relativamente alta para efetuar diferentes tarefas, pois não existe um fluxo único e padronizado (BLACK, 2001).

A PPS normalmente está presente em empresas de pequeno e médio porte que fabricam produtos ou componentes para empresas de manufatura maiores (OEM[1]) ou prestam serviços para grandes empresas de diversos setores (CM[2]). Alguns exemplos típicos onde ocorre PPS são (JURAN *et al.*, 1998):

- equipamentos complexos grandes, tais como: equipamentos agrícolas, aeronaves e automóveis especiais;
- componentes, como por exemplo: placas eletrônicas, dispositivos pequenos, embalagens plásticas, componentes eletrônicos e componentes automotivos;
- produtos acabados simples, tais como: livros, sapatos e móveis;
- produtos produzidos sob encomenda, tais como: ferramentas, peças fundidas, plásticos moldados e peças de máquinas.

Algumas das principais dificuldades na PPS são destacadas a seguir (DORO, 2009; HITOMI, 1996; JURAN *et al.*, 1998; MAHONEY, 1997; PAVIM, 2011; SLACK *et al.*, 1997):

- **complexidade no gerenciamento do estoque:** geralmente a estratégia "produzir por pedido" (*Make to order*[3]) é utilizada. Como consequência, problemas como a falta de materiais e insumos podem ocorrer durante a produção. Estes problemas podem originar deficiências de qualidade no produto, atrasos de entrega e, conseqüentemente, lucros menores. Além disso, a variedade de itens produzidos, a diversificação dos volumes de produção e as datas de entrega provocam a necessidade de

[1] OEM - *Original Equipment Manufacturer*: empresa detentora da marca

[2] CM - *Contract Manufacturer*: empresa prestadora de serviços

[3] Produção baseada nas encomendas recebidas (MTO – *Make to order*) – A etapa de produção só é iniciada após se ter recebido o pedido de encomenda. Neste caso, faz sentido que a interação entre o cliente e o produto seja extensiva, podendo o produto sofrer modificações durante a fase de produção. O tempo de entrega tende a ser de médio a longo prazo.

produzir vários produtos simultaneamente, resultando normalmente numa elevada quantidade de estoque intermediário.

- **longo *lead time*:** a necessidade de realizar um novo *setup* quase sempre que o produto entra em produção, a espera por compras de insumos e as constantes mudanças de especificações do produto resultam em um longo tempo para que o serviço seja totalmente executado, desde sua solicitação até sua entrega.
- **alta variedade de operações:** a grande diversidade de produtos gera diferentes formas de conversão dos materiais de entrada em produtos acabados, que variam desde simples até complicadas. Consequentemente, os operadores devem ter uma habilidade relativamente alta para efetuar diferentes tarefas.
- **falta de previsibilidade**: as frequentes trocas nas especificações do produto, nos volumes, nas datas, o atraso na compra de insumos, entre outros, provocam incertezas nos resultados do processo.
- **dificuldade de planejamento e programação da produção:** a carência de informações precisas, a dinâmica das ordens de pedido, as listas de materiais (BOM[4]) ora extensas e complexas, ora curtas e simples, a variedade de processos de produção e uma série de outros fatores, dificultam um ótimo planejamento e programação da produção, bem como, a estimação dos custos.
- **planejamento e controle da qualidade dinâmicos:** devido à incerteza de um ótimo controle do processo para toda a variedade de produtos produzidos, executar as atividades de planejamento e controle de qualidade em uma PPS se torna complicado.

As estratégias para a PPS são diferentes das aplicáveis à produção em massa, por isso, a PPS exige uma mudança de paradigma de produção. Por exemplo, os fabricantes de altos volumes ganham mercado baseado primeiramente no custo de seus produtos. Já os fabricantes de pequenas séries conquistam o mercado quando possuem a capacidade de atender o que seus clientes desejam. Similarmente, enquanto que a estratégia de demanda de fluxo é muito importante para

[4] BOM: *Bill of Materials* – Lista de Materiais é uma lista de todos os materiais necessários para a produção de um determinado produto.

a produção em massa, ela não é apropriada para a PPS, pois é muito difícil balancear a capacidade produtiva quando os tempos de produção variam demasiadamente. Num ambiente de PPS, podem não existir dados suficientes para tornar o controle estatístico do processo confiável. Neste caso, é necessário encontrar alternativas para o controle da qualidade, como por exemplo, a combinação de instruções de trabalho detalhadas e inspeções automáticas. Adicionalmente, técnicas de gerenciamento da qualidade, como *design review*[5], auditoria de processos, acompanhamento do produto em campo, ajudam assegurar o sucesso da produção (BURGESS, 1999; MAHONEY, 1997; SPROVIERI, 2004).

2.1.3. Desafios

Esta grande diversidade e variabilidade de produtos e a constante introdução de novos produtos no ambiente da PPS acarretam uma série de desafios (BRECHER, 2012; DORO, 2009; LEITÃO *et al.*, 2013b; MARÍK *et al.*, 2013):

- a falta de previsibilidade sobre o processo e o comportamento do produto;
- a constante criação/atualização da documentação da qualidade, tais como os planos de inspeção;
- o aumento dos ciclos de instalação (*setup*) e nenhum ou apenas alguns produtos que podem ser utilizados para o treinamento da linha de produção (*golden product*[6]);
- o curto período de tempo para observar e corrigir o processo durante a produção;
- a dificuldade para a reutilização da informação e realização de ações corretivas;
- a falta de dados para a tomada de decisão.

Entre estes, o maior desafio da PPS é garantir a qualidade e a viabilidade econômica da produção neste contexto de grande diversidade de produtos, lotes reduzidos e ínumeras mudanças no *setup* da linha de produção.

Neste contexto, este trabalho de pesquisa atuará nos seguintes desafios relacionados a PPS:

---

[5] *Design review*: técnica de gerenciamento usada para a avaliação de um projeto proposto, a fim de garantir que o projeto do sistema ou produto atende aos requisitos do cliente.

[6] *Golden Product*: é um produto que se sabe previamente que está correto e é utilizado para ajustar os parâmetros de fabricação e inspeção dos equipamentos e sistemas da linha de produção.

- **reduzir o tempo de *setup*** por meio da configuração automatizada da linha de produção;
- **garantir a qualidade na produção** através do monitoramento do processo de produção e do produto;
- **aprender baseado em experiências** anteriores e similariedades adicionando um sistema especialista que auxiliará na análise das causas de uma falha e quais a ações a serem tomadas para evitar que aconteça novamente.

2.1.4. Pesquisas

Os principais avanços para a PPS tiveram início na década de 1950, no Japão, mais especificamente na Toyota, com uma nova abordagem que passou a ser conhecida como manufatura enxuta (WOMACK, JONES e RODOS, 1998). A manufatura enxuta introduziu importantes conceitos para a PPS, entre os quais se destacam:

- engenharia simultânea;
- tecnologia de grupo (GT);
- troca rápida de ferramentas (SMED);
- modularidade;
- células de produção;
- desdobramento da função qualidade (QFD);
- sistemas flexíveis da manufatura (FMS);
- manufatura integrada por computador (CIM).

A busca de um melhor aproveitamento e racionalização da PPS tem motivado os pesquisadores a desenvolver novas técnicas gerenciais e de produção. Nota-se que os caminhos para se atingir a eficiência na PPS passam cada vez mais por uma crescente flexibilidade dos sistemas de manufatura, não apenas de equipamentos, mas também na tomada de decisões de projeto, planejamento, agendamento, manuseio de materiais e gerenciamento das informações (SETCHI e LAGOS, 2004). Neste contexto, uma composição de experiências, valores, informações e idéias são pesquisadas nas academias e aplicadas nas rotinas, processos e práticas das organizações. A Tabela 2 resume os principais temas pesquisados sobre PPS e os trabalhos relacionados.

Conclui-se que a PPS é um sistema de produção atual que visa atender as exigências do consumidor de um produto de alta tecnologia personalizado a custo acessível. Para alcançar isto a PPS deve fazer intenso uso de tecnologias que permitam o controle do processo e também da qualidade do produto final (MARÍK *et al.*, 2013). As soluções para conceber um sistema de configuração e monitoramento

que seja capaz de atender as demandas específicas da Produção de Pequenas Séries ainda é um campo de pesquisa importante a ser explorado – o desenvolvimento de arquiteturas de integração do processo e garantia da qualidade modificáveis, extensíveis, reconfiguráveis, adaptáveis e tolerantes a falhas (SETCHI e LAGOS, 2004). Esta demanda motiva a pesquisa e o desenvolvimento de uma nova geração de sistemas de manufatura e sistemas que possam responder dinamicamente aos requisitos dos clientes e as condições do ambiente de produção de pequenas séries.

Tabela 2 – Pesquisas sobre PPS.

Tema de Pesquisa	Autores
Otimização na alocação de recursos de fabricação	(BLAZEWICZ, DOMSCHKE e PESCH, 1996; LUCERO, 2001; ZATTAR, 2003).
Controle estatístico do processo (CEP)	(ARENAS, 2005; ELAM, 2008; JADHAV, 2005; WHEELER, 1991; ZHU, 2005).
Tecnologia de Grupo	(AL-SALTI e STAHAN, 1984; CHENG, 1989).
Metodologias para o projeto de produto	(HOWEL e YOUNG, 1996; PFEIFER e MERGET, 1999)
Técnicas para auxílio no *setup* da linha de produção	(DORO, 2009; LIAN, COLOSIMO e DEL CASTILLO, 2006).
Técnicas para a garantia da qualidade	(DAVÉ *et al.*, 2003; JIANG *et al.*, 2008; LI e WANG, 2008).
Sistemas baseados em agentes para flexibilização da PPS	(LEITÃO *et al.*, 2013b; MARÍK *et al.*, 2013; PAVIM, 2011).

2.2. SISTEMAS MULTIAGENTE

A área de pesquisa de Agentes e Sistemas Multiagente é ampla e diversificada, desta forma, optou-se pela delimitação à área deste trabalho de pesquisa.

2.2.1. Definições

Sistemas Multiagente (SMA) são sistemas compostos de um conjunto de agentes de software que interagem com o objetivo de alcançar metas individuais ou coletivas (DEMAZEAU, 1995).

Mas o que são agentes? Ou agentes de software? Várias definições para agentes têm sido propostas ao longo das últimas décadas, algumas fornecendo noções gerais sobre o agente, outras fornecendo conceitos mais detalhados. Uma definição mais restrita foi proposta por Wooldridge (2000). A definição é composta de duas partes: a primeira afirma a natureza do *agente*, e em seguida, define-se o que é um agente inteligente. Assim:

> (i) Um agente é um sistema computacional localizado em algum ambiente que pode executar ações autônomas para alcançar os seus objetivos projetados.

Nesta definição *autonomia* significa que os agentes[7] possuem controle sobre o seu próprio estado interno e sobre o seu comportamento, sendo capaz de agir sem a intervenção de seres humanos ou de outros sistemas. Um agente geralmente possui um conjunto possível de ações que representam a sua capacidade de modificar o seu ambiente, também chamado de capacidades do agente (WOOLDRIDGE, 2000). Além disso, os agentes podem atuar em diferentes tipos de ambiente, tais como: um ambiente dinâmico que pode mudar enquanto um agente está atuando ou um ambiente estático que permanece inalterado durante a ação do agente, um ambiente discreto em que o tempo é tratado de forma discreta, ou um ambiente contínuo em que o tempo é tratado de uma forma contínua. (RUSSEL e NORVIG, 2003).

Um ponto importante a observar é que um agente normalmente possui controle parcial sobre seu ambiente. Isto significa que, do ponto de vista do agente, a mesma ação realizada duas vezes pode resultar em

[7] A autonomia para decidir se uma ação será ou não executada é diferente nos sistemas baseados em agentes dos sistemas baseados em objetos. Esta decisão de executar ou não uma ação reside com o agente que recebe o pedido e pode aceitar ou não, graças a sua autonomia. Nos sistemas baseados em objetos a decisão está com o objeto que chama o método do objeto que segue uma rotina. Esta distinção entre objetos e agentes é destacada na seguinte frase: "os objetos fazem isto gratuitamente, os agentes fazem isto por dinheiro" (WOOLDRIDGE, 2000 p.35).

resultados diferentes e também as ações podem não ter o resultado desejado. Assim, pode-se afirmar que um agente normalmente esta inserido em um ambiente não determinista (WOOLDRIDGE, 2000).

> (ii) "Um agente inteligente é aquele que é capaz de ações autônomas flexíveis a fim de alcançar seus objetivos projetados" (WOOLDRIDGE, 2000).

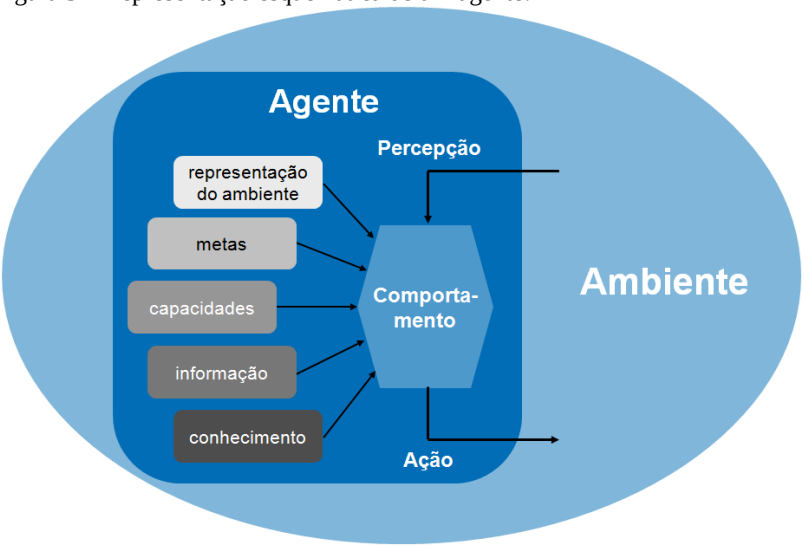
Nesta definição, *flexibilidade* significa reatividade, pró-atividade e sociabilidade. Devido a sua reatividade, os agentes inteligentes são capazes de observar seu ambiente e responder rapidamente às mudanças que ocorrem no ambiente. Com base na sua pró-atividade, os agentes inteligentes são capazes de tomar iniciativas para cumprir suas metas. Finalmente, usando suas habilidades sociais, estes agentes são capazes de interagir uns com outros e também com os seres humanos, de forma a alcançar seus objetivos de projeto (metas) (WOOLDRIDGE, 2000). Esta definição de *agente inteligente* é a adotada nesta tese e o termo *agente inteligente* será referido apenas como *agente* na sequência deste texto.

Para escolher uma ação específica a ser realizada dentro de seu ambiente, o agente leva em consideração a sua percepção atual e/ou uma sequência inteira de percepções históricas sobre este ambiente até o momento da sua ação. Isso significa que o comportamento do agente é descrito por uma função de agente, que mapeia uma seqüência de percepções dada uma ação (Figura 3) (RUSSEL e NORVIG, 2003).

Um agente de software pode ser compreendido como sendo um indivíduo, constituído por um módulo de software fechado, que interage de forma contínua com o seu ambiente. Contudo, grande parte dos sistemas baseados em agentes são projetados com vários agentes que desempenham diferentes tarefas e podem cooperar uns com os outros ou então competir para realizar suas tarefas e alcançar suas metas. E para que os agentes interajam é preciso algumas condições básicas (GÖHNER *et al.*,2004; WAGNER, 2008):

- os agentes devem ter conhecimento se outros agentes fazem parte do ambiente;
- a meta de cada agente, em um ambiente de interação, deve ser conhecida;
- um meio comum de comunicação deve estar disponível através do qual os agentes podem interagir uns com os outros.

Figura 3 – Representação esquemática de um agente.



Fonte: Adaptado de Russel & Norvig (2003)

O Sistema Multiagente (SMA) é um sistema em que vários agentes interagem para executar tarefas e atingir metas individuais e/ou coletivas. Assim, o SMA apresenta como padrão de interação uma coordenação que pode ser orientada para cumprir suas metas (*goal-oriented*) e/ou orientada para cumprir determinada tarefa (*task-oriented*). E estas interações podem ocorrer de forma cooperativa ou competitiva. Na primeira situação, vários agentes buscam combinar as suas ações para atingir uma meta específica, como em um grupo, uma vez que tal objetivo seja considerado inatingível para um único agente. Na segunda situação, vários agentes agem para alcançar o que somente uma parte deles poderá alcançar (WEISS, 2001).

Além disso, os SMA são tipicamente imersos em um ambiente constituído por um conjunto de recursos, como dados, equipamentos, e softwares. As interações com o ambiente ocorrem através de alguns tipos de sensores e atuadores, ou seja, mecanismos para perceber e atuar sobre alguma parte do ambiente (ZAMBONELLI; JENNINGS; WOOLDRIDGE, 2001).

Diante do posto, a organização e a padronização das arquiteturas de software é importante para tornar os sistemas multiagente flexíveis e até mesmo para permitir que diferentes sistemas baseados em agentes,

apesar de utilizarem diferentes plataformas e sistemas operacionais, ou linguagens de programação possam interagir uns com os outros (BELLIFEMINE *et al.*, 2001).

2.2.2. Paradigma VOGAL

O Paradigma VOGAL de Demazeau (1995) propõe um modelo conceitual para representar de uma forma geral os sistemas baseados em agentes. Demazeau (1995) divide o sistema baseado em agentes em quatro dimensões - agentes, ambientes, interações e organizações - que correspondem às vogais A (*Agent*), E (*Environmet*), I (*Interaction*), O (*Organization*) em inglês, respectivamente (Figura 4). Além disso, Ricordel e Demazeau (2002) estenderam o paradigma vogal propondo um conjunto de fases de desenvolvimento, que devem ser utilizadas na construção de um SMA baseado nas quatro dimensões. Eles propõem quatro fases: análise, projeto, desenvolvimento e implementação.

No Paradigma VOGAL, os agentes podem ser simples autômatos, bem como complexos sistemas baseados em conhecimento. O ambiente, na maioria dos casos, consiste de um espaço de recursos que depende do domínio da aplicação. As interações variam desde interações baseadas em fenômenos da física (temperatura, pressão...) até atos de fala. Finalmente, as organizações representam comportamentos desde aqueles inspirados por estudos biológicos até as organizações inspiradas por metáforas sociológicas, que envolvem leis e hierarquias sociais complexas.

2.2.2.1. Dimensão: Agente

Quatro tipos de arquitetura foram propostas para aplicações baseadas em agentes: baseada em lógica, reativa, crença-desejo-intenção (BDI) e camadas de agentes (WOOLDRIDGE, 2000).

Os agentes baseados em lógica são aqueles onde o raciocínio e a tomada de decisão ocorrem através de dedução lógica, como mostrado em (LESPERANCE *et al.*, 1996). Os agentes baseados na arquitetura reativa implementam a tomada de decisões usando um mapeamento direto de situação para a ação, como em (BROOKS, 1986; STEELS, 1990), enquanto os agentes BDI (*Belief-Desire-Intention*) implementam o raciocínio e a tomada de decisão com base na manipulação de alguma representação de suas crenças, desejos e intenções (BRATMAN *et al.*, 1988; RAO e GEORGEFF, 1992). Finalmente, os agentes baseados na arquitetura de camadas são aqueles sistemas que implementam o

raciocínio e a tomada de decisão através de várias camadas de software, cada uma das camadas se relaciona com o Ambiente em diferentes níveis de abstração (FERGUSON, 1995).

Figura 4 – Dimensões do Paradigma Vogal.

A arquitetura BDI se tornou o padrão de fato para os modelos de agentes. A arquitetura BDI é a base do padrão IEEE-FIPA. Este é o tipo de arquitetura utilizada neste trabalho de pesquisa.

2.2.2.2. Dimensão: Ambiente

Atualmente é consenso geral na comunidade de pesquisa multiagente que o ambiente é uma entidade essencial para o SMA. No entanto, muitos pesquisadores deixam de integrar o ambiente como uma abstração de alto nível nos modelos e nas ferramentas multiagente; já outros pesquisadores minimizam as responsabilidades e importância do ambiente. Como consequência deste fato, estima-se que um conjunto potencial de aplicações que poderiam ser desenvolvidas utilizando SMA é negligenciado.

Frameworks populares como Jade (BELLIFEMINE *et al.*, 2001), Jack (HOWDEN *et al.*, 2001), Retsina (SYCARA *et al.*, 2001) ou Zeus (NWANA *et al.*, 1999) reduzem a representação do ambiente a um simples sistema de troca de mensagens ou a uma infraestrutura

marginal. Metodologias para desenvolvimento de sistemas baseados em agentes bem populares, como MESSAGE (EVANS *et al.*, 2001), Prometheus (PADGHAM e WINIKOFF, 2002) ou Tropos (BRESCIANI *et al.*, 2004) oferecem suporte para alguns elementos básicos do ambiente, no entanto elas não consideram o ambiente como uma entidade de alto nível de abstração. Na literatura sobre SMA (HUHNS e STEPHENS, 1999; RUSSEL e NORVIG, 2003; WOOLDRIDGE, 2000) apenas apresentam muito brevemente o ambiente. Mesmo nas especificações da FIPA é difícil encontrar uma especificação para o ambiente, além de um meio para transporte de mensagens. Restringindo a representação do ambiente e delimitando a sua interação com a dimensão de agente, negligencia-se um vasto potencial de possibilidades para aplicação do paradigma do SMA.

Por outro lado, existem pesquisadores que atuam na área de sistemas multiagente que integram o ambiente como uma entidade de alto nível de abstração. Em alguns SMA, o ambiente é uma entidade ativa com seus próprios processos e que pode alterar seu próprio estado independentemente da atividade dos agentes que fazem parte do SMA. Inspirados nos sistemas biológicos, vários pesquisadores têm argumentado que o ambiente pode servir como uma memória compartilhada para os agentes do SMA. Segundo estes pesquisadores, isto pode "aliviar" os agentes de continuamente manterem atualizado o seu conhecimento sobre o sistema. Além disso, permitiria que os agentes utilizassem o ambiente como um excelente meio de coordenação indireta do sistema. Várias aplicações práticas têm demonstrado como o ambiente pode contribuir para gerenciar problemas complexos. Existem exemplos relevantes desde a década de 90 até hoje em domínios como: sistemas de cadeia de suprimentos (*Supply Chain Management*) (SAUTER e PARUNAK, 1999), suporte de rede (BONABEUAU *et al.*, 1998), sistemas *peer-to-peer* (BABAOGLU *et al.*, 2002) controle de produção (PARUNAK, 2001), logística (WEYNS *et al.*, 2005), sistema flexível de manufatura (TICHÝ *et al.*, 2012), etc. Nestes trabalhos os autores afirmam que exploração do ambiente em um SMA resulta em melhores soluções e que a dimensão do ambiente é uma alternativa promissora para lidar com a crescente complexidade e dinamismo dos sistemas.

Estudar SMA com foco na dimensão ambiente é uma tarefa difícil. Durante este trabalho de pesquisa foram encontradas duas dificuldades: (i) o termo ambiente possui vários significados diferentes, causando uma grande confusão; (ii) as funcionalidades associadas com o

ambiente são muitas vezes tratadas de forma implícita, ou integradas nas plataformas de SMA como uma funcionalidade marginal.

A confusão sobre o que compreende o ambiente é causada principalmente por misturar conceitos e plataformas. Às vezes, os pesquisadores se referem ao ambiente como a entidade lógica de um SMA em que os agentes e outros objetos/recursos são incorporados. Por vezes, a noção de ambiente é usada para se referir à plataforma no qual o SMA é executado. Outras vezes, o ambiente até pode aparecer como um recurso disponível na plataforma em que o SMA é executado. As funcionalidades do ambiente são muitas vezes tratadas implicitamente ou de uma forma *ad-hoc*, isto indica que a comunidade científica dos SMA falha ao não tratar o ambiente como uma entidade de alto nível de abstração. Conclui-se que o ambiente deve ser tratado como um módulo independente que encapsula suas próprias responsabilidades em um SMA independentemente dos agentes.

2.2.2.3. Dimensão: Organização

Lemaitre e Excelente (1998) sugerem que a pesquisa no campo de SMA pode ser dividida em duas classes de acordo com a abordagem adotada para representar os aspectos sociais: **SMA centrado em agente** e **SMA centrado em organização**.

A abordagem centrada no agente propõe a representação dos aspectos sociais do SMA através de conceitos voltados para o comportamento dos agentes como uma entidade social: com intenções coletivas (FERBER, 1999) com compromissos sociais (CASTELFRANCHI, 1995) ou usando o raciocínio social (SICHMAN e DEMAZEAU, 2001). Esta abordagem é focada fortemente no agente e engloba pesquisas sobre formalismos para representar o conhecimento individual do agente. No entanto, esta abordagem não define explicitamente as organizações. É importante notar que os métodos de modelagem e implementação de sistemas baseados em agentes mais populares, como Tropos (BRESCIANI *et al.*, 2004), MaSE (WOOD e DELOACH, 2001), PASSI (COSSENTINO, 2005), Prometheus (PADGHAM e WINIKOFF, 2002), e ASEME (SPANOUDAKIS e MORAITIS, 2010) permitem desenvolver SMA centrados nos agentes.

Por outro lado, a pesquisa sobre SMA centrados em organização adota uma visão sociológica e organizacional para a modelagem do SMA. Esta abordagem abrange a especificação de tipos distintos de grupos de agentes que fazem parte de organizações e de equipes. Estes grupos estabelecem regras e normas para restringir o comportamento do

agente, bem como para especificar os direitos e deveres do agente independentemente de um modelo de agente em particular.

O conceito básico na abordagem do SMA centrado na organização é a **organização** (ou o agente organizacional). O agente organizacional é composto por um conjunto de metas, normas e funcionalidades, além de uma estrutura interna de componentes (ou subsistemas) (LEMAITRE e EXCELENTE, 1998).

O desenvolvimento de um SMA que emprega a abordagem organizacional pode ser classificado de acordo com a evolução da organização durante o ciclo de vida do SMA. Os SMA centrados em organização são divididos em duas categorias: a engenharia orientada a agentes (*agent-oriented engineering*) e o SMA orientado a organização (*organization-oriented MAS*) (HÜBNER, 2003; PICARD *et al.*, 2009).

A primeira categoria engloba as abordagens que lidam com especificação organizacional durante a fase de projeto do SMA. Nesta categoria existe um modelo explícito para representar as organizações. No entanto, estas abordagens não permitem que os agentes modifiquem os aspectos organizacionais durante a execução do SMA. Não é possível nesta categoria a criação ou a eliminação de papéis, a alteração da hierarquia da organização ou dos objetivos. Exemplos de tais abordagens são encontrados nos métodos: GAIA (ZAMBONELLI *et al.*, 2001), Ingenias (PAVON *et al.*, 2005) e O-MaSE (GARCIA-OJEDA *et al.*, 2008). Por exemplo, Gaia fornece funcionalidades de analisar e projetar o SMA com base em aspectos organizacionais, tais como papéis, estrutura e normas. No entanto, Gaia não especifica como adicionar novas funções ou alterar a estrutura organizacional de forma dinâmica, ou seja, após a implementação do SMA com ele em execução.

Na segunda categoria estão as abordagens de desenvolvimento de SMA que permitem especificar aspectos organizacionais durante a fase de projeto do SMA e, possivelmente, alterá-los ao longo da execução do SMA. Estas mudanças são realizadas através de ações de agentes - os chamados atos organizacionais - que podem modificar a organização (alterando a estrutura organizacional, ou adicionando funções). Exemplos de *frameworks* que seguem esta abordagem são: MOISE+ (HÜBNER; SICHMAN; BOISSIER, 2007, 2008), Islander (ESTEVA *et al.*, 2002) e OperA (DIGNUM, 2004).

2.2.2.4. Dimensão: Interação

É comum em um SMA que os agentes necessitem interagir para atingir suas metas. Em alguns casos o SMA pode ser composto pelos

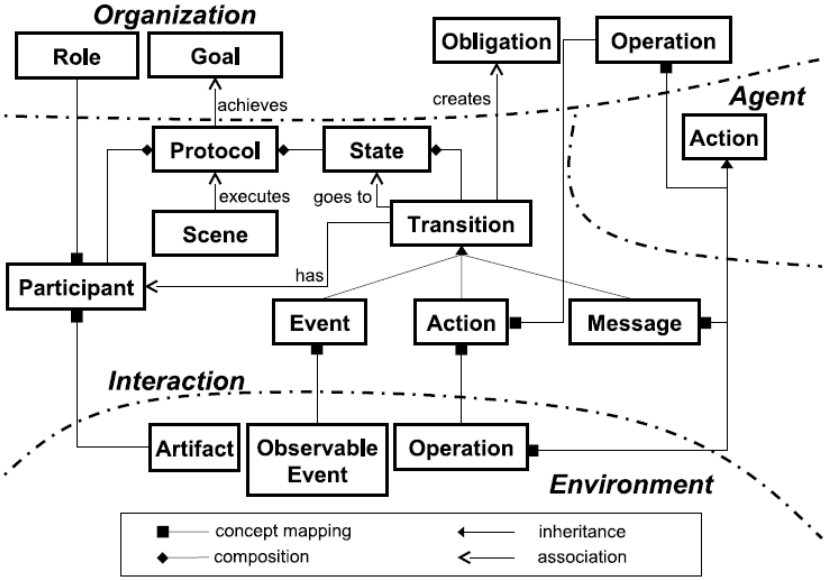
agentes, pelo ambiente, pela interação e pela organização como apresentado em (DEMAZEAU, 1995). Em um SMA que segue este paradigma – VOGAL – a interação não está presente somente entre os agentes, mas também com o ambiente e a organização que fazem parte do sistema. Neste paradigma, além dos agentes interagirem com outros agentes, eles também podem interagir (agir e sentir) com os recursos do ambiente. Existem muitos trabalhos sobre agentes, organização e ambiente. Existem ferramentas para especificar, modelar e implementar cada uma dessas dimensões. Por exemplo, um SMA que utiliza o *framework* CArtAgO (RICCI *et al.*, 2007) é capaz de construir o seu próprio ambiente, a dimensão organizacional pode ser construída utilizando *frameworks* como AGR (FERBER *et al.*, 2005), ISLANDER (ESTEVA *et al.*,2004) e Moise (HÜBNER *et al.*, 2007). E finalmente, os agentes podem ser programados se utilizando plataformas como GOAL (HINDRIKS, 2009), JADE (BRAUBACH *et al.*, 2005), 2APL (DASTANI *et al.*, 2008), Jason (BORDINI *et al.*, 2007) entre outras. Existem inclusive ferramentas que unificam as dimensões em um framework único como é o caso de EIS (BEHRENS *et al.*, 2011) e JaCaMo (BOISSIER *et al.*, 2011). Esta separação em dimensões auxilia na manutenção, modularidade, organização, reuso do código, etc. Também é fácil perceber que cada um destes componentes pode ser programado por diferentes desenvolvedores, o que facilita a divisão de tarefas.

Existem inúmeras propostas que defendem a ideia da Interação como uma abstração de alto nível deste modelo multidimensional, pregando a necessidade de um *framework* específico para a sua modelagem e programação (DELOACH e VALENZUELA, 2006; MILLER e McBURNEY, 2007; MILLER e McGINNIS, 2008; SILVA *et al.*, 2004; SINGH, 2011). Contudo, nenhum dos trabalhos citados apresenta um *framework* para modelagem e implementação da Interação em um ambiente multidimensional. O trabalho mais próximo disso é o de Zatelli (2014) que apresenta um modelo conceitual integrando a Interação às outras dimensões do Modelo de Referência JaCaMo. O trabalho de Zatelli (2014) propõe um modelo conceitual, uma linguagem de programação e a integração com o *framework* JaCaMo. O objetivo é fornecer um mecanismo para institucionalizar como os agentes podem interagir com os diferentes elementos de um SMA que segue o paradigma VOGAL para alcançar os objetivos organizacionais.

Neste trabalho de pesquisa a Interação não está inserida como uma abstração de alto nível, um dos motivos é a ausência de um *framework* que integrasse a Interação – o trabalho de Zatelli (2014)

ainda não estava concluído – e outro motivo é a relativa simplicidade (quanto a quantidade e conteúdo) das interações entre os agentes, o ambiente e a organização na abordagem a ser proposta para a configuração e o monitoramnto da PPS. O objetivo desta seção foi apresentar brevemente a dimensão da Interação cuja integração com a abordagem será sugerida como trabalho futuro quando se pensar em uma abordagem multiagente para todos os níveis hierárquicos da empresa que segue a PPS.

Figura 5 – Modelo proposto de interação entre as quatro dimensões.



Fonte: Zatelli *et al.* (2014)

2.2.3. Desenvolvimento de um Sistema Multiagente

Weiss (2001) identifica duas razões para se utilizar o desenvolvimento de sistemas baseados em agentes. Em primeiro lugar, os SMA estão entre as abordagens que são adequadas para o gerenciamento de sistemas computacionais modernos, que são na maioria dos casos distribuídos, grandes, abertos e heterogêneos. Os SMA são sistemas conectados uns com os outros e com os seus usuários, como a Internet. Para lidar com este cenário, os sistemas exigem alto nível de interações entre as entidades dos sistemas e

precisam atuar como “agentes de software” e não como meros “módulos de software”. Em segundo lugar, o SMA permite a exploração de características típicas dos seres humanos como conceitos sociológicos e psicológicos. Estes conceitos permitem ao SMA implementar o processo de negociação, a formação de uma organização e a resolução de conflitos. Constata-se também que o SMA é uma abordagem adequada para o desenvolvimento, a análise de teorias e o teste de modelos de interatividade em sociedades humanas.

No entanto, é preciso resaltar que o paradigma baseado em agentes não é “uma bala de prata”, uma vez que não existe qualquer evidência que sugira que tal abordagem ofereça uma solução universal para o desenvolvimento de sistemas computacionais (WOOLDRIDGE e JENNINGS, 1999).

Tomando uma perspectiva de programação, os sistemas baseados em agentes podem ser implementados com: Programação Orientada a Agentes (*Agent Oriented Programming*), inicialmente introduzido em (SHOHAM, 1993), que abrange principalmente a conceituação de agente; Programação Orientada a Interação (*Interaction-Oriented Programming*) (HUHNS, 2001), Programação Orientada a Organização (*Organisation-Oriented-Programming*) (BOISSIER *et al.*, 2006) Modelos e Linguagens de Coordenação (*Coordination Models and Languages*) (OMICINI *et al.*, 2001), Programação Orientada a Times (*Team-Oriented Programming*) (PYNADATH *et al.*, 1999), Programação Normativa (*Normative Programming*) (DASTANI *et al.*, 2008), Programação de Ambientes (*Environment Programming*) (RICCI *et al.*, 2011).

Quando o foco é a programação, para cada proposta são necessárias ferramentas computacionais, como linguagens e *frameworks* com um nível adequado de abstração (BORDINI et al. 2005) que contribuem para a usabilidade do paradigma. Os paradigmas da programação tradicionais que se tornaram úteis são aqueles que possuem ambientes de programação que correspondam diretamente as abstrações e aos conceitos utilizados para modelar o sistema computacional. É o mesmo neste novo paradigma, é essencial que existam ambientes para enfrentar a complexidade que é o contexto da programação de um SMA. Constata-se que a programação de um SMA pode ser realizada eficientemente unificando as dimensões e integrando todos os conceitos apresentados na Figura 4. Isto já foi discutido na literatura sobre modelagem e metodologia que seria possível (DEMAZEAU, 1995; STRATULAT, 2009). Por fim, argumenta-se que é necessário um padrão de método/metodologia para a modelagem do

SMA e também é necessária a criação de um ambiente de programação que eficientemente e 'facilmente' integre todas estas dimensões.

Para representar este desejo de um ambiente multidimensional para programação multiagente, utiliza-se o termo *Multi-Agent Oriented Programming* (MAOP) (BOISSIER *et al.*, 2011) para enfatizar a integração deste conjunto de abstrações originárias das dimensões propostas pelo Paradigma VOGAL (DEMAZEAU, 1995) para o desenvolvimento de um sistema multiagente: o agente (A), o ambiente (E), a interação (I) e a organização (O). A integração deve cobrir todas as dimensões que se acredita serem relevantes para o desenvolvimento de programas multiagente.

Assim, em comparação com a Programação Orientada a Agentes, originalmente proposta em (SHOHAM, 1993), o objetivo aqui é possuir abstrações de alto nível não só para os agentes, mas também para os níveis de organização, interação e ambiente. No nível da organização, têm-se abstrações como grupo, papel, norma e esquema social (que é também referenciado como plano social, ou plano global parcial, ou atribuição de tarefas e vários outros termos na literatura de agentes). No nível do agente, estão abstrações como: crença, meta, plano, ação e percepção. No nível de interação estão abstrações como: o ato de fala, baseado em mensagens e protocolos de interação. No nível de ambiente, além da semântica das ações e percepções, há uma abstração chamada *artefato* que pode ser usada para representar recursos, ferramentas, serviços e uma variedade de outros **componentes não autônomos** de um sistema multiagente.

A integração destas dimensões deve ser mais do que simplesmente unir um conjunto de abstrações. Por um lado, precisa-se adequar a semântica dos conceitos e métodos de programação, por outro lado, o diferencial é que se permita o emprego dos conceitos de forma sinérgica melhorando a expressividade global da abordagem de programação do sistema multiagente respeitando os conceitos de cada dimensão.

2.2.4. Modelagem e Implementação de Sistemas Baseados em Agentes

A modelagem e a implementação de sistemas baseados em agentes pode ser: orientada a objetos; orientada a agentes ou orientada a multiagentes. Existem inúmeras plataformas, linguagens de programação, APIs (*Application Programming Interfaces*) e *frameworks* que seguem um ou outro paradigma (BERGENTI *et al.*, 2004;

BERNON et al., 2005; BOISSIER *et al.*, 2011; BORDINI *et al.*, 2005; CIANCARINI e WOOLDRIDGE, 2001; GÖHNER *et al.*, 2004; JENNINGS, 2000; TRENCANSKY & CERVENKA, 2005; WAGNER *et al.*, 2003; WEISS e JAKOB, 2005).

Já a Engenharia de Software Orientada a Agentes (*Agent-Oriented Software Engineering* – AOSE) engloba a identificação, definição e aplicação de linguagens, métodos, ferramentas, técnicas e plataformas de desenvolvimento para apoiar o desenvolvimento de sistemas baseados em agentes (BERGENTI *et al.*, 2004). Na literatura sobre AOSE existem vários exemplos de plataformas de desenvolvimento orientadas a agentes, entre elas Zeus (NWANA *et al*., 1999.), FIPA-OS (POSLAD *et al.*, 2000), JADE (*Java Agent Development Framework*) (BELLIFEMINE *et al.*, 2001), Jack (COBURN, 2001) e *Jason* (BORDINI *et al.*, 2005). Algumas delas envolvem linguagens de programação orientada a objetos como JADE e Jack, que são construídas baseadas em Java, enquanto outras plataformas envolvem realmente linguagens de programação orientadas a agente, como é o caso de *Jason* baseada em *AgentSpeak* (RAO e GEORGEFF, 1992).

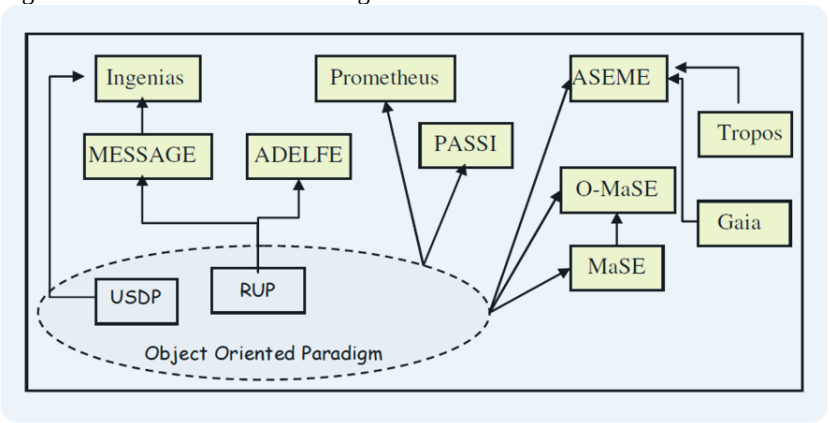
Vários métodos[8] foram propostos na última década com o objetivo de guiar e estruturar o desenvolvimento de sistemas orientados a agentes: Gaia (ZAMBONELLI *et al.*, 2003), Tropos (BRESCIANI *et al*., 2004), MaSE (WOOD e DELOACH, 2001), O-MaSE (GARCIA-OJEDA *et al.*, 2008), MESSAGE (CAIRE *et al.*, 2001), Prometheus (PADGHAM e WINIKOFF, 2002), ADELFE (BERNON *et al.*, 2002), PASSI (COSSENTINO, 2005), Ingenias (PAVON *et al.*, 2005), e ASEME (SPANOUDAKIS e MORAITIS, 2010) estão entre os métodos AOSE mais conhecidos.

A Figura 6 foi inspirada em (GIORGINI; HENDERSON-SELLERS, 2005) e descreve a relação entre os métodos AOSE através de uma perspectiva de diagrama, bem como a sua relação com os métodos mais populares para desenvolvimento de software utilizando o paradigma orientado a objetos – RUP *(Rational Unified Process*) e USDP (*Unified Software Development Process*). Em primeiro lugar, a figura mostra que alguns métodos AOSE são baseados em métodos orientados a objeto: ADELFE e MESSAGE são construídos em RUP, enquanto Ingenias é construído sobre USDP. Além disso, a figura representa métodos como Prometheus, PASSI, MaSE, O-MaSE e

[8] É importante destacar que muitos métodos AOSE se classificam como metodologias, ao invés de somente métodos.

ASEME, que envolvem técnicas provenientes do paradigma orientado a objetos, principalmente com base em UML. Em segundo lugar, a Figura 6 indica que alguns métodos AOSE são construídos derivados de métodos anteriores: Ingenias é fortemente baseada na MESSAGE, O-MaSE é uma extensão do MaSE e ASEME envolve algumas técnicas originalmente propostas por Tropos e Gaia.

Figura 6 – Relacionamento entre alguns dos métodos AOSE.



Fonte: Adaptado de Giorgini & Henderson-Sellers (2005)

Uma análise comparativa destes métodos (GOMEZ-SANZ *et al.*, 2004; STURM e SHEHORY, 2004; TRAN & LOW, 2005) mostra que de um lado alguns dos métodos propõem fases e ferramentas de desenvolvimento semelhantes e outras, por outro lado, possuem características específicas, como técnicas de modelagem e ferramentas específicas. Por exemplo, alguns dos métodos como Tropos, adotam uma notação gráfica descrever os modelos, enquanto outros, como Gaia, adotam uma notação matemática.

Uma descrição completa e detalhada destes métodos AOSE pode ser encontrada em (BERGENTI *et al.*, 2004; GIORGINI e HENDERSON-SELLERS, 2005).

3. ANÁLISE DO ESTADO DA ARTE: AGENTES E MANUFATURA

Os sistemas multiagente podem ser encontrados em diversos campos de aplicação, tais como a Internet, jogos de computador, telecomunicações, *dataminning*, controle e supervisão de processos, gestão de energia, logística, etc. (CIANCARINI *et al.*, 2001; GRAND *et al.*, 1997; JENNINGS, 2000; LEITÃO, 2009; LEITÃO e VRBA, 2011; LEITÃO *et al.*, 2013; MAES, 1995; MAGEDANZ *et al.*, 1996; NANCY *et al.*, 2009; PARUNAK, 2001; SCHUH *et al.*, 2005; SHEN *et al.*, 2006; WAGNER, 2003). A conclusão nos vários campos de aplicação é que este paradigma de programação é interessante para aplicações descentralizadas, dinâmicas e complexas.

Dentro de ambientes de produção flexíveis, pelo menos três fatores contribuem para a utilização de sistemas baseados em agentes (BRECHER, 2012; PAVIM, 2009):

- a infraestrutura de um sistema baseado em agentes permite uma fácil inserção de novos agentes (software e/ou hardware) no sistema, sem a necessidade de reprogramar a lógica de controle;
- permite implantar a solução em um ambiente distribuído, multiplataforma, com diferentes componentes de software e hardware e diferentes protocolos de comunicação. Os agentes tornam este sistema heterogêneo em um meio comum com o mesmo protocolo de comunicação;
- a cooperação ou a competição entre os diferentes agentes é o que garante autonomia na operação do sistema de controle da produção.

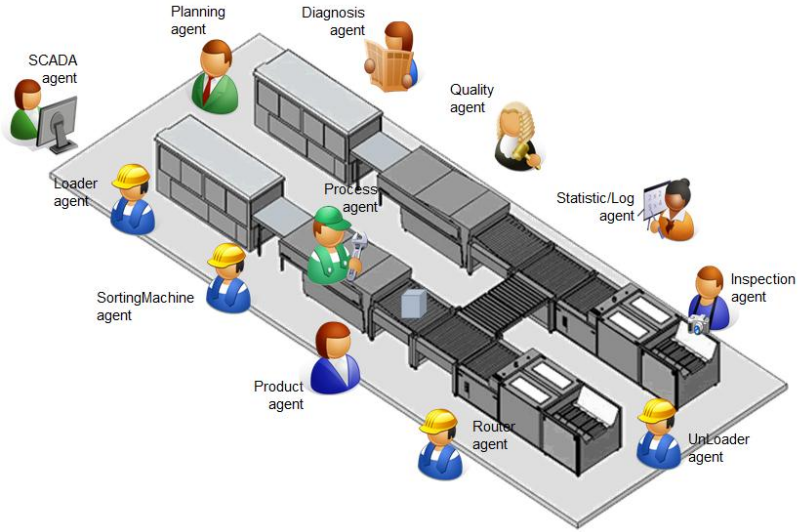
A teoria dos sistemas baseados em agentes contribui para uma melhoria da inteligência ou da cognição do sistema a ser implementado. Além disso, os sistemas baseados em agentes propiciam que a aplicação seja executada em um ambiente dinâmico, flexível, autônomo e adaptativo (ZAEH *et al.*, 2010).

3.1.1. Agentificação na Indústria

Uma abordagem tradicional e predominante para a concepção de SMA na manufatura é a Agentificação. A Agentificação consiste no processo de integrar software/hardware de equipamentos existentes na

produção utilizando SMA (MARÍK e LAZANSKY, 2007; PECHOUCEK e MARÍK, 2008). Na agentificação os objetos do paradigma Orientado a Objetos são transformados em Agentes. Por esta técnica o "objeto agente" pode representar uma máquina, uma esteira, um sistema de Planejamento e Controle da Produção (PCP), enfim todo o componente da produção que atenda parcialmente as características de um agente ou seja importante para o cumprimento do objetivo do projeto (Figura 7).

Figura 7 – Agentificação de uma linha de produção.



Uma desvantagem desta abordagem é a explosão no número de agentes no sistema, que tende a tornar a implementação complexa. Contudo, a desvantagem principal desta técnica é a limitação do emprego das potencialidades do paradigma orientado a agentes. Neste tipo de abordagem, limita-se a criar um caminho de comunicação único para os componentes do sistema que antes não poderiam se comunicar por falta de interoperabilidade. As características de autonomia, interação, aprendizagem, orientação por metas e adaptabilidade, por exemplo, não são exploradas com todo o seu potencial.

A vantagem desta técnica é a não necessidade de mudança de paradigma de modelagem e implementação e o treinamento em um paradigma totalmente novo, como o Orientado a Multiagente. Nesta abordagem, o projetista concebe o sistema baseado em agentes com a abordagem do paradigma Orientado a Objetos, inclusive utiliza o

mesmo método de modelagem – método UML (JACOBSON *et al.*, 1999). Esta abordagem torna o processo de desenvolvimento do sistema baseado em agentes mais rápido devido a familiaridade do projetista com o paradigma e ainda conta com o auxílio das potencialidades (metodologia e ferramentas consolidadas e eficientes) da Orientação a Objetos (OO)

3.1.2. Pesquisas sobre Agentes na Indústria

Iniciativas de implementação de sistemas baseados em agentes na indústria não é uma novidade, algumas iniciativas já foram citadas anteriormente Seção 2.1.4 e outras serão citadas aqui. A compilação das aplicações mais representativas não é uma tarefa fácil, porque são minoria as aplicações que abrangem um domínio completo da manufatura – por exemplo, controle da produção – e a maioria das aplicações relatadas nos trabalhos são desenvolvidas em ambientes controlados – acadêmicos/laboratoriais – e não em plantas industriais. A boa notícia é que este cenário vem mudando nos últimos 5 anos como descrito no trabalho do Leitão *et al.* (2013).

A intenção aqui não é ser exaustivo e apresentar cada proposta e tampouco apresentar todas as propostas que existem. O critério foi relacionar as propostas onde o uso de um SMA representou passo importante para cumprir o objetivo do projeto e àqueles projetos que foram citados com maior freqüência em outros trabalhos de pesquisa. O foco do trabalho de pesquisa foi examinar a aplicação da tecnologia baseada em agentes para resolver os problemas industriais. Uma linha do tempo com as principais arquiteturas e metodologias desenvolvidas e também um mapeamento das aplicações é apresentado na Figura 8.

Dentre as iniciativas algumas são listadas na Tabela 3 e na Tabela 4. Mesmo que estes projetos atuem nas mais variadas áreas da manufatura, o objetivo comum deles é proporcionar maior controle e flexibilidade a produção. Estes projetos foram estudados em mais detalhe para extrair informações para a proposição da abordagem MAS4SSP. A legenda das tabelas significa que os projetos contribuíram no conceito, no modelo/arquitetura e/ou com técnicas ou tecnologias para a proposição do MAS4SSP.

Figura 8 – Linha do tempo das aplicações baseadas em agentes na indústria.

Fonte: Adaptado de Leitão *et al.* (2013)

Tabela 3 – Referências de SMA na manufatura para o MAS4SSP – Parte I.

Nome do Projeto	Instituição Envolvida	Domínio de Aplicação	Campo de Aplicação	Contribuição
Production 2000+	Daimler Chrysler, Schneider	Controle da Manufatura	Indústria	○ ⊕ ●
Car Body Painting	Daimler-Benz	Controle da Manufatura	Indústria, Protótipo	○
BHP Billiton	Rockwell	Controle do Processo	Indústria	○ ●
Chilled Water System	Rockwell	Controle Distribuído	Indústria, Protótipo	○ ●
Cambridge packing cell	U. Cambridge	Controle da Manufatura	Laboratório	○
FABMAS	Technical University of Ilmenau	Controle da Manufatura	Indústria	○ ⊕ ●
PS-Bikes	Universita de Genova	Controle da Manufatura	Indústria, Protótipo	⊕
Axion-Holding	Magenta	Escalonamento da Produção	Indústria	○
Shop Modelarna Liaz	Certicon, Gerstner Laboratory	Planejamento da Produção	Indústria	○
SkodaAuto	Gedas, Certicon, Gerstner Laboratory	Planejamento da Produção	Indústria	○
Agent Steel System	Saarstahl AG, DFKI Gmbh	Planejamento da Produção	Indústria	⊕ ●
NASA airspace satellites	NASA	Gerenciamento das demandas do satélite	Indústria	○
Energia, Korolev Rocket and Space Corporation	Magenta	Re-escalonamento dinâmico	Indústria	○ ⊕
Aerogility	LostWax	Suporte inteligente na tomada de decisão	Modelagem e Simulação	○ ⊕

Tabela 4 – Referências de SMA na manufatura para o MAS4SSP – Parte II.

Turkey energy forecast	KKK Per. Bsk.	Previsão de demanda de energia	Simulação	○
California Energy Commission	AESC, Acronymics	Coordenação e Escalonamento	Indústria	○
Large urban area	Rockwell	Tratamento de Água	Simulação	○
SDM Laboratory	Yokogawa	Controle de Equipamento	Laboratório	●
NovaFlex	UNINOVA	Controle da Manufatura	Laboratório	○ ⊕ ●
ADACOR	Polytechnic Institute of Braganca	Controle da Manufatura	Laboratório	○ ⊕ ●
ABAS	Tampere University of Technology, Schneider Electric	Controle da Manufatura	Laboratório	○ ⊕ ●
OntoReA	TU Wien, Rockwell Automation, COPADATA	Controle da Manufatura	Laboratório	○
GRACE	UNIV. POLITECNICA delle MARCHE, SINTEF, Instituto Polit. de Bragança, AEA-Loccioni Group, Whirlpool Europe, SIEMENS AG	Controle da Manufatura	Indústria, Laboratório	○ ⊕ ●
COGMET	RWTH-Aachen, UFSC	Controle da Manufatura	Indústria, Laboratório	○ ⊕ ●

Legenda: conceitos ○ modelo/arquitetura ⊕ técnicas /tecnologias ●

A adoção da tecnologia de agentes na indústria é promovida por várias iniciativas, sobretudo por aquelas desenvolvidas por alguns comitês técnicos internacionais e por projetos de Pesquisa e Desenvolvimento (P&D). Os esforços de pesquisa, desenvolvimento e implantação de SMA iniciaram no passado com diversos projetos promovidos pela UE e concluídos com sucesso. Sucesso pelo menos do ponto de vista de contribuir para a evolução e maturidade da tecnologia

de agentes no ambiente industrial, pois, a grande maioria dos projetos se restringiu a aplicação de agentes em um equipamento ou a protótipos acadêmicos/laboratório que não foram implantados em um ambiente industrial real. Um breve resumo de alguns projetos da UE pesquisados e que contribuíram de uma forma ou outra com este trabalho de pesquisa é apresentado na Tabela 5.

O panorama descrito mostra que os esforços promovidos pela UE para divulgar e implantar a tecnologia de agentes não são poucos ou esporádicos. Em outras regiões do mundo, os esforços acontecem em diferentes formas de incentivo. Como exemplo, nos EUA o uso de soluções baseadas em agente é estimulada pela existência de parcerias e cooperação estreita entre a indústria e universidades[9]. No Japão não existe um fundo específico, ou programa prioritário para sistemas multiagente, mas existem programas P&D financiados pelo centro *Intelligent Manufacturing Systems* (IMS) japonês. Entre outros programas existentes, destaque ao Programa Internacional de Inovação em Negócios e P&D (*Intelligent Manufacturing System* - www.ims.org), que é uma iniciativa industrial para desenvolver uma nova geração de tecnologias de manufatura e processamento de dados da produção com a participação de empresas e instituições de pesquisa da UE, México, Coréia, Suíça e EUA. Este programa possui um forte papel na disseminação dos agentes industriais.

[9] Por exemplo, o projeto com a participação da NASA e IHMC (2003-2004) - *Design of a root cause detection system in hydrogen manufacturing domain*. E outros projetos principalmente na área de defesa.

Tabela 5 – SMA apoiados pela UE pesquisados.

Nome	Identificação	Breve descrição	Contribuição
GRACE	InteGration of pRocess and quAlity Control using multiagEnt technology (www.grace-project.org/)	Uso de sistema multiagente para a integração de processos e controle de qualidade, considerando aspectos auto-adaptativos em um sistema de controle de diagnóstico a nível local e global do processo. A validação dos resultados do projeto ocorrerá em uma linha de produção de máquina de lavar roupa.	○ ⊕ ●
IDEAS	Instantly Deployable Evolvable Assembly Systems (www.ideas-project.eu/)	Desenvolvimento de soluções tecnológicas que comprovem que o equipamento de montagem pode ser altamente adaptável, aplicando os conceitos do paradigma de *Evolvable Assembly Systems* (EAS). Foco baseado em tolerancia a falhas e controle de reconfiguração.	○ ⊕
COSMOS	COSt-driven adaptive factory based on MOdular Selfcontained factory units (EU ref. 246371)	Concepção, desenvolvimento e implementação de um sistema de controle para a gestão de fábrica com um sistema flexível de manufatura. Objetiva aumentar a produtividade da fábrica de montagem sem perder a flexibilidade. Concentra-se no processo de montagem de turbinas eólicas.	○ ⊕ ●
COLLIS.EUS	Soft Collaborative Intelligent Systems (FP7 ref. 255425)	Desenvolvimento de sistemas de informação colaborativos envolvendo múltiplos agentes interagindo com robôs e redes de sensores. Abrange uma vasta gama de aplicações, tais como fabricação, programação, controle, diagnóstico, logística, energia e gestão do tráfego rodoviário.	○
CONET	Cooperative Objects Network of Excellence (www.cooperatingobjects.eu)	Desenvolvimento de uma comunidade na área da Cooperação Objetos capazes de conduzir a pesquisa necessária para alcançar avanços na combinação de sistemas embarcados para robótica e controle, computação ubíqua e redes de sensores sem fio.	○
AgentLink III	A co-ordination network for agentbased computing (www.agentlink.org)	Atuar como uma comunidade unificadora das pesquisas baseadas em agentes em diferentes domínios. O objetivo de longo prazo é para colocar a Europa na vanguarda internacional nessa área.	○ ⊕ ●
I*PROMS	Network of Excellence for Innovative Production Machines and Systems (www.iproms.org/)	Abordagens de produção inovadoras, através do desenvolvimento de conceitos, ferramentas e técnicas que permitam a criação e operação de forma flexível, reconfigurável, tolerante a falhas e *user-friendly* dos sistemas de fabricação reagindo as demandas do ambiente internos (processo) e externo (consumidores) da produção.	○
PABADIS-PROMISE	PABADIS based product oriented manufacturing systems for reconfigurable enterprises (www.pabadis-promise.org/)	Desenvolvimento de uma nova metodologia de controle baseado em inteligência distribuída, uma nova ontologia para a manufatura, uma plataforma de multiagente de controle e uma nova geração de dispositivos RFID, dispositivos de controle de campo e ferramentas de ERP, permitindo a produção de forma rápida, flexível e eficiente.	○
MASCADA	Manufacturing Control Systems Capable of Managing Production Change and Disturbances (www.mech.kuleuven.be/mascada/)	Desenvolvimento de um sistema multiagente de controle baseado em manufatura capaz de gerir a mudança de produção e tratar distúrbios de forma eficaz e eficiente, maximizando a produção. Agentes autônomos e inteligentes foram usados para representar os componentes de fábrica.	○ ⊕ ●
RI-MACS	Radically Innovative Mechatronics and Advanced Control Systems (www.ri-macs.eu/)	Desenvolvimento de abordagens abertas que utilizam tecnologia multiagente, sensores sem fio e método de design (engenharia) virtual para tornar ágeis as plantas de fabricação e os equipamentos reconfiguráveis.	●
SOCRADES	Service-oriented cross-layer infrastructure for distributed smart embedded systems (www.socrades.eu)	Com o objetivo de desenvolver uma plataforma de projeto, execução e gestão para a próxima geração de sistemas da automação industrial, explorando o paradigma SOA. Utiliza um agente baseado em tomada de decisão para apoiar a dinâmica de uma plataforma de automação baseada em SOA.	⊕ ●

Legenda: conceitos ○ modelo/arquitetura ⊕ técnicas /tecnologias ●

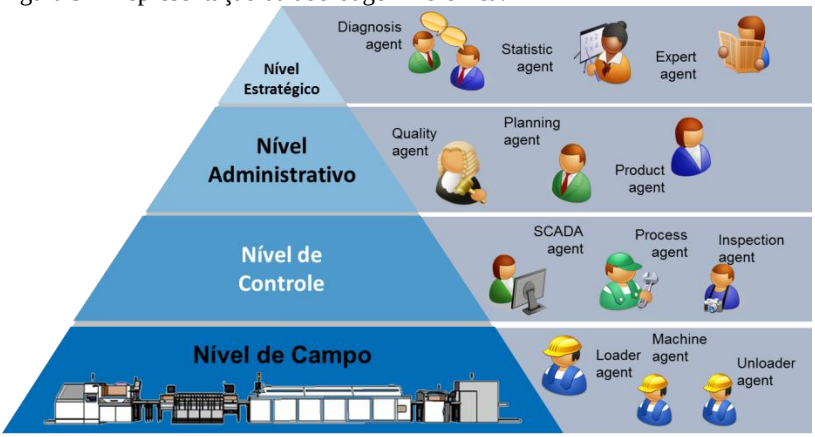
3.1.3. Sistemas Holônicos e Sistemas Multiagente

Dentre as iniciativas de aplicação dos agentes no ambiente industrial, a abordagem que emprega os Sistemas Holônicos em conjunto com os Sistemas Multiagente tem apresentado resultados importantes e por isso é uma abordagem com várias iniciativas (LEITÃO *et al.*, 2013b). Como esta abordagem possui similaridades com a abordagem a ser proposta, optou-se por fazer uma apresentação breve sobre sistemas holônicos e SMA na manufatura. Isto servirá que subsídio para o contraponto entre esta abordagem e a proposta neste trabalho de pesquisa.

O termo Sistema Holônico de Manufatura (*Holonic Manufacturing System - HMS*) foi criado em um trabalho de pesquisa desenvolvido no contexto do programa internacional Sistemas Inteligentes para a Manufatura (*Intelligent Manufacturing Systems - IMS*) (LEITÃO *et al.*, 2013b). O objetivo do trabalho de pesquisa foi incorporar os conceitos holônicos originalmente descritos por Arthur Koestler ao cenário da manufatura. A partir de observações dos organismos vivos e das organizações sociais Koestler criou o conceito de *holon* que busca descrever uma entidade independente de natureza híbrida que é simultaneamente parte e todo (LEITÃO *et al.*, 2013b *apud* KOESTLER, 1969). De fato, Koestler criou o conceito de holarquia que é uma hierarquia de *holon* auto-regulados que são simultaneamente conjuntos de entidades independentes do ponto de vista inferior da hierarquia e conjuntos de entidades dependentes do ponto de vista superior da hierarquia de controle. O *holon* possui um grau de autonomia que lhe permite tomar decisões sem pedir auxílio ou instruções para outros *holons* em níveis hierárquicos superiores; no entanto, cada *holon* de um nível inferior é objeto de controle dos *holons* dos níveis mais elevados. Este tipo de organização apresenta um comportamento estável e também a capacidade de tomar decisões caso distúrbios e situações imprevistas ocorram.

Os sistemas holônicos na manufatura aplicam os princípios holônicos no contexto da produção (LEITÃO *et al.*, 2013b). O controle da produção é distribuído entre os *holons*, que representam entidades da produção como: linhas de produção, máquinas, operários, materiais, encomendas, e assim por diante. Como já foi dito, os holons são capazes de trabalhar de forma autônoma e cooperativa quando subordinados a uma organização maior. Esta filosofia criou a metáfora de uma fábrica holônica que até o momento não foi totalmente implementada (Figura 9).

Figura 9 – Representação da abordagem holônica.



O objetivo de criar uma fábrica holônica resultou no desenvolvimento do padrão IEC 61499 baseado em blocos de função e na comunicação das entidades dos níveis mais baixos do controle do processo. Esta norma representa os principais resultados da iniciativa dos sistemas holônicos de manufatura e possui um grande potencial de aplicabilidade futura.

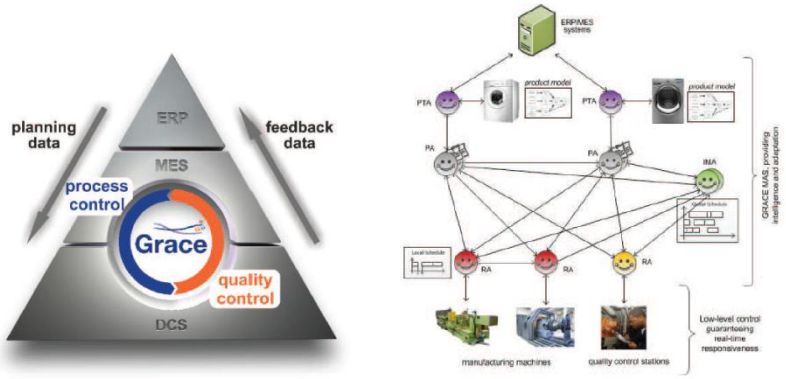
Diversas metodologias e arquiteturas holônicas para o domínio da produção foram propostas, o objetivo principal foi visando a formalização dos tipos de *holons*, suas responsabilidades e comportamentos e os cenários de interação. Um exemplo é PROSA (BRUSSEL *et al.*, 1998) que é uma arquitetura para o controle de recursos, produtos, ordens de produção e operários que foi desenvolvida baseado na arquitetura ADACOR (LEITÃO *et al.*, 2005), ou HCBA (*Holonic Component-Based Architecture*) que é uma arquitetura de componentes baseada na filosofia holônica (CHIRN e McFARLANE, 2000). Existem outros métodos holônicos que focam no domínio industrial, como AARIA (PARUNAK *et al.*, 2001), MetaMorph (MATURANA *et al.*, 1999) ou MASCADA (VALCKENAERS *et al.*, 1999).

Várias arquiteturas de controle de produção utilizando sistemas baseados em agentes e HMS têm sido propostas na literatura (LEITÃO *et al.*, 2013b). Uma destas arquiteturas que se destaca é a ADACOR (Arquitetura de Controle Adaptativo para Sistemas Holônicos de Produção Distribuídos) (LEITÃO *et al.*, 2005) que objetiva ser uma plataforma de resposta ágil quanto à alterações na linha de produção. O

foco do projeto ADACOR é aumentar a agilidade e a flexibilidade dos sistemas de controle da produção, especialmente aqueles localizados em ambientes de produção caracterizados por distúrbios e mudanças constantes.

A partir do projeto ADACOR outras iniciativas sugiram como: GRACE (*inteGration of pRocess and quAlity Control using multi-agEnt technology*) (Figura 10) e ARUM (*Adaptive Production Management*), que são projetos do programa da União Europeia (EU) *Factory of the Future* em atividade. Tanto o projeto GRACE, que possui o objetivo de fazer o controle do processo e de qualidade de uma linha de máquinas de lavar roupa, como o projeto ARUM (LEITÃO *et al.*, 2013), que foca na parte de planejamento e *start-up* da produção, se inspiraram na abordagem HMS e SMA da arquietura antecessora ADACOR.

Figura 10 – Visão geral do projeto GRACE.



Fonte: www.grace-online.org

3.1.4. Considerações sobre os Agentes nas Indústrias

Constata-se uma rica variedade de projetos de SMA para a manufatura, o que mostra o potencial de aplicação da tecnologia de agentes na indústria. Existe um entendimento comum entre os pesquisadores que vários requisitos da indústria moderna podem ser atendidos pelas características de um SMA ser distribuído, flexível e autônomo, fornecer um meio de comunicação comum e além disso os agentes podem atuar cooperativamente em sociedade.

Ainda assim, o número de sistemas multiagente implantados e que estão em operação em ambientes reais industriais é surpreendentemente pequeno. Pesquisas recentes (LEITÃO et al.,

2013b) concluem com a observação de que não existe avanço significativo ainda na transferência da tecnologia de agentes para a indústria.

Esta transferência relativamente lenta possui várias razões. Em primeiro lugar, a simples introdução de agentes – sem método, a princípio não reduz a complexidade dos problemas. Em segundo lugar, a aquisição de equipamentos para a interoperabilidade entre os sistemas é cara. Em alguns casos, somente a sobrecarga do meio de comunicação com a implantação de agentes pode degradar o desempenho do meio e consequentemente degradar o SMA. Em outros casos a ampliação do SMA pode ser difícil devido a abordagem selecionada para o desenvolvimento. Embora o paradigma de agente seja útil para a indústria e já existam iniciativas de metodologias e ferramentas para auxílio na engenharia destes sistemas, o apoio industrial ainda é pequeno. Outra dificuldade é a ausência de um sistema automatizado que proporcione a migração dos sistemas legados para o paradigma de agentes (PECHOUCEK *et al.*, 2005). Neste contexto, as aplicações hoje na indústria possuem alto risco e o custo de engenharia de software ainda é alto.

Outro desafio está relacionado ao comportamento do SMA, existe certa resistência a aceitação de soluções baseadas em agentes pela aparente ausência de determinismo do sistema. A indústria teme um comportamento imprevisível do SMA e cobra garantias dos desenvolvedores em relação ao desempenho e a confiabilidade. E isto ainda é uma lacuna a ser preenchida com o desenvolvimento de ferramentas de simulação e verificação formal de SMA.

Já as constatações sobre os SMA na manufatura do autor deste trabalho de pesquisa podem ser sumarizadas nos itens a seguir:

- é predominante o emprego do paradigma orientado a objetos para solucionar um problema orientado a multiagente;
- não é empregado um conceito de dimensões (multidimensional), ou quando empregado, foca-se em categorização das entidades agentes (ação/reação x cognitivo);
- grande esforço é empregado em customizações do tipo *ah-hoc* entre os sistemas legados e o sistema orientado a agentes;
- existe a ausência de um método para modelagem e implementação realmente orientado a multiagente;

- não existe um ambiente integrado de desenvolvimento realmente orientado a multiagente.

Neste contexto, propor uma nova abordagem para a implementação de sistema multiagente para a configuração e o monitoramento da produção de pequenas séries baseado em uma Arquitetura de Referência integrada que possa gerar um Modelo Genérico de Modelagem e Implementação se torna uma contribuição interessante.

O próximo capítulo apresenta uma nova abordagem para a modelagem e a implementação de SMA para a manufatura, em especial, para a configuração e o monitoramento da PPS. Esta abordagem é baseada em um Modelo de Referência que serviu de referência para a proposição de uma Arquitetura de Referência que por sua vez possibilitou a criação de um modelo genérico de modelagem e implementação, inspirados pelo aprofundamento na temática.

4. PROPOSTA DE SMA PARA PPS

Conforme apresentado no primeiro capítulo, o objetivo deste trabalho de pesquisa é *propor uma nova abordagem para a implementação de sistema multiagente para a configuração e o monitoramento da produção de pequenas séries*. Esta seção descreve esta proposta de abordagem apresentando a arquitetura elaborada e cada um dos elementos que a compõe baseados na estratégia apresentada na Figura 2.

Como ponto de partida para a concepção de uma Arquitetura de Referência para o SMA para configuração e monitoramento da PPS foi definido que a arquitetura deve ser baseada em padrões abertos e atuais, de modo a fornecer flexibilidade nas funcionalidades do SMA e interoperabilidade na comunicação entre os elementos envolvidos no auxilio à realização das tarefas de configuração e monitoramento da PPS.

Ainda, para que um SMA para PPS possa fornecer a configuração e o monitoramento da produção em um certo nível bastante genérico, é necessário que tal SMA possa comportar a mais variada gama de tipos de linha de produção. Para isso, a abordagem trata na dimensão do ambiente de fornecer um tipo padrão de recurso (artefato) que pode ser customizado para qualquer tipo de equipamento presente na linha de produção. A partir deste artefato genérico, os artefatos específicos são instanciados como especializações do artefato genérico.

A interação entre o ambiente do SMA com o ambiente da linha de produção é realizada por meio de um Sistema de Supervisão e Aquisição de Dados (SCADA) que configura e monitora as variáveis desejadas do processo (*tags*). Desta forma, o SCADA padroniza a comunicação com o SMA e para comunicação com o processo ele utiliza *drivers* de comunicação específicos para os equipamentos. O SCADA é um sistema normalmente presente nas linhas de produção e a maioria dos equipamentos fornecem algum mecanismo de comunicação (aberto ou proprietário) que permite interconectá-lo ao SCADA.

Outra vantagem do emprego de SMA com a abordagem baseada no Paradigma VOGAL e uma camada de aquisição de dados utilizando SCADA é a distribuição de processamento. O SMA para PPS não necessita se situar obrigatoriamente no mesmo dispositivo computacional. Dessa forma, diversas operações podem ser efetuadas em paralelo, em computadores separados, liberando a carga de processamento de um núcleo central.

É importante frisar que seguir uma Arquitetura de Referência traz certas vantagens que são de grande valor para as empresas de hoje, que possuem processos distribuídos, heterogêneos, flexíveis, etc. Como vantagens encontradas nesta abordagem, pode se citar: a interoperabilidade, a flexibilidade das funcionalidades (entrada e saída de agentes, mudança nas regras organizacionais, alterações na linha de produção), a independência quanto ao tipo do equipamento (SCADA padroniza o acesso aos recursos) e quanto as funcionalidades, e a possibilidade de que qualquer linha PPS possa ter seu SMA para configuração e monitoramento da PPS.

Antes de apresentar a Arquitetura de Referência para SMA para configuração e monitoramento da PPS, é necessário que seja apresentado o Modelo de Referência que serve de referência seguido dos Requisitos Desejáveis. Após apresentar a Arquitetura de Referência são apresentados os Métodos e as Ferramentas que propõem o Modelo Genérico de Modelagem e Implementação do SMA para configuração e monitoramento da PPS.

4.1. MODELO DE REFERÊNCIA

Um Modelo de Referência é caracterizado por ser uma abstração da realidade, que pode ser expressa por um formalismo de um método de modelagem, conforme os objetivos de um usuário (VERNADAT, 1996).

> Um Modelo de Referência é um *framework* abstrato para entendimento dos relacionamentos significantes entre as entidades de algum ambiente. Ele habilita o desenvolvimento de arquiteturas específicas usando padrões consistentes ou especificações suportando aquele ambiente. Um Modelo de Referência consiste de um conjunto mínimo de conceitos unificados, axiomas e relacionamentos com um domínio de um problema particular, e é independente de padrões específicos, tecnologias, implementações, ou outro detalhe concreto (MACKENZIE et al., 2006, 2009).

Para Bass *et al.* (2003), um Modelo de Referência é uma divisão de funcionalidades, juntamente com o fluxo de dados entre as partes. Ele se caracteriza como um padrão de decomposição do problema. Os

modelos de referência possuem características de domínios maduros, decorrente da experiência sobre este domínio.

Na próxima seção é apresentado o Modelo de Referência para SMA baseado no Paradigma VOGAL. Este modelo integra de forma sistêmica e sinérgica as dimensões de Agente, Ambiente e Organização.

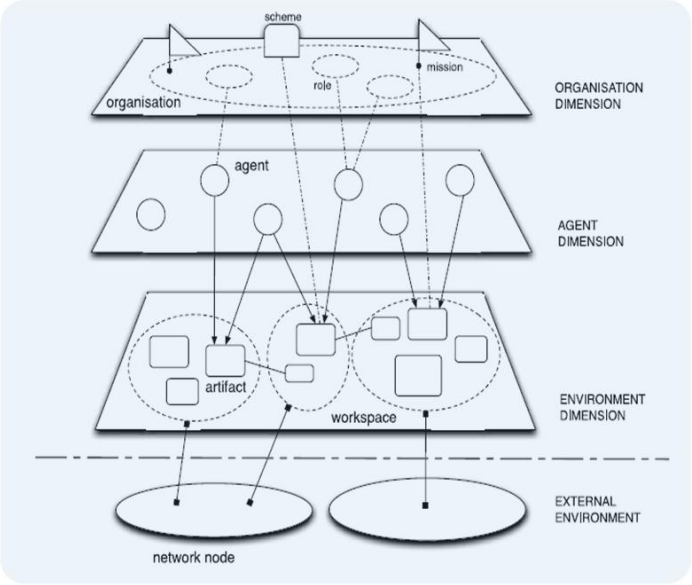
4.1.1. Modelo de Referência JaCaMo

Demazeau (1995) defende que a integração das dimensões do Paradigma VOGAL no SMA deve ser mais do que simplesmente unir um conjunto de abstrações. Por um lado, precisa-se adequar a semântica dos conceitos e métodos de programação, por outro lado, o diferencial é que se permita o emprego dos conceitos de forma sinérgica melhorando a expressividade global da abordagem de programação do SMA respeitando os conceitos de cada dimensão. A plataforma JaCaMo, tem sido desenvolvida para suportar esta ideia. JaCaMo é tida como a única plataforma prática para programação orientada a multiagente que integra de maneira sinérgica dimensões do Paradigma VOGAL (Agente, Ambiente e Organização). Outras abordagens existentes consideram apenas as dimensões Agente & Organização (A&O) ou as dimensões Agente & Ambiente (A&E) e a sua integração é realizada por mecanismos estáticos e normalmente não são nada amigáveis do ponto de vista de programação (*ad hoc way*).

JaCaMo integra em uma perspectiva unificadora agentes, organizações e ambientes (Figura 11). Para detalhes históricos da proposta consulte (HÜBNER *et al.*, 2010), que introduz a ideia de projetar uma plataforma de gestão organizacional em termos de ambientes baseados em artefatos e (HÜBNER *et al.*, 2010b) que se concentra particularmente na introdução da noção de integração do Ambiente&Organização (E&O). Mais recentemente, a pesquisa de Zatelli (2013) integra a dimensão – Interação – no metamodelo JaCaMo.

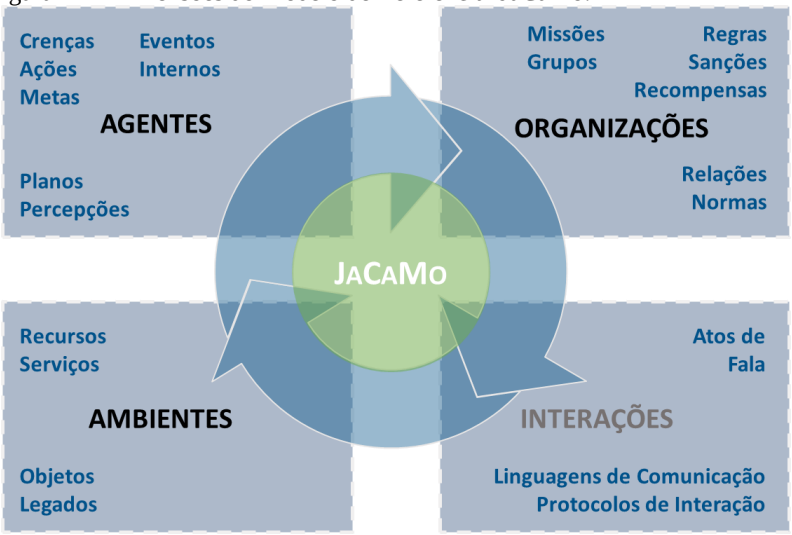
JaCaMo integra três plataformas (de três dimensões diferentes) e define uma ligação semântica entre os conceitos das dimensões – Agente, Ambiente e Organização (Figura 12) – por meio do seu metamodelo e do ambiente de programação. O objetivo é obter um ambiente de programação que seja uniforme e consistente no sentido de simplificar a combinação das dimensões para a programação de SMA.

Figura 11 – Visão geral do sistema JaCaMo contemplando as suas dimensões.



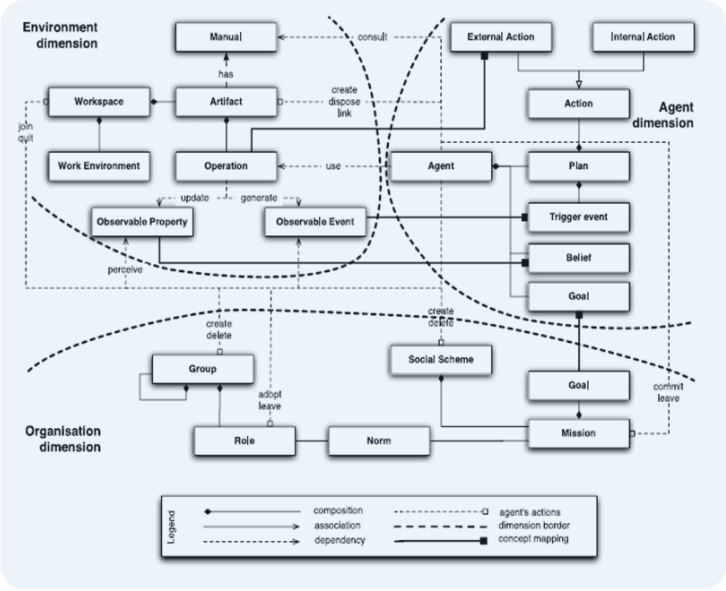
Fonte: Boissier *et al.* (2011)

Figura 12 – Dimensões do Modelo de Referência JaCaMo.



A Figura 13 apresenta o metamodelo integrado de programação JaCaMo. As abstrações relacionadas a cada uma das dimensões estão envoltas em linhas tracejadas na Figura 13. Sendo um metamodelo integrado de programação – isto é, um metamodelo de programação focado em abstrações e construções – que não inclui conceitos ou abstrações que não fazem parte das estruturas ou linguagens de programação pesquisadas. Por exemplo, o conceito de intenção está presente no *runtime* de *Jason*, mas não na linguagem de programação *Jason*, então o conceito de intenção não está incluído no metamodelo. Além de apresentar o conceito principal da programação JaCaMo, outro objetivo deste metamodelo é apresentar explicitamente as dependências, conexões e mapeamentos conceituais entre as abstrações pertencentes as diferentes dimensões.

Figura 13 – Metamodelo JaCaMo. Cardinalidades não representadas.



Fonte: Boissier *et al.* (2011)

As abstrações que pertencem à dimensão do agente – *Agent dimension* – (relacionadas com o metamodelo *Jason*) são principalmente inspiradas na arquitetura BDI na qual *Jason* é derivado (baseado na linguagem *AgentSpeak*). Assim, um agente é uma entidade constituída

por um conjunto de crenças que representam o estado atual do agente e do conhecimento sobre o ambiente no qual está situado; além disso, um conjunto de metas que correspondem às funções que o agente deve desempenhar e um conjunto de planos que são ações (internas ou externas) desencadeadas por eventos que os agentes podem dinamicamente conceber, iniciar e executar para alcançar suas metas. Os eventos podem estar relacionados a mudanças no ambiente ou a alteração da base de crenças do agente ou mudanças nas suas metas.

O ambiente – *Environment dimension* – (baseado no metamodelo A&A) é composto por um ou mais *workspaces* que são utilizados para definir uma topologia para o ambiente. Cada *workspace* é um espaço lógico que contém um conjunto dinâmico de artefatos que **são os recursos computacionais básicos que definem a estrutura do ambiente e seu comportamento**. Os agentes podem criar, descobrir, perceber e utilizar estes recursos em tempo de execução para cumprir suas metas. Cada artefato fornece um conjunto de operações e propriedades observáveis que definem a interface do artefato. Esta interface é utilizada pelos agentes para observar e realizar operações com/nos artefatos. A execução de uma operação pode gerar atualizações nas propriedades observáveis e provocar eventos do artefato. O último tipo de entidade presente na dimensão ambiente é o manual, uma entidade utilizada para apresentar uma descrição das funcionalidades fornecidas pelo artefato.

Do lado organizacional – *Organization dimension*, o *framework* Moise+ descreve uma organização: (i) a partir de um ponto de vista estrutural, em termos de grupos e de papéis; (ii) a partir de um ponto de vista funcional, em termos de esquema social, missão e metas - o esquema social decompõe a estrutura das metas da organização em sub-metas, que são agrupadas em missões; (iii) a partir de um ponto de vista normativo, em termos de normas que vinculam os papéis as missões, restringindo o comportamento do agente através de um conjunto de metas que terá de seguir quando optar por entrar em um grupo e desempenhar um certo papel neste grupo.

Um dos objetivos principais da integração das dimensões descritas acima é simplificar o modelo de programação no desenvolvimento de SMA. No nível da plataforma (*Platform level*) cada nó integra as plataformas de *Jason*, CArtAgO e Moise+, bem como a tecnologia de interface necessária, tudo isto sendo executado em uma máquina virtual Java que por si só torna o acesso transparente para todos os recursos do sistema operacional.

O *framework* JaCaMo ainda carece de aplicações em diversas áreas, o principal domínio de suas aplicações atualmente é na área da computação. O que se pretende aqui é contribuir para preencher esta lacuna com a proposta de uma abordagem de desenvolvimento de SMA para o configuração e monitoramento da PPS baseada no *framework* JaCaMo.

Um sistema multiagente JaCaMo (isto é, um software programado em JaCaMo) é constituído por uma organização de agentes programada em Moise+, agentes autônomos programados em *Jason*, que trabalham compartilhando recursos do ambiente (*artefatos*) programados em CArtAgO.

4.1.1.1. Dimensão do Agente: *Jason*

Jason é uma plataforma para o desenvolvimento de sistemas multiagente, que incorpora uma linguagem de programação orientada a agentes. A linguagem é baseada na arquitetura BDI inspirada na linguagem *AgentSpeak* (OMICINI *et al.*, 2004). A linguagem *Jason* foi posteriormente expandida como resultado de uma série de publicações por Bordini, Hübner e colegas (BOISSIER *et al.*, 2006, 2011; BORDINI *et al.*, 2007; HÜBNER *et al.*, 2010, 2010b). Estes trabalhos adequaram *Jason* como uma linguagem prática de programação de agentes. Estas extensões também criaram uma linguagem derivada de *AgentSpeak* que é então conhecida como *Jason* (STRATULAT *et al.*, 2009).

4.1.1.2. Dimensão do Ambiente: CArtAgO

CArtAgO (RICCI *et al.*, 2007b) é um *framework* para a programação de Ambientes em sistemas multiagente. A ideia é que o ambiente possa ser utilizado como uma abstração de alto nível fornecendo recursos para a concepção e operação do SMA. CArtAgO cria uma camada que encapsula funcionalidades e serviços que os agentes podem explorar durante a sua execução. O metamodelo A&A (RICCI *et al.*, 2007b) é a base conceitual do *framework* CArtAgO. Desta forma, o ambiente é projetado e programado como um conjunto dinâmico de entidades computacionais chamadas artefatos que fazem parte dos *workspaces* que podem estar distribuídos em uma rede.

4.1.1.3. Dimensão da Organização: Moise+

O *framework* Moise+ (BOISSIER *et al.*, 2011) implementa um modelo de programação para a dimensão organizacional. Esta abordagem inclui uma linguagem de modelagem de organização, uma plataforma de gestão da organização (HÜBNER *et al.*, 2010) e suporte para mecanismos de raciocínio organizacional baseado em agente (BOISSIER *et al.*, 2011).

4.2. REQUISITOS DESEJÁVEIS

Esta seção apresenta os Requisitos Desejáveis para o SMA para configuração e monitoramento da PPS. Requisitos estes considerados genéricos o suficiente para aportar uma abordagem que garanta a configuração e o monitoramento da PPS. É importante destacar que as características do SMA se enquadram bem para atender aos requisitos apontados por SHEN *et al.* (2006) (Tabela 6) e aos Requisitos Desejáveis levantados neste trabalho de pesquisa. Nesta seção o SMA é apresentado como uma alternativa de estilo arquitetural[10] viável tecnicamente para a PPS.

De modo a manter maior conformidade com os conceitos apresentados sobre a PPS no Capítulo 2, os requisitos a seguir são apresentados como forma de justificar o emprego de Sistema Mutiagente (SMA) como integrante do estilo arquitetural da abordagem proposta para atender ao objetivo de configurar e monitorar a PPS.

- A abordagem deve atender as características de lote reduzido, diversidade de produtos e vários *setups* da linha de PPS.
- A abordagem proposta deve ser baseada em padrões abertos e atuais que possibilitem atender a qualquer linha de PPS.
- A abordagem deve permitir que novos objetivos de produção possam ser definidos em tempo de execução

[10] O estilo arquitetural descreve os tipos de elementos e suas relações, juntamente com um conjunto de restrições sobre como eles podem ser utilizados, além de padrões de interação entre os elementos. Tais restrições, sobre a arquitetura e sobre o sistema em si, são vistas na forma de uma imagem da utilização do sistema como um todo. Por exemplo, o estilo arquitetural cliente-servidor utiliza dois tipos de elementos, o cliente e o servidor, e sua coordenação é descrita em termos de protocolos de comunicação entre eles (BASS *et al.*, 2003).

sem que se tenha que obrigatoriamente alterar o código fonte.

- Deve ser possível modelar qualquer linha de PPS derivando os recursos do ambiente de um recurso genérico geral.
- A modelagem e a implementação do SMA para configuração e monitoramento da PPS deve ser um processo unificado e sinérgico sem a necessidade de customizações do tipo *ad hoc*.
- Deve haver a possibilidade do SMA não estar necessariamente localizado no mesmo computador e sim distribuído em vários nós de uma rede.
- As tecnologias utilizadas devem privilegiar o emprego de todo o potencial da programação orientada a multiagente e da especificidade de cada uma das dimensões propostas pelo paradigma VOGAL.

Tabela 6 – Requisitos dos sistemas de manufatura inteligente.

Requisitos dos sistemas de manufatura inteligente (SHEN et al., 2006)	
R1	Integração total de sistemas de software e hardware heterogêneos seja dentro da empresa, ou em uma empresa virtual, ou em uma cadeia de fornecimento;
R2	Uma arquitetura de sistemas aberta que possa acomodar novos sistemas (hardware e software) ou que possa substituir sistemas legados *on the fly* (em tempo de operação).
R3	Sistemas eficientes e eficazes de comunicação e cooperação entre os departamentos de uma empresa e entre empresas.
R4	Incorporação das capacidades cognitivas humanas nos sistemas de manufatura.
R5	Resposta rápida a ordens de mudança externas e a distúrbios inesperados provenientes do ambiente (interno ou externo) da produção.
R6	Tolerância a falhas, tanto ao nível do sistema e como a nível de subsistema de forma a detectar e recuperar o sistema de falhas e minimizar os seus impactos no ambiente de trabalho.

4.2.1. Atender a PPS

> "A abordagem deve atender as características de lotes reduzidos, diversidade de produtos e vários *setups* da PPS."

Nos Capítulos 1 e 2 a PPS foi conceituada como um sistema de produção que se concentra na fabricação de uma **grande variedade de produtos** em um **curto período de tempo** e com um **baixo volume de produção** (possivelmente unitário). Isto traz uma série de desafios para PPS que são: (i) a falta de previsibilidade; (ii) a constante criação/atualização da documentação de qualidade – como estratégias de inspeção; (iii) o aumento do tempo de *setup* devido as várias trocas de produto que o torna uma variável importante na viabilidade econômica da produção; (iv) a falta de informações sobre o produto a ser produzido – não há tempo e recursos para treinamento da linha; (v) o curto período de tempo para corrigir falhas na fabricação do produto; (vi) a dificuldade de realizar ações corretivas *online* e por fim (vii) a falta de dados para tomadas de decisão – aprovar/reprovar o produto/lote.

Diante deste cenário, o emprego de SMA neste ambiente de produção flexível apresenta pelo menos três fatores positivos que contribuem para atender as especificidades da PPS:

- a estrutura de um SMA permite uma fácil inserção de novos agentes (hardware e/ou software) no sistema atual, sem a necessidade de reprogramar toda a lógica de controle do sistema multiagente;
- permite implantar a solução em um ambiente distribuído, multiplataforma, com diferentes componentes de software e hardware e diferentes protocolos de comunicação. Os agentes tornam este sistema heterogêneo em um meio comum com o mesmo protocolo de comunicação;
- a cooperação ou competição entre os diferentes agentes do sistema traz uma autonomia de operação que é desejada do ponto de vista dos processos de fabricação, bem como dos sistemas de inspeção e controle da qualidade.

As potencialidades da abordagem de SMA se evidenciam quando o domínio de aplicação apresenta características como: é intrinsecamente distribuído; requer a união de diferentes domínios de conhecimento para a solução do problema; requer a aplicação de diferentes ‘resolvedores’ de problemas, integrados em um mesmo ambiente, e com variados protocolos de comunicação; inclui diferentes níveis de autonomia e descentralização de resultados e decisões; é dinâmico; é extremamente conflitante, em função das muitas restrições dinâmicas usualmente existentes, o que requer variados níveis de

cooperação e negociação a fim de que o processamento não seja interrompido.

Comparando as características do ambiente da PPS com as potencialidades dos SMA se percebe que empregar um SMA para configuração e monitoramento da PPS é justificável. Mas é importante que esta análise seja criteriosa, pois em alguns casos, o emprego de sistemas baseados em agentes pode ser desnecessário e ainda trazer uma grande desvantagem – o aumento da complexidade do problema a ser solucionado.

Neste contexto, foram definidos os seguintes sub-requisitos para que a abordagem proposta atenda ao requisito – Atender a PPS – seja cumprido:

- o SMA deve permitir a configuração automatizada da linha de produção;
- o SMA deve realizar o monitoramento automatizado do processo de produção e do produto;
- o SMA deve ser capaz de realizar ações corretivas autônomas em caso de falha;
- o planejamento da produção deve ser realizado baseado em similaridades entre os produtos.

4.2.2. Padrões Abertos e Atuais

> "A abordagem proposta deve ser baseada em padrões abertos e atuais que possibilitem atender a qualquer linha de PPS"

Uma das questões mais críticas na manufatura é a integração entre os elementos envolvidos na produção. Para isso, um requisito básico de integração da manufatura é que devem existir mecanismos adequados e interfaces abertas para que seja possível esta integração (GESSER, 2006). Porém, uma das grandes barreiras à interoperabilidade entre os sistemas é que muitos dos sistemas industriais são legados e desenvolvidos pelos fabricantes. Estes sistemas atuam no processo de forma específica e de maneira independente, contribuindo para a dificuldade de integração com outros sistemas. Para que esses sistemas sejam capazes de interoperar de maneira transparente e automatizada, customizações do tipo *ah hoc* (estáticas e inflexíveis) podem ser necessárias para que as partes envolvidas sejam capazes de interoperar.

Nesse sentido, a utilização de tecnologias padronizadas permite que qualquer desenvolvedor que siga determinados padrões possa

utilizá-los e interagir com os demais desenvolvedores que aderem aos mesmos padrões (GESSER, 2006). Neste contexto, tecnologias baseadas em agentes apresentam uma solução bastante atraente para a integração entre os diversos elementos envolvidos nos processos.

Aliado a utilização de padrões a opção por soluções de código aberto (*open source*) garante que a implementação da abordagem não fique sujeita a fabricantes que possuem recursos disponíveis para investir no pagamento de licenciamentos ou *royalties*. Outra vantagem da tecnologia aberta é a existência de comunidades de desenvolvimento formadas por inúmeros usuários e desenvolvedores que auxiliam na implementação e manutenção de sistemas baseados nesta tecnologia.

Quanto a atualidade, optou-se por padrões atuais que fazem uso de métodos e técnicas difundidas na comunidade, como é o caso de *web service*[11], um mecanismo de comunicação difundido em diversas áreas nos setores educacional, serviços e indústria.

4.2.3. Definição de objetivos dinâmica

> "A abordagem deve permitir que novos objetivos de produção possam ser definidos em tempo de execução sem que se tenha que obrigatoriamente alterar o código fonte."

Dentre a série de desafios da PPS, o maior desafio é garantir o nível de qualidade exigido já na produção da primeira unidade do lote (que pode ser constituído por uma única unidade) ou seja, *first time right on time*. A meta é que os produtos manufaturados na PPS possuam o mesmo nível de qualidade desde o início que os modelos de produção em massa. Para solucionar este desafio é importante que o sistema de produção permita que objetivos de produção possam ser alterados durante a produção do lote. A presença de uma falha em uma unidade pode requerer que um (ou mais) objetivo da produção seja alterado – como o *setup* de um determinado equipamento – garantindo que as próximas unidades sejam produzidas dentro da normalidade.

O *framework* Moise+, que integra o Modelo de Referência JaCaMo, incorpora um interpretador que fornece à plataforma

[11] *Web service* é uma solução utilizada na integração de sistemas e na comunicação entre aplicações diferentes. Com esta tecnologia é possível que novas aplicações possam interagir com aquelas que já existem e que sistemas desenvolvidos em plataformas diferentes sejam compatíveis. Os *Web services* são componentes que permitem às aplicações enviar e receber dados em formato XML. Cada aplicação pode ter a sua própria linguagem, que é traduzida para uma linguagem universal, o formato XML. Fonte: http://pt.wikipedia.org/wiki/Web_service

organizacional as informações necessárias para a gestão da organização, inclusive em tempo de execução (*runtime*). Com Moise+ é possível definir qual é o objetivo global do SMA, quem serão os agentes comprometidos com este objetivo global e como eles atuarão para atingir tal objetivo. Se o objetivo for alterado, os agentes envolvidos também alterarão seus objetivos individuais para atender ao novo objetivo global.

4.2.4. Modelar qualquer linha PPS

> "Deve ser possível modelar qualquer linha de PPS derivando os recursos do ambiente de um recurso genérico geral."

Modelar qualquer linha de PPS é um desafio para a proposta e para se conseguir isto, se valeu das potencialidades do *framework* da dimensão de Ambiente (CArtAgO) do Modelo de Referência e da criação de uma camada de aquisição de dados padronizada dos equipamentos da linha com o uso de um sistema SCADA.

O Modelo de Referência JaCaMo contribui para atender este requisito através da dimensão do Ambiente. O modelo A&A, do *framework* CArtAgO, cria o chamado artefato, que são recursos **não-autônomos** para a concepção e operação do ambiente do SMA. A dimensão do ambiente contribui positivamente em dois casos:

- simplificação do SMA: em muitas abordagens baseadas em agentes na manufatura que não consideram a dimensão do Ambiente, os equipamentos que compõem a linha de produção são modelados como agentes. Isto causa um aumento significativo na quantidade de agentes presentes no sistema. E em muitos casos, alguns equipamentos são modelados como agentes, mas muitas vezes eles são meros produtores ou receptores de dados sem autonomia o que não o caracteriza como agente.
- generalização do ambiente: a existência de uma dimensão do Ambiente garante flexibilidade na modelagem e implementação do ambiente. Na abordagem proposta, modela-se um artefato genérico geral que a partir dele os demais equipamentos da linha são derivados. Para se conseguir isto, a modelagem do ambiente segue o paradigma orientado a objetos que a

> partir de um objeto genérico geral – artefato *Machine* – todos os demais são especializações do objeto geral.

Já o SCADA soluciona o desafio da comunicação com os equipamentos na linha de produção. O SCADA atua como um sistema de aquisição de dados (sensores e atuadores). A inserção da camada do SCADA na abordagem criou um meio de comunicação único e padronizado com diferentes equipamentos que compõem uma linha de produção. Atualmente, os equipamentos possuem pelo menos um protocolo de comunicação que permite sua integração com algum sistema SCADA. Desta forma, o SCADA realiza a comunicação com os equipamentos e o SCADA envia e recebe dados para o SMA. Para o SMA o SCADA torna o ambiente da linha de produção homogêneo.

4.2.5. Processo unificado de modelagem e implementação

> "A modelagem e a implementação do SMA para configuração e monitoramento da PPS deve ser um processo unificado e sinérgico sem a necessidade de customizações do tipo *ad hoc*."

Como já foi exposto quando se discutiu a padronização, no ambiente da manufatura dominam as soluções legadas de cada fabricante, isto dificulta a interoperabilidade e intercambialidade de equipamentos. E a solução normalmenta utilizada quando se necessita a integração é a construção de mecanismos estáticos e inflexíveis (*ad hoc way*) que permitam a integração entre equipamentos *i* e *j*. A abordagem proposta busca conviver com esta diversidade de soluções técnicas, mas não com o emprego de mecanismos *ad hoc*. O *framework* JaCaMo contribui com a integração sinérgica do desenvolvimento de um SMA baseado no paradigma VOGAL. Até o momento, JaCaMo se apresenta como única plataforma operacional que integra as dimensões Agente, Ambiente e Organização do paradigma VOGAL.

A implementação da comunicação com os dispositivos, como citado anteriormente, emprega um sistema SCADA como mecanismo de padronização do ambiente heterogêneo dos equipamentos da linha de produção em um ambiente homogêneo de comunicação com o SMA utilizando *Web Service* como interface de comunicação entre o SMA e o SCADA.

A interface com o usuário pode ser construída com o emprego do mesmo mecanismo de comunicação, *Web Service*, para integração com sistemas legados, ou uma interface em Java pode ser desenvolvida

dentro da plataforma JaCaMo – utilizando o *framework* CArtAgO para criação dos recursos da interface –, ou uma interface em outra linguagem de programação – utilizando para isto mecanismos de interfaceamento como JNI[12] para Java e C++, ou utilizar um sistema SCADA como o proposto para integração com a linha de produção via *Web Service*. Enfim, a interface com o usuário é uma decisão de implementação que depende do cenário no qual o SMA para configuração e monitoramento da PPS é aplicado. Fica a critério da equipe de desenvolvimento qual opção escolher.

4.2.6. Distribuído

> "Deve haver a possibilidade do SMA não estar necessariamente localizado no mesmo computador e sim distribuído em vários nós de uma rede."

A distribuição de um sistema pode ser tanto de hardware como de software, sendo que tais elementos podem estar localizados em redes de computadores e se comunicam, coordenando suas ações por passagem de mensagens. Algumas características que os sistemas distribuídos fornecem são a concorrência de componentes, independência de falhas de componentes e transparência de localização. Os usuários destes sistemas o percebem como um sistema único com funcionalidades integradas.

Dentre as motivações para a utilização de sistemas distribuídos estão o compartilhamento de recursos, gerenciados por servidores e acessados por clientes (COULOURIS *et al.*, 2005).

Outras características que são encontradas nestes sistemas são a concorrência e heterogeneidade de seus componentes. Isso permite que vários componentes, inclusive desenvolvidos em diferentes linguagens e tecnologias, possam ser executados ao mesmo tempo. Estes componentes também podem ser adicionados ou substituídos de forma dinâmica. Esta dinamicidade também traz a vantagem da tolerância a falhas, sendo que com a indisponibilidade de um elemento do sistema, outro pode assumir sua função (COULOURIS *et al.*, 2005).

[12] *Java Native Interface* (JNI) é um padrão de programação Java que permite utilizar código C/C++ (ou qualquer outro código nativo) em um aplicativo Java. Mas, existe uma desvantagem importante, quando se utiliza JNI o aplicativo Java deixa de ser multiplataforma para ser dependente da plataforma utilizada para programar a sua parte em C/C++.

Ainda, os sistemas distribuídos são paralelos e vários usuários podem invocar serviços do sistema simultaneamente, assim como vários processos servidores são executados concorrentemente, sendo que cada um atendendo várias requisições dos clientes (COULOURIS *et al.*, 2005).

Uma aplicação baseada em agentes é intrinsecamente distribuída. No *framework* JaCaMo não é diferente, no nível de Execução (*Execution level*), uma aplicação JaCaMo é representada por uma organização composta por um ou múltiplos *workspaces*, rodando em nós JaCaMo. Os agentes em execução em um nó JaCaMo podem se unir e trabalhar simultaneamente em vários *workspaces*, incluindo os *workspaces* remotos (ou seja, *workspaces* não hospedados pelo mesmo nó em que os agentes estão executando). Desta forma, uma aplicação JaCaMo pode possuir múltiplos *workspaces* hospedados em vários equipamentos distribuídos pela linha produção ou até mesmo fora dela.

4.2.7. Todo o potencial da programação orientada multiagente

> "As tecnologias utilizadas devem privilegiar o emprego de todo o potencial da programação orientada a multiagente e da especificidade de cada uma das dimensões propostas pelo paradigma VOGAL."

A abordagem denominada *Multi-Agent Oriented Programming* (MAOP) (BOISSIER *et al.*, 2011), ou Programação Orientada a Multiagente, enfatiza a integração de um conjunto de abstrações, com origens diferentes, mas com dimensões correlacionadas ao paradigma VOGAL, a saber: organização (O), agente (A), interação (I) e ambiente (E). Esta abordagem é construída de tal forma que englobe todos os aspectos que se acredita serem fundamentais para o desenvolvimento do Sistema Multiagente.

Comparando esta abordagem com a programação orientada a agentes originalmente apresentada por Shoham (1993), o objetivo aqui é: *possuir abstrações de alto nível de programação também para a dimensão de organização, de interação e do ambiente, não somente ao nível de agente*. E estas dimensões necessitam estar integradas com as abstrações de alto nível. Na dimensão da organização, estão presentes abstrações que remetem aos seguintes conceitos: grupo, papel, norma e esquema social (algo que se refere a um tipo de plano social, ou plano

global, ou a delegação de tarefas e outros vários termos presentes na literatura sobre agentes). Na dimensão do agente, existem as seguintes abstrações dos conceitos de: crença, meta, plano, ação e percepção. Na dimensão de interação não existem abstrações, o foco desta dimensão é a ‘conversa’ entre os agentes, a qual é realizada por meio de mensagens seguindo um protocolo de comunicação (modelado por uma ontologia). Além disso, existe uma coordenação (normalmente um ou mais agentes) que administra as mensagens entre os agentes e dos agentes com as demais dimensões. Na dimensão de ambiente estão presentes as funcionalidades de executar ações e capturar as percepções. A principal abstração deste nível, o artefato, pode ser utilizado para representar recursos, ferramentas, serviços e uma variedade de outros componentes não-autônomos de um SMA.

As pesquisas neste paradigma possuem um amplo campo de atuação indo da psicologia até a engenharia. Uma desvantagem desta abordagem é a complexidade no processo de desenvolvimento passando pela modelagem de agentes, dos artefatos, de diagramas normativos, funcionais, esquemas sociais e de interação, etc. Esta complexidade também é resultante da abrangência do problema a ser estudado. Normalmente os projetos de SMA atuam em ambientes naturalmente complexos, onde os outros paradigmas não atuam ou o esforço do desenvolvimento de um solução seria imenso.

4.3. ARQUITETURA DE REFERÊNCIA

Tendo como premissa inicial o Modelo de Referência e os Requisitos Desejáveis, uma Arquitetura de Referência tem por objetivo fornecer um rumo para o mapeamento das funcionalidades do Modelo de Referência para um modelo que deve se transformar em um sistema, ou seja, a Arquitetura de Referência é o mapeamento do Modelo de Referência para elementos de software (BASS *et al.*, 2003).

A Arquitetura de Referência é um padrão genérico para um projeto e que deve abordar requisitos para o desenvolvimento de soluções, guiadas pelo Modelo de Referência, de forma a atender as necessidades do projeto (Requisitos Desejáveis) formando um caminho a ser seguido (HOFMEISTER *et al.*, 2000).

O Modelo de Referência JaCaMo, resolve bem o desenvolvimento de SMA na área de sistemas de informação e comunicação. Contudo, o ambiente de produção é tipicamente composto por diversos modelos e fabricantes de equipamentos e normalmente são dotados de sistemas proprietários que dificultam a integração. Assim,

um desafio desta abordagem foi: *tornar o ambiente da produção tipicamente heterogêneo em um meio homogêneo do ponto de vista do SMA*. A estratégia para isto é apresentada na Figura 14 com o uso de camadas de software especializadas em tratar de cada etapa do processo de configuração e monitoramento da PPS.

A Figura 15 apresenta uma visão geral da abordagem MAS4SSP (*Muti-Agent System for Small Series Production*). A dimensão do Usuário – topo – pode possuir diversos níveis de conhecimento. O usuário pode tanto ser uma pessoa que opera um SCADA, como também pode ser um usuário avançado, que desenvolve sistemas para configuração e monitoramento da PPS. Nesta dimensão os sistemas podem ser sistemas de gestão da empresa (ERP, SCM, PCP, MES, etc.) ou sistemas específicos desenvolvidos em linguagens de programação próprias (como Java, C++, C#, Python, etc.). A comunicação desta dimensão com as dimensões do SMA pode ser feita com a tecnologia *Web Service*, ou tecnologias específicas como JNI (integração de código Java e C/C++), ou utilizando também JAVA como recurso da dimensão de Ambiente do SMA para desenvolvimento de uma interface com o usuário.

Após a dimensão de INTERFACE com o Usuário (que pode ser outro sistema) estão as dimensões do SMA com a dimensão organizacional – com suas metas organizacionais, seguida da dimensão dos agentes – onde as entidades autônomas, agentes, para configurar e monitorar a produção estão posicionados, e a dimensão do ambiente – com os recursos não autonômos que são a representação da linha produção com seus equipamentos dotados de sensores/atuadores. A última dimensão da Arquitetura de Referência é a linha de produção que é o ambiente exterior onde o sistema supervisório atua para a aquisição dos dados que serão utilizados nas dimensões superiores e também atua no sistema atualizando dados do estado dos equipamentos. O interfaceamento do SCADA com os equipamentos é realizado por meio de *drivers* específicos para cada equipamento, já para a comunicação como o SMA a tecnologia de *Web Service* padroniza a forma de comunicação. O uso do ScadaBR (e qualquer outro SCADA que possua uma API *Web Service*) na arquitetura possibilitou a transformação de um ambiente exterior antes heterogêneo do ponto de vista do SMA para um ambiente homogêneo padronizando a comunicação com a linha de produção através da tecnologia *Web Service* com o ScadaBR.

Figura 14 – Estratégia para o desafio do ambiente da PPS.

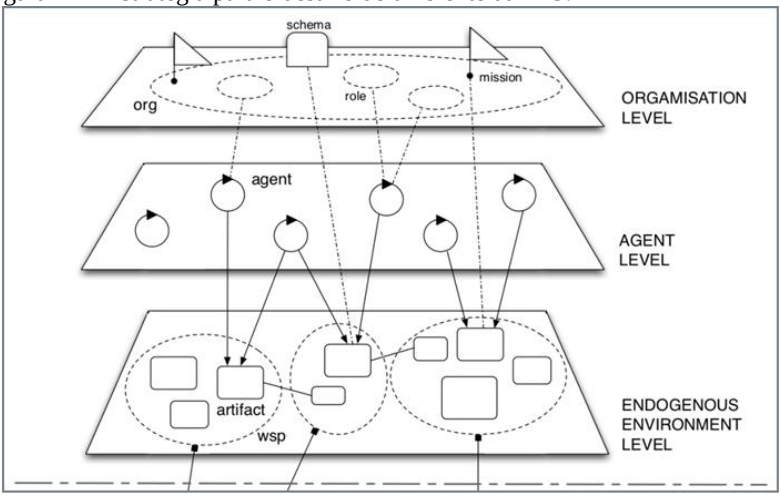
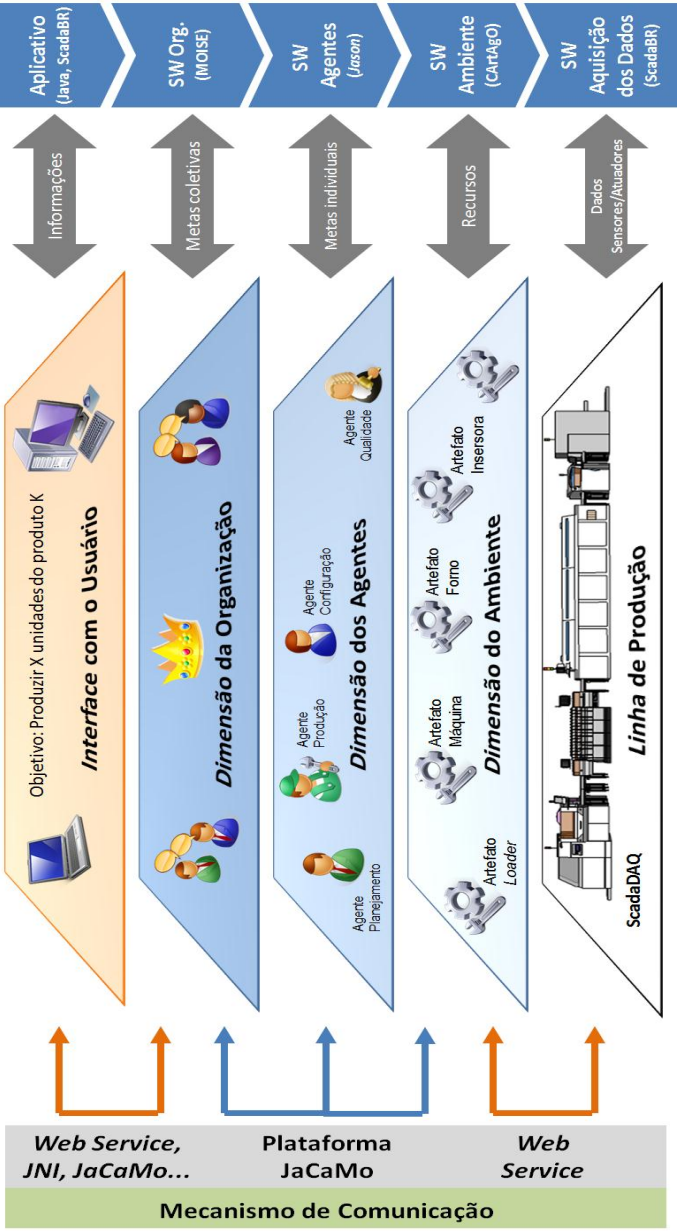


Figura 15 – Visão geral da abordagem proposta.



Dessa forma, pode-se agora apresentar a Figura 16 como uma visão da Arquitetura de Referência para SMA para configuração e monitoramento da PPS – *MAS4SSP*, proposta no presente trabalho de pesquisa.

Sobre esta Arquitetura de Referência se constata que:

- atende ao propósito de apresentar uma solução genérica para o configuração e monitoramento da PPS com SMA.
- é um modelo conceitual que direciona o desenvolvimento para elementos implementáveis.
- atende aos requisitos elencados para a PPS e da tese.

Na próxima seção os Métodos e Ferramentas selecionados para auxiliarem esta Arquitetura de Referência a se tornar implementável em um Modelo Genérico de Modelagem e Implementação são descritos.

4.4. FERRAMENTAS E MÉTODOS

O Modelo de Referência JaCaMo já atende alguns dos Requisitos Desejáveis da proposta, contudo, algumas das constribuições estão explícitas nas Ferramentas e Métodos selecionados. A Figura 17 apresenta a Arquitetura de Referência com os Métodos e Ferramentas posicionados em cada um dos níveis.

Os métodos Prometheus AEOlus e UML auxiliam na modelagem e o SCADA, com o ScadaBR, realiza a integração do ambiente interno ao SMA com o ambiente externo – linha de produção.

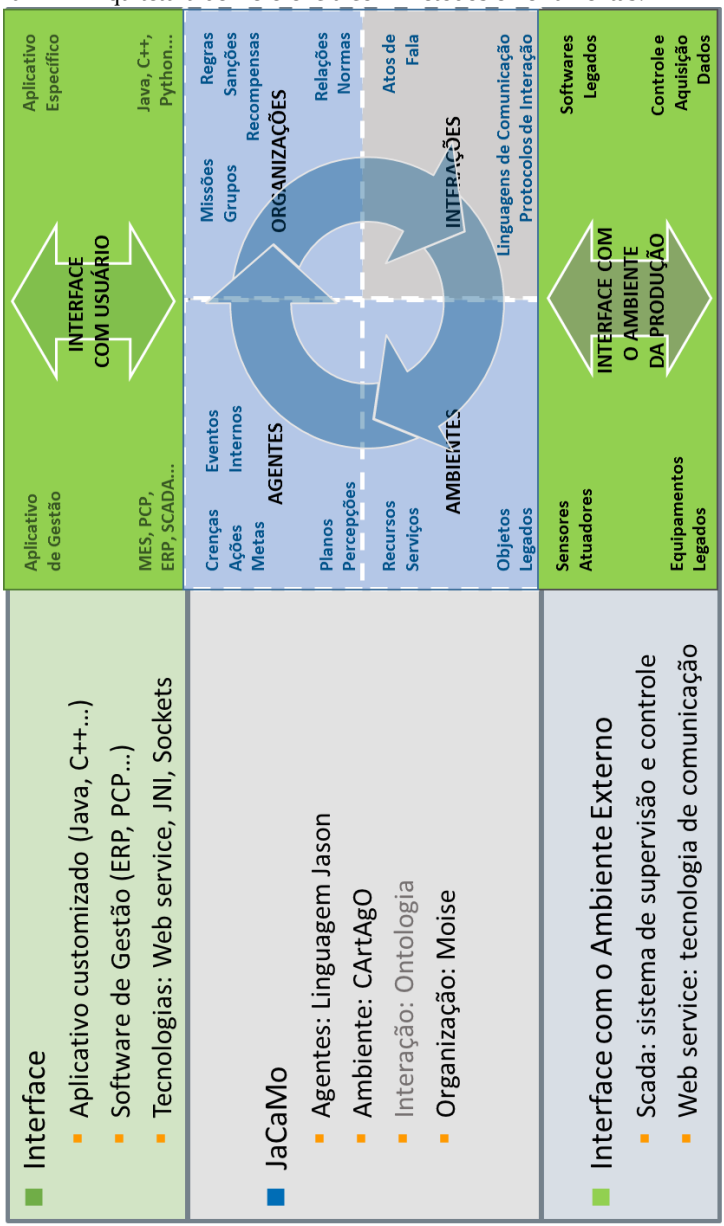
A seguir, são apresentadas os Métodos e as Ferramentas selecionados, com exeção de *Jason*, CArtAgO e MOISE que foram apresentados na seção do Modelo de Referência JaCaMo.

4.4.1. Modelagem do SMA: Prometheus AEOlus

A metodologia Prometheus (PADGHAM & WINIKOFF, 2002) foi desenvolvida pelo grupo de pesquisa *Agent Group* da RMIT *University*, Melbourne, Austrália há mais de dez anos. A metodologia propõe um processo detalhado para especificar, projetar e implementar SMA baseados na arquitetura BDI, mas podendo ser estendida para as outras arquiteturas. O processo de modelagem do Prometheus é iterativo, permitindo que alterações sejam feitas nos *work products* em qualquer fase do projeto. Por isso, para evitar inconsistências, deve ser feita a verificação dos *work products* posteriores, que foram gerados a partir das informações daquele que foi alterado (*crosschecking*).

Figura 16 – Arquitetura de Referência para a nova abordagem proposta.

Figura 17 – Arquitetura de Referência com Métodos e Ferramentas.



Já sua derivação Prometheus AEOlus (UEZ e HÜBNBER, 2014) é um método estendido de Prometheus para desenvolvimento de software orientado a agentes que permite a modelagem integrada do agente, do ambiente e da organização, seguindo assim o paradigma orientado a multiagente. O método foi desenvolvido tendo como base o Prometheus, no qual foram incluídos modelos que permitem a especificação detalhada da organização e do ambiente provenientes do framework JaCaMo. Desta forma, o método Prometheus AEOlus permite a modelagem integrada das três dimensões, sendo que para cada uma são utilizados conceitos específicos, o que diminui o *gap* conceitual existente entre a modelagem e o código gerado.

Para considerar as dimensões de ambiente e da organização novos diagramas foram adicionados à proposta inicial como é destacado na Figura 18 (retângulos preenchidos).

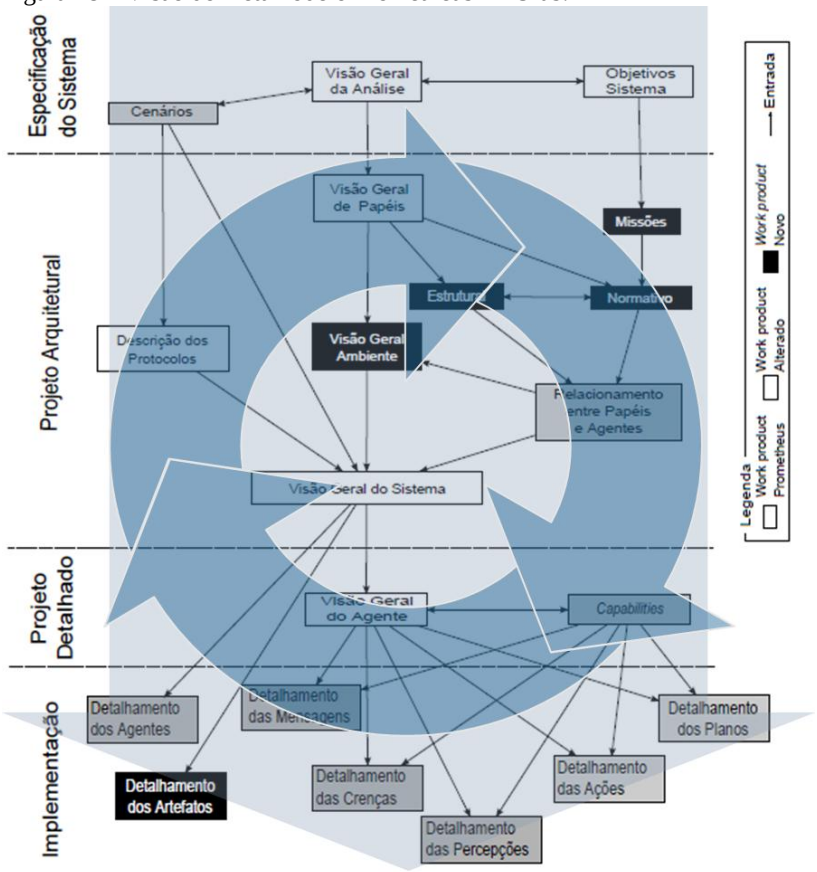
4.4.2. Modelagem dos Artefatos e Recursos: ArgoUML

Os artefatos do *framework* CArtAgO, que faz parte do Modelo de Referência JaCaMo, são entidades **não autônomas.** Os artefatos do *framework* CArtAgO podem ser representados por objetos Java seguindo paradigma orientado a objetos. Por isso, utiliza-se a ferramenta ArgoUML (argouml.tigris.org) – que segue o método UML (JACOBSON et al., 1999) – para modelar os recursos do ambiente (inclusive os artefatos) seguindo o paradigma orientado a objetos em Java.

Com os benefícios da orientação a objetos é possível modelar uma estrutura de classes que representam qualquer linha de PPS bastando derivar artefatos específicos de artefatos genéricos presentes na Arquitetura de Referência modelada. O diagrama de classes é gerado se seguindo o método UML com a ferramenta ArgoUML.

ArgoUML é uma aplicação *Open Source* que usa o método UML para modelar o projeto do software seguindo o paradigma orientado a objetos. ArgoUML é multiplataforma, uma vez que é implementada em Java. ArgoUML é distribuído sob a licença *Eclipse Public License*. O software apresenta suporte para quase todos os tipos de diagrama da UML padrão e além disso, inclui um suporte cognitivo. Atualmente, ArgoUML suporta todos os nove diagramas da UML versão 1.4, por isso, não apresenta uma compatibilidade completa com a versão atual do método UML 2.5.

Figura 18 – Visão do metamodelo Prometheus AEOlus.

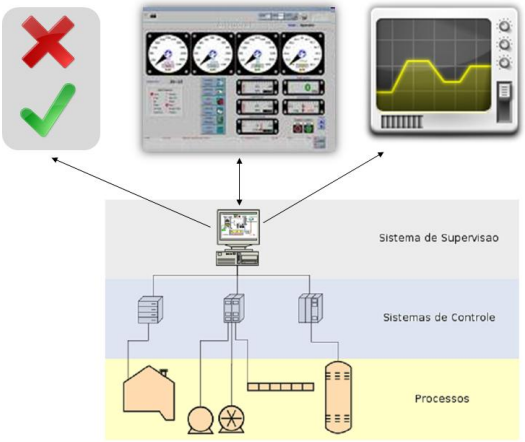


Fonte: Adaptado de Uez e Hübner (2014)

4.4.3. Sistema de Supervisão do processo: ScadaBR

Os Sistemas de Supervisão e Aquisição de Dados (Figura 19) também chamado de Software Supervisório, são sistemas que utilizam software para monitorar e supervisionar as variáveis e os dispositivos de sistemas de controle conectados através de *drivers* específicos em um ambiente industrial (ALBUQUERQUE et al., 2007). Resumidamente, um sistema de supervisão é um tipo software que permite monitorar e controlar partes ou todo um processo industrial.

Figura 19 – Visão geral de um SCADA.



Fonte: Adaptado de www.scadabr.org.br

Já a ferramenta ScadaBR (www.scadabr.org.br) é um SCADA completo, disponibilizado em licença *Open Source* (software livre). O ScadaBR funciona como uma aplicação *web* que é acessada através de um *browser* (Figura 20). O ScadaBR é concebido como um aplicativo Java armazenado em um servidor de aplicações web do tipo Apache Tomcat[13]. Desta forma o ScadaBR se torna independente de plataforma, para executá-lo o sistema operacional deve prover uma *Java Virtual Machine* e um servidor web Apache Tomcat.

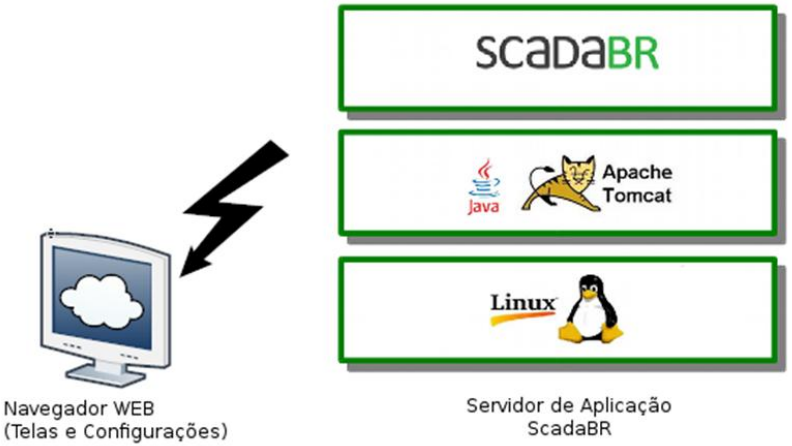
4.4.4. Técnica de integração de sistemas: *Web Service*

Web Service é uma solução utilizada na integração de sistemas e na comunicação entre aplicações diferentes (BENSLIMANE et al., 2008). Com esta tecnologia é possível que novas aplicações possam interagir com aquelas que já existem e que sistemas desenvolvidos em plataformas diferentes sejam compatíveis. Os *Web Services* são componentes que permitem às aplicações enviar e receber dados em

[13] O Tomcat é um servidor de aplicações Java do tipo JEE, porém não é um servidor de EJBs. Desenvolvido pela Apache Software Foundation, é distribuído como software livre dentro do projeto Apache Jakarta, sendo oficialmente apoiado pela Sun como a implementação de referência para as tecnologias Java Servlet e Java Server Pages (JSP). Ele tem a capacidade de atuar também como servidor web, ou pode funcionar integrado a um servidor web dedicado como o Apache ou o IIS. Como servidor web, ele provê um servidor web HTTP puramente em Java.

formato XML. Cada aplicação pode ter a sua própria ‘linguagem’, que é traduzida para uma ‘linguagem universal’, o formato XML. Essencialmente, o *Web Service* faz com que os recursos da aplicação do software estejam disponíveis sobre a rede de uma forma padronizada.

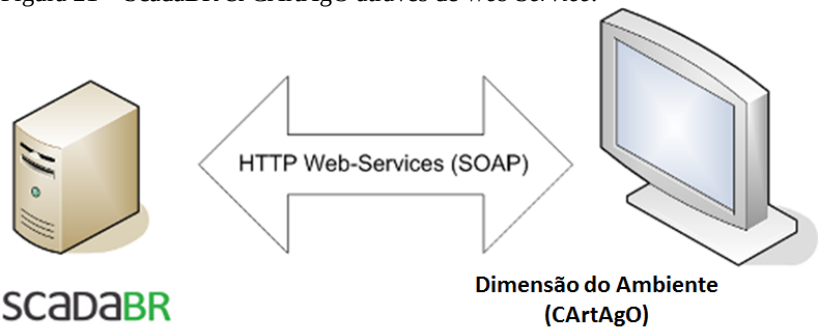
Figura 20 – Como funciona o ScadaBR.



O objetivo dos *Web Services* é a comunicação de aplicações através da rede de computadores. Esta comunicação é realizada com intuito de facilitar a EAI (*Enterprise Application Integration*) que significa a integração das aplicações de uma empresa, ou seja, interoperabilidade entre a informação que circula numa organização nas diferentes aplicações como, por exemplo, o comércio eletrônico com os seus clientes e seus fornecedores. Esta interação constitui o sistema de informação de uma empresa. E para além da interoperabilidade entre as aplicações, a EAI permite definir um *workflow* entre as aplicações e pode constituir uma alternativa aos ERP (*Enterprise Resource Planning*). Com um *workflow* é possível otimizar e controlar processos e tarefas de uma determinada organização.

Na abordagem proposta neste trabalho de pesquisa, o *Web Service* é utilizado como meio de comunicação entre o sistema ScadaBR e o *framework* CArtAgO responsável pela gestão dos recursos da dimensão Ambiente do JaCaMo (Figura 21). Pode ser também uma opção para interfaceamento entre o SMA e a interface com o usuário.

Figura 21 – ScadaBR & CArtAgO através de *Web Service*.



Utilizando a tecnologia *Web Service*, uma aplicação pode invocar outra para efetuar tarefas simples ou complexas mesmo que as duas aplicações estejam em diferentes sistemas e escritas em linguagens diferentes. Em outras palavras, os *Web Services* fazem com que os seus recursos estejam disponíveis para que qualquer aplicação cliente possa operar e extrair os recursos fornecidos pelo *Web Service*.

A etapa seguinte na estratégia (Figura 2) de propor uma nova abordagem SMA para a configuração e o monitoramento da PPS é a proposta de um Modelo Genérico de Modelagem e Implementação.

4.5. MODELO GENÉRICO DE MODELAGEM & IMPLEMENTAÇÃO

O modelo genérico, cujo o objetivo é servir para qualquer cenário de configuração e monitoramento da PPS, utiliza os elementos conceituais da Arquitetura de Referência e já os direciona para elementos implementáveis por meio da modelagem do SMA – com Prometheus AEOlus, dos recursos do Ambiente – com o método UML e do Sistema Supervisório – com o projeto e definição das tags no ScadaBR (via uma interface para *Web Service*).

4.5.1. Modelagem utilizando o Método Prometheus AEOlus

A modelagem do SMA genérico para configuração e monitoramento da PPS utiliza o método Prometheus com a sua extensão para o *framework* JaCaMo, chamada Prometheus AEOlus (UEZ & HÜBNER, 2013). O método Prometheus AEOlus possui as seguintes

fases: especificação do sistema, projeto de arquitetura, projeto detalhado e implementação. Cada fase gera um conjunto de diagramas do método Prometheus AEOlus que modelam o SMA considerando as suas três dimensões (Agentes, Ambiente e Organização).

4.5.1.1. Especificação do Sistema

O objetivo desta fase é criar uma definição clara e precisa do sistema que será desenvolvido. Nesta fase são identificadas: as ações que os agentes podem executar; as percepções recebidas do ambiente; os objetivos e subobjetivos do sistema; e os cenários.

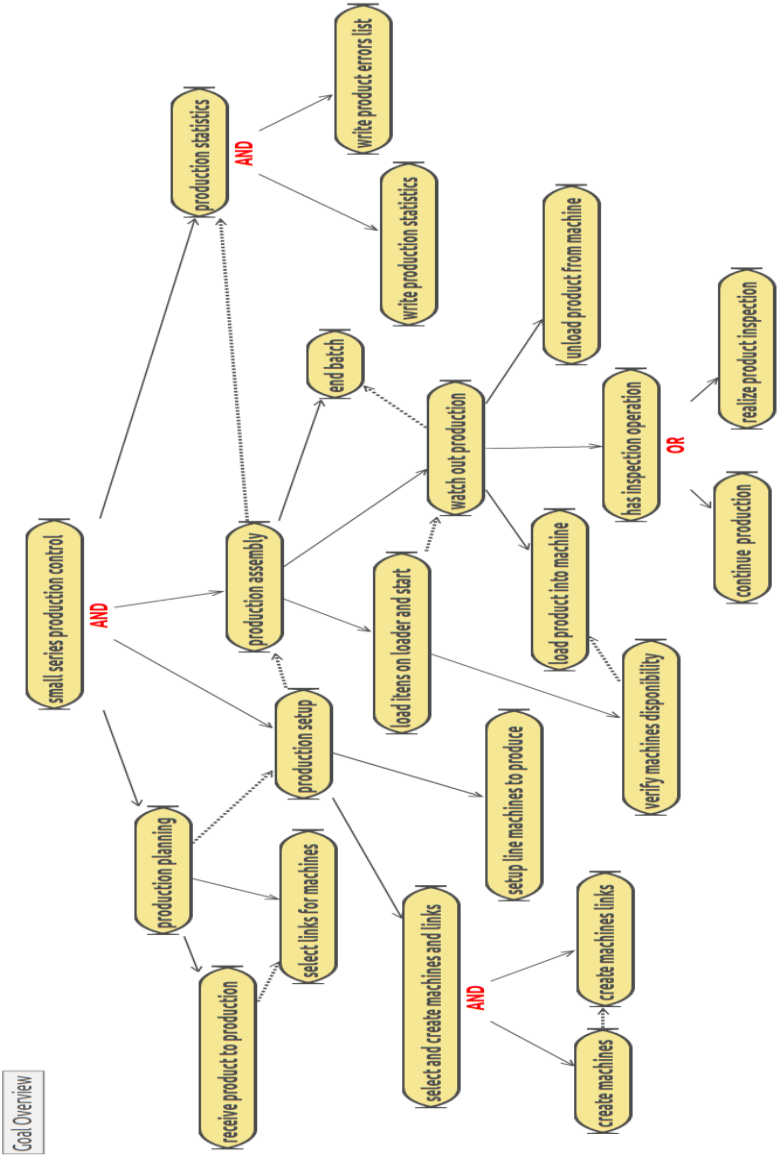
Inicialmente, define-se o que o sistema deve fazer, ou seja, quais são os objetivos do sistema. Os objetivos são definidos com base na descrição inicial do sistema. A partir disto, os objetivos são refinados em subobjetivos por meio da pergunta: "Como este objetivo pode ser alcançado?". A resposta gera um subobjetivo que é ligado ao objetivo-pai através de uma linha contínua que indica que o objetivo filho precisa ser atingido para que o objetivo pai possa ser alcançado. Se um objetivo for decomposto em subobjetivos do tipo *and*, isto indica que todos os subobjetivos devem ser alcançados para que se alcance o objetivo pai. Se a decomposição for do tipo *or* isto indica que qualquer objetivo filho atingido possibilita o objetivo pai seja alcançado. Pode-se também estabelecer uma relação de precedência entre os objetivos, ligando-os com uma seta pontilhada.

A Figura 22 apresenta os objetivos e subobjetivos do SMA para o configuração e monitoramento da PPS. Percebe-se que o objetivo principal *small series production control* é composto pelos subobjetivos *production planning*, *production setup*, *production assembly* e *production statistics*. Estes subobjetivos são uma decomposição do tipo *AND* do objetivo principal, neste tipo de decomposição o objetivo principal somente será atingido quando cada um dos subojetivos for atingido. Estes objetivos são na verdade objetivos que não mudam em nada dos objetivos de outros tipos de sistema de produção, o diferencial aqui é que todas estes objetivos deverão ser alcançados autonomamente pelos agentes que fazem parte do SMA.

Outro ponto que merece destaque neste diagrama é a indicação de precedência (seta tracejada) que mostra que mesmo sendo um sistema complexo com várias entidades de hardware e software existe um ordenamento que deve ser seguido na linha de produção. Atualmente estes objetivos são cumpridos normalmente pelos operadores da linha de

produção que controlam a produção seguindo os procedimentos elaborados pela engenharia – seguindo um manual de boas práticas.

Figura 22 – Diagrama de objetivos do SMA para a PPS.



Quando se pensa nos objetivos do sistema, normalmente também se pensa nos cenários que descrevem como esses objetivos serão alcançados. Os cenários são utilizados para descrever como o sistema deve se comportar em determinadas situações, seja durante a execução normal do sistema ou caso ocorra alguma exceção. Os cenários são formados basicamente por um conjunto de passos que indicam as ações, percepções, objetivos ou outros processos do sistema. Os cenários são descritos de forma textual e permitem visualizar quais papéis estão envolvidos, as percepções, as ações e se há a necessidade de um novo objetivo.

Os cenários deste trabalho de pesquisa possuem relação direta com os sub-objetivos traçados para o cumprimento do objetivo principal do SMA. As Tabelas 7, 8, 9, 10 e 11 descrevem cada um dos cenários indicando as ações, percepções e objetivos do sistema.

Tabela 7 – Descrição do cenário – *production planning*.

Cenário *production planning*	
Descrição:	o operador seleciona os produtos a serem produzidos durante o turno (ou insere um novo produto). Um lista com os produtos cadastrados deve existir (com suas características). Após a seleção dos produtos o SMA organiza a lista seguindo algum critério previamente definido: similaridade entre os produtos, maior/menor tamanho de lote, maior/menor tempo de produção do lote, maior/menor tempo de setup, etc.
Gatilho:	início do sistema
Passos:	▪ **Ação**: criar os produtos; selecionar os produtos a produzir; ▪ **Percepção**: início da produção; lista ordenada de produtos a produzir.

A Tabela 7 descreve o cenário de planejamento da produção onde após o início do turno de produção é necessário selecionar quais serão os produtos a produzir (ou criá-los) e após a análise do especialista (que pode ser um agente) é criada uma lista ordenada de produtos. Este ordenamento pode levar em conta as similaridades entre os produtos que resulta em tempos de configuração (*setup*) menores entre a troca de produtos, pois as configurações podem ser similares ou até mesmo idênticas. Ou então utilizar outros critérios mais simples como: tamanho do lote ou tempo de produção.

Tabela 8 – Descrição do cenário – *machines setup*.

Cenário *machines setup*	
Descrição:	com a lista ordenada, carregar o produto ***i*** e configurar as máquinas (que são artefatos) com os parâmetros específicos para o produto ***i*** (por exemplo, temperatura do forno de refusão). Após todas as máquinas estarem configuradas para o produto ***i*** se aguarda pelo início da produção do lote.
Gatilho:	lista ordenada de produtos
Passos:	▪ **Ação**: selecionar produto; criar lista de máquinas; configurar as máquinas; ▪ **Percepção**: máquinas configuradas.

A Tabela 8 descreve o cenário de configuração que a princípio parece ser simples, porém, com a diversidade de equipamentos na linha de produção, cada qual com um software específico, torna esta tarefa normalmente complexa. Após a lista ordenada de produtos ser criada e o produto a produzir ser selecionado na lista. Este cenário cria a linha de produção através da criação da lista de máquinas, em seguida cada máquina da lista é configurada para o produto selecionado, por fim a linha de produção está pronta com as máquinas configuradas e prontas para iniciar a produção do lote.

Tabela 9 – Descrição do cenário – *product assembly*.

Cenário *product assembly*	
Descrição:	com as máquinas configuradas (e disponíveis) para o produto ***i*** se inicia a montagem dos produtos do lote. O exemplar ***n*** do produto ***i*** é carregado na máquina ***k***, que realiza a sua operação (***op***) durante o intervalo de tempo (***Δt***) e após a execução da operação (***op***) o exemplar ***n*** é descarregado da máquina ***k*** seguindo para a próxima máquina da lista de máquinas da linha de produção.
Gatilho:	máquinas configuradas
Passos:	▪ **Ação**: começa produção do lote; carrega exemplar ***n*** do produto ***i*** na máquina ***k***; executa operação ***op*** durante o intervalo de tempo (***Δt***); ▪ **Percepção**: descarrega exemplar ***n*** do produto ***i*** na máquina ***k***;

O cenário da Tabela 9 consiste na fase de montagem dos itens do lote. O objetivo do SMA é controlar o fluxo de cada item do produto na linha de produção passando de uma máquina a outra até o produto pronto chegar na última máquina da linha de produção. Neste cenário se inicia a produção, o item do lote é carregado na máquina, a máquina

executa suas operações sobre o item do produto, após o produto é descarregado da máquina seguindo para a próxima até o final da linha de produção.

Tabela 10 – Descrição do cenário – *product inspection*.

Cenário *product inspection*	
Descrição:	durante a montagem o produto ***i*** pode ser inspecionado várias vezes na busca por falhas. Cada inspeção emite um diagnóstico (APROVADO ou REPROVADO – lista de defeitos). Quando APROVADO o produto deve seguir para a próxima etapa da montagem já quando REPROVADO a linha deve parar o DEFEITO é enviado para o especialista que com a CAUSA do defeito CORRIGE a máquina que originou o defeito. Após a produção recomeça. A lista de DEFEITOS é armazenada para criação do relatório de produção e lista de ações a serem tomadas para melhorar a produção.
Gatilho:	chegada exemplar ***n*** do produto ***i*** na máquina de inspeção ***kmv***
Passos:	▪ **Ação**: carrega exemplar ***n*** do produto ***i***; realizar a inspeção visual do exemplar ***n*** do produto ***i***; ▪ **Objetivo**: diagnóstico da inspeção (OK/NOK).

A Tabela 10 descreve o processo de inspeção de um item do lote de produtos. A inspeção na PPS é peça fundamental para o bom funcionamento da linha de produção pois um produto com defeito pode significar que os demais também foram produzidos com o mesmo defeito e já não há tempo de correção devido ao tamanho reduzido do lote, isto pode inviabilizar economicamente o lote. Neste cenário a máquina de inspeção visual automática APROVA ou REPROVA o item do lote. Caso um DEFEITO seja detectado, a linha PARA, a máquina informa o defeito para o sistema especialista que emite um diagnóstico indicando a origem da falha e que atitude tomar na linha de produção para evitar a repetição da falha.

Por fim, o cenário da Tabela 11 emite um relatório estatístico com os itens produzidos e também uma lista com os defeitos encontrados e as ações tomadas. Estas ações corretivas são alterações nas configurações de equipamentos ou processo para evitar novos defeitos na produção de um novo lote do produto no futuro

O próximo passo é modelar a Visão Geral da Análise que permite que se observe quais ações e percepções foram utilizadas nos cenários. Este diagrama pode também incluir os atores que já foram identificados. Este diagrama é uma forma de visualizar graficamente o

que foi descrito em cada cenário, desta forma também pode ser substituído pela descrição textual dos cenários.

Tabela 11 – Descrição do cenário – *statistics*.

Cenário *statistics*	
Descrição:	quando a produção do lote finaliza um relatório de produção deve ser emitido (estatística da produção: total produzido, número e tipo dos defeitos, ações corretivas executadas, etc.). Com base na lista de defeitos o especialista define as alterações da configuração dos equipamentos a fim de evitar que as falhas ocorram na produção de outro lote.
Gatilho:	fim da produção do lote do produto ***i***
Passos:	▪ **Ação**: emitir relatório estatístico do lote do produto ***i***; ▪ **Ação**: realizar ações corretivas para novo lote do produto ***i***; ▪ **Percepção**: fim da produção do lote do produto ***i***.

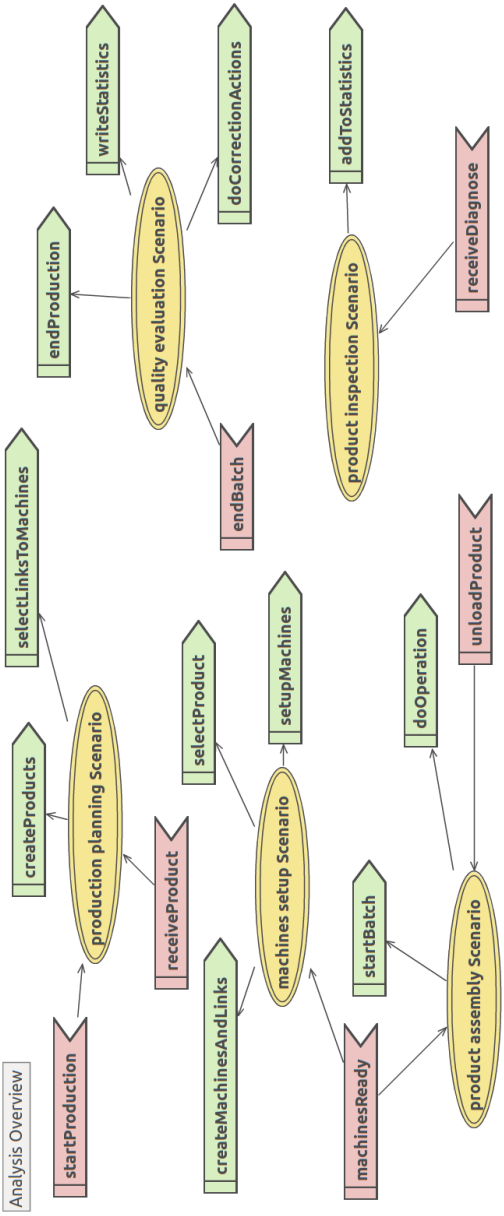
O benefício deste diagrama é o relacionamento dos cenários com as ações e percepções do SMA. É interessante notar no diagrama modelado (Figura 23) que uma percepção pode interagir com mais de um cenário, como é o caso da percepção *machinesReady* que atende tanto ao cenário *machines setup* como o cenário *product assembly*.

4.5.1.2. Projeto de Arquitetura

Essa fase visa definir quais tipos de agentes farão parte do sistema e como estes interagem entre si para atingir os objetivos. Nessa fase ocorre: definição dos tipos de agentes e papéis que podem assumir; descrição das interações entre os agentes e os papéis; definição dos artefatos do ambiente; definição da estrutura macro do sistema.

Depois de definido o sistema, tendo como base a descrição dos cenários e os objetivos se definem os papéis que os agentes podem desempenhar. Os papéis são definidos a partir da descrição dos cenários, que oferece uma visão de qual comportamento é esperado para cada estado em determinada situação. Na visão geral dos papéis, os papéis identificados são visualizados juntamente com as ações e as percepções compartilhadas entre eles.

Figura 23 – Diagrama de Visão Geral da Análise.



A Figura 24 mostra os papéis definidos para o SMA genérico para controlar a PPS:

- *rPlanning* – responsável pela seleção do produto a ser produzido;
- *rSetup* – assume a responsabilidade de configurar as máquinas para a produção de um lote do produto selecionado;
- *rAssembly* – reponsável controle da montagem do produto, este papel interage fortememte com os equipamentos;
- *rInspection* – é o papel responsável por verificar a presença de falhas no produto;
- *rStatistics* – assume a função de emitir um relatório com dados da produção;
- *rExpert* – reponsável por avaliar a causa dado um defeito e também sugerir ações para evitar este defeito novamente.

O Diagrama de Missões permite a definição das missões que podem ser atribuídas a cada papel. As missões são definidas a partir do agrupamento dos objetivos correlatos. É importante ressaltar que o papel que assumir uma missão deve se comprometer a atingir cada um dos objetivos que fazem parte da missão.

O Diagrama de Missões (Figura 25) apresenta as missões (leadUp, setup, assembly e statistics) definidas para o SMA genérico para configurar e monitorar a PPS. As missões agrupam os objetivos do SMA em fases que caracterizam as etapas de preparação da produção, produção e análise da produção.

A definição dos grupos de papéis pode ser feita com base nos cenários de uso do sistema e nas características dos papéis definidos anteriormente. Papéis com características relacionadas tendem a fazer parte de um mesmo grupo. As ligações entre os papéis e a formação dos grupos são definidas por meio do diagrama Estrutural. O Diagrama Estrutural serve para representar a estrutura da organização que rege o sistema definindo grupos de papéis e as ligações de autoridade, comunicação, compatibilidade e conhecimento entre os agentes de um mesmo grupo ou de grupos diferentes. Algumas vezes, para auxiliar na especificação estrutural é necessário a definição de um ‘papel abstrato’. Os grupos são formados a partir de papéis que podem ou não herdar características de um papel abstrato. Para cada papel pode existir uma cardinalidade mínima e máxima que indica quantos agentes podem exercer aquele papel no grupo para que o grupo seja bem formado. A relação entre os grupos e os papéis é representada um por uma linha contínua que liga o grupo ao papel. Um diamante define qual é o grupo

e a cardinalidade do papel é exibida ao lado do papel. Uma seta vazada liga o papel ao seu ‘papel pai’.

Figura 24 – Visão Geral dos Papéis.

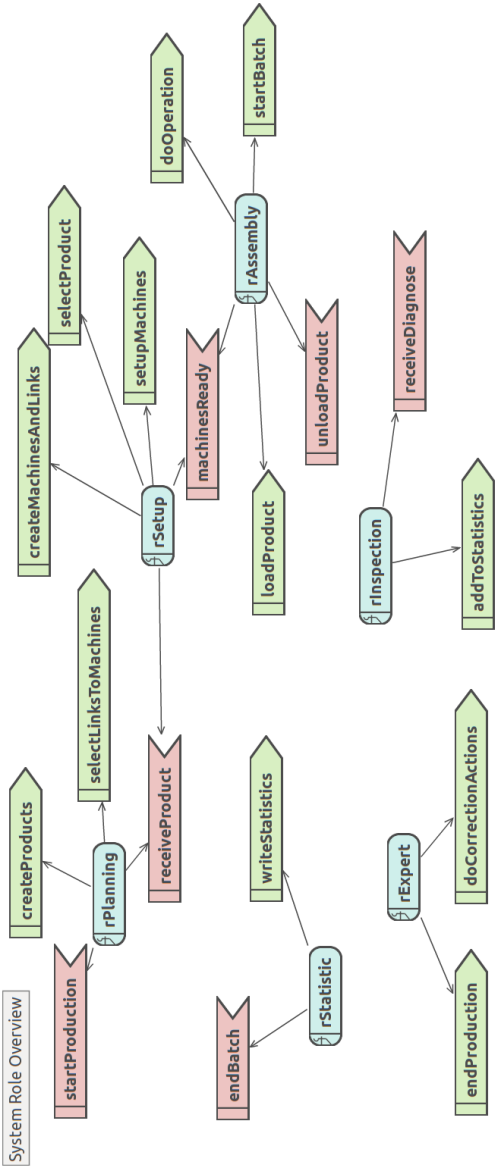
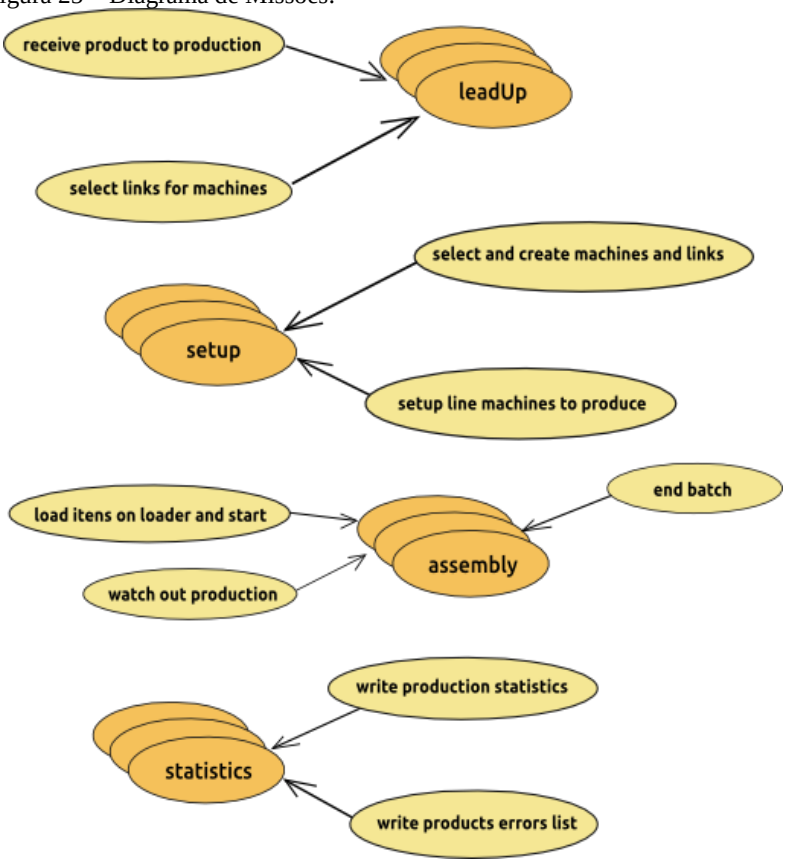
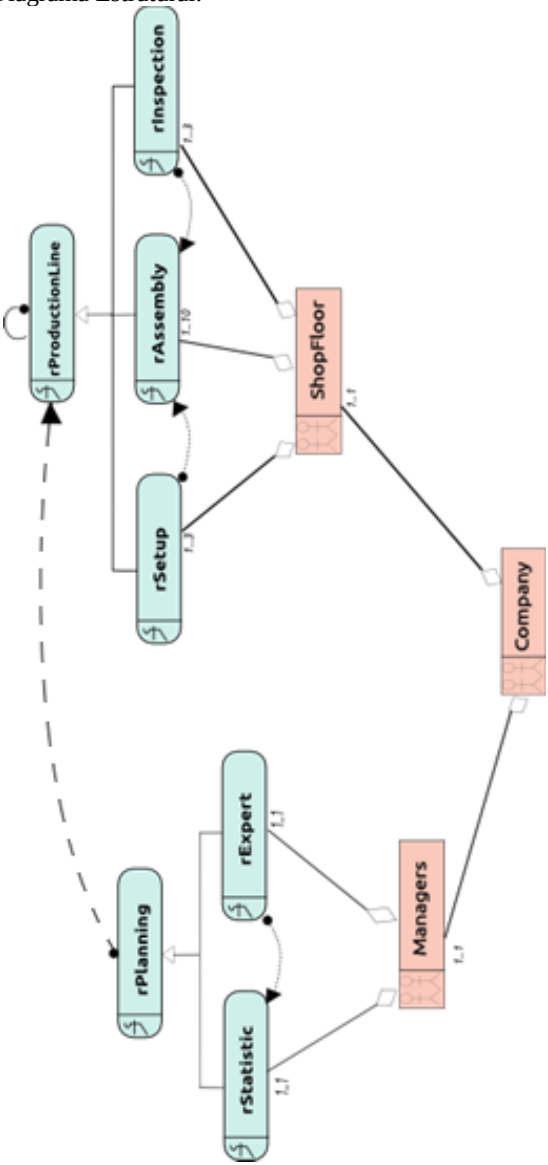


Figura 25 – Diagrama de Missões.



Observa-se no Diagrama Estrutural (Figura 26) que os papéis estão divididos em dois grupos (*Managers* e *ShopFloor*) que fazem parte do grupo maior (*Company*). O papel *rPlanning* tem autoridade sobre o papel abstrato *rProductionLine* e, consequentemente, sobre todos os papéis que herdam as características desse papel. Porém, essa autoridade somente vale entre o *rPlanning* e a *rProductionLine*. O diagrama Estrutural foi incluído no método Prometheus AEOlus.

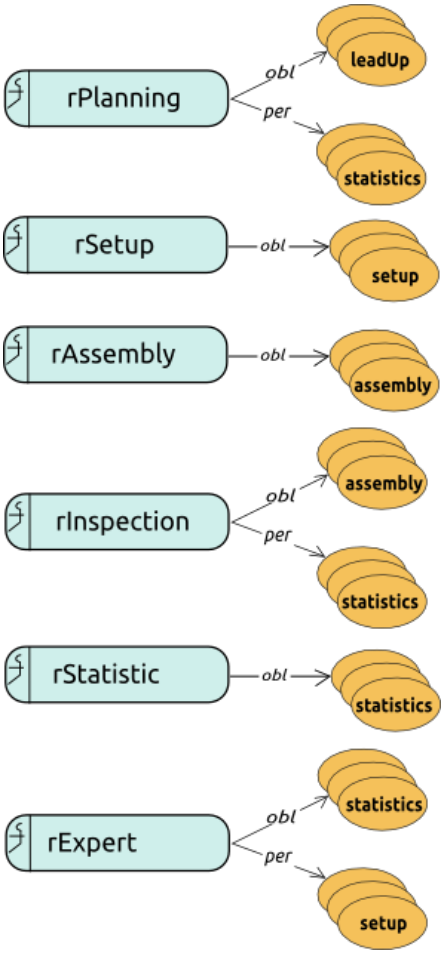
Figura 26 – Diagrama Estrutural.



Os diagramas de Missões e Estrutural podem ser especificados em qualquer ordem, já que não dependem de informações um do outro.

No caso do diagrama Normativo, porém, é importante que as missões já tenham sido definidas, bem como a estrutura da organização. O Diagrama Normativo possui a função de determinar com quais missões cada papel pode ou deve se comprometer. Um papel possui a obrigação (*obligation*) de se comprometer com uma missão ou possui a permissão (*permition*) para fazê-lo.

Figura 27 – Diagrama Normativo.



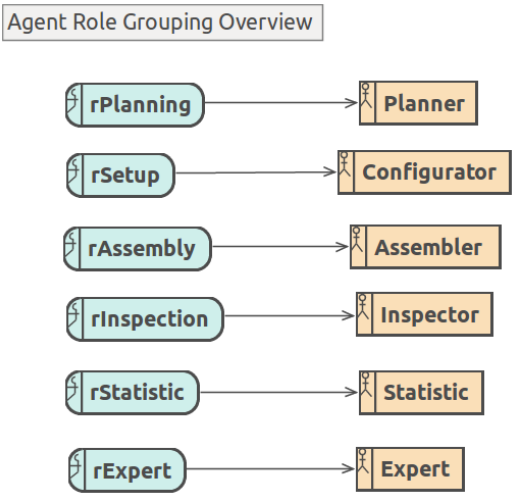
A Figura 27 demonstra que as missões estão correlacionadas com os papéis conforme as fases da produção já mencionadas: *leadUp*, *setup, assembly* e *statistics*. Merece destaque também a relação de obrigação ou permissão entre os papéis e as missões. É fácil constatar que o agente que assumir o papel *rSetup* deve ser responsável pela missão *setup*. Já o agente que assumir o papel de *rExpert* tem a permissão de assumir a missão *setup* – alterar a configuração devido a detecção de uma falha no processo – mas tem obrigação de assumir a missão *statistics*.

Um agente pode assumir mais de um papel no sistema, por isso é importante definir quantos e quais serão os agentes desenvolvidos e quais papéis cada agente pode assumir. Deve-se levar em conta que alguns papéis exigem do agente algumas características específicas, portanto um tipo de agente que possua essas características deverá ser criado.

Na Figura 28 se observa que a maioria dos papéis possui um agente correlato, a exceção é o agente *Statistic* que desempenhará dois papéis, pois durante a produção está focado principalmente em sugerir ações corretivas na ocorrência de uma falha – especialista – e no final da produção tem o objetivo de emitir um relatório com os dados sobre a produção – estatístico.

Os tipos de agente, na verdade, são os agentes que são efetivamente implementados no sistema. Um agente pode assumir mais de um papel, da mesma forma que pode haver mais de um agente desempenhando o mesmo papel. Por isso, os tipos de agente podem ser definidos conforme as necessidades detectadas pela equipe de análise, que pode ser a partir da descrição dos cenários, quando fica evidente que um papel só pode ser exercido por um tipo específico de agente, ou com base nos diagramas Estrutural e de Missões, quando pode ser visualizado que um agente precisa ter alguma característica específica para exercer um determinado papel ou até mesmo por uma decisão de projeto. Obviamente, a definição dos agentes e de como estes se relacionam com os papéis pode levar a revisão dos papéis já definidos e das ligações entre esses papéis.

Figura 28 – Diagrama agrupamento de papéis e agentes.



O Diagrama Visão Geral do Ambiente descreve como o sistema se relaciona com o ambiente no qual está inserido através das ações realizadas e das percepções recebidas do ambiente. O ambiente possui um conjunto de artefatos distribuídos em *workspaces*. Os artefatos podem prover ao agente funcionalidades específicas ou permitir que os agentes tenham contato com outros sistemas ou com usuários. Para utilizar um artefato, o agente precisa entrar no ambiente no qual este artefato se encontra.

Dada a quantidade de equipamentos representáveis na linha de PPS se estima que este diagrama (Figura 29) se tornaria o mais complexo, contudo, com a abordagem seguindo o Modelo de Referência JaCaMo este diagrama promove uma simplificação do SMA quando comparado com a abordagem de agentificação[14], onde todas as entidades – relevantes para o cumprimento do objetivo – do sistema são modeladas como agentes. O Diagrama Visão Geral do Ambiente mostra o relacionamento de cada agente com o ambiente (povoado de artefatos). Este diagrama simplifica o SMA porque permite que os agentes foquem sua atividade somente no cumprimento da sua meta e

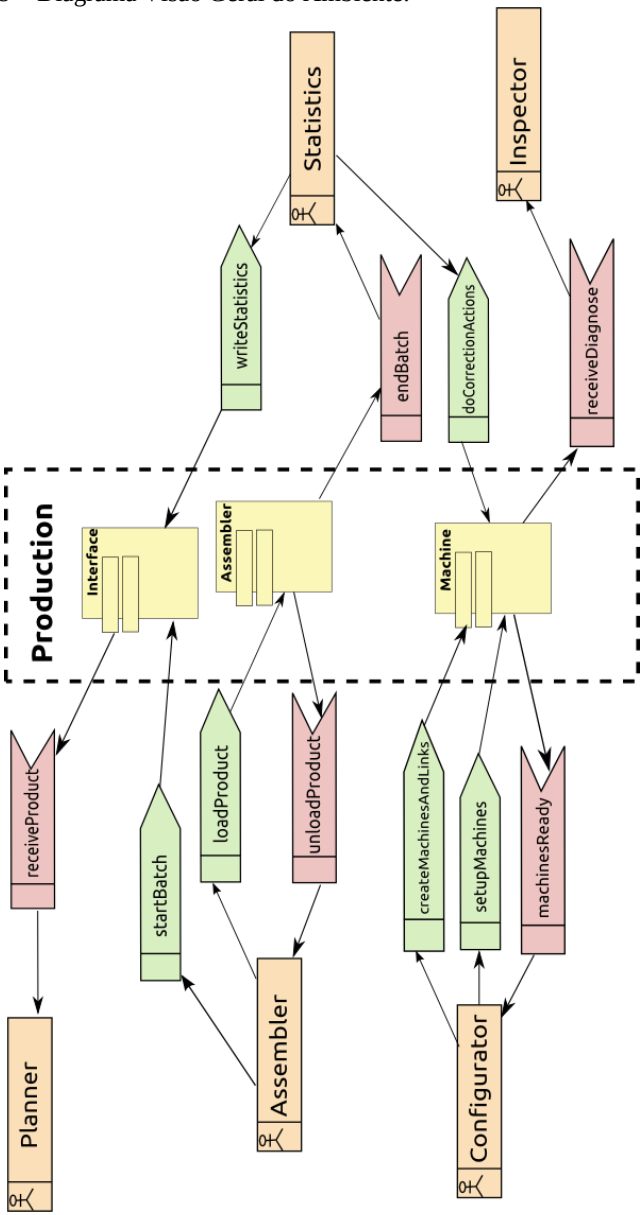
[14] Discutida na seção sobre agentes na indústria, basicamente consiste em transformar todos os ´objetos representáveis´ em agentes.

não se criam ‘agentes recursos’ que seriam meros produtores ou consumidores de ‘dados brutos’ (temperatura, presença/ausência).

O Diagrama Visão Geral do Ambiente mostra que os agentes se relacionam com um *workspace* chamado *Production* onde lá estarão os artefatos genéricos – como o artefato *Machine* – que na implementação serão instanciados recursos especializados que representaram cada um dos equipamentos da linha de produção alvo (máquinas, esteiras, robôs...). Também está presente um outro artefato chamado Interface que é um espaço que disponibiliza recursos para a criação de uma interface com o usuário dentro do próprio SMA. E o artefato *Assembler* é responsável pela execução das ações do agente *Assembler* nos recursos do ambiente, optou-se por modelar desta forma para deixar claro a divisão entre a entidade autônoma agente *Assembler* e o recurso não autônomo artefato *Assembler*.

O próximo diagrama, Visão Geral do Sistema descreve a estrutura geral e estática do sistema, mostrando os agentes, as ações e percepções de cada agente. As mensagens que os agentes trocam e protocolos de comunicação – quando o número de mensagens é muito grande, também são representados neste diagrama. As mensagens são enviadas por um agente e recebidas por outro, indicando a fonte e o destino da mensagem. Em alguns casos a mesma mensagem deve ser enviada para todos os agentes do sistema (envio em *broadcast*), neste caso, o destinatário não é informado.

Figura 29 – Diagrama Visão Geral do Ambiente.



Este diagrama (Figura 30) apresenta semelhanças com o diagrama de Visão Geral dos Papéis e realmente é semelhante pois o agrupamento entre agentes em papéis (Figura 28) demonstra isto. Contudo, este diagrama destaca os agentes e como estes interagem entre si e com o ambiente. Merece destacar ações de *broadcasting* como *selectProducts* que 'avisa' todos os agentes que esta ação está acontecendo.

Quando há necessidade, podem ser descritos os protocolos de comunicação entre os agentes. Os protocolos são possíveis sequências de mensagens trocadas entre os agentes e são descritos utilizando-se a notação do diagrama de sequências da AUML. Por hora não foram modelados protocolos de comunicação, que são uma sugestão de pesquisa futura quando expandir o SMA para outros níveis da empresa.

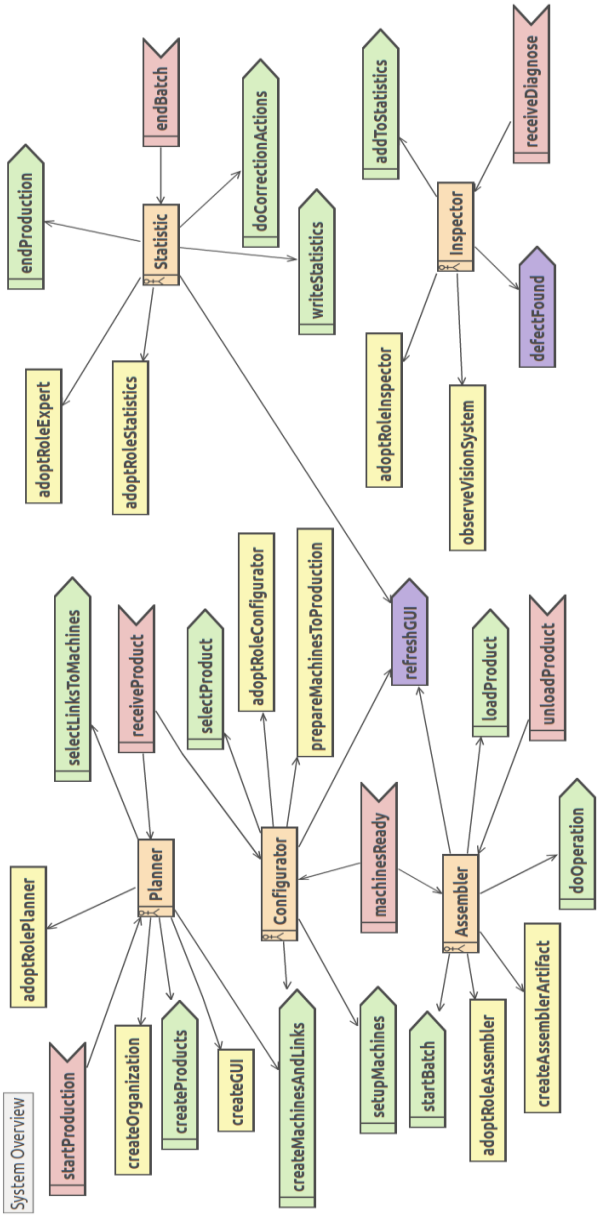
4.5.1.3. Projeto Detalhado

Essa fase possui o objetivo de definir a estrutura interna dos agentes e como eles satisfarão seus objetivos dentro do sistema. Especificar a estrutura interna do agente é um processo de refinamento, que inclui:

- definir e desenvolver as habilidades utilizadas pelos agentes e o relacionamento entre elas;
- descrever os planos, eventos e crenças e o relacionamento entre eles.

Com base no Diagrama Normativo, que descreve os objetivos de cada agente, nos Diagramas Estrutural e de Visão Geral do Sistema, que descrevem como os agentes interagem (o primeiro em termos de permissões e o segundo através de protocolos e mensagens) e no Diagrama de Visão Geral do Ambiente, que demonstra como o agente se relaciona com o ambiente, cria-se o Diagrama de Visão Geral do Agente. Este diagrama descreve o agente internamente, mostrando seus planos, habilidades e crenças.

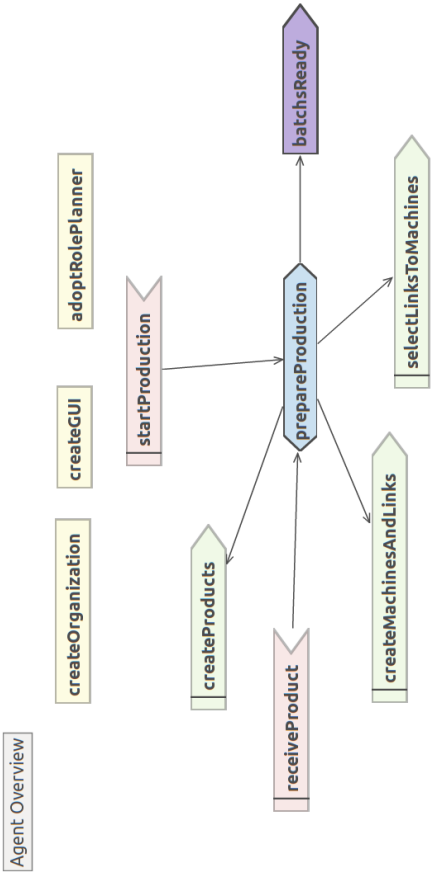
Figura 30 – Diagrama Visão Geral do Sistema.



A Figura 31 apresenta a visão geral do agente *Planner* que tem como plano *prepareProduction* e ainda emite uma mensagem *batchReady*. As ações e percepções já são conhecidas.

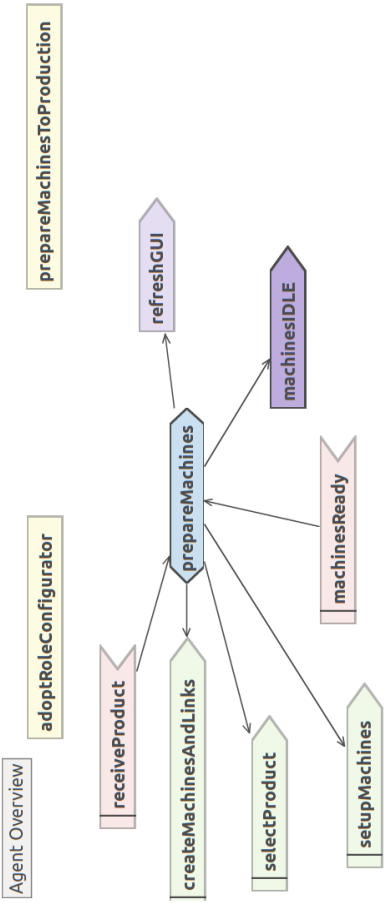
Além disso, cada agente possui um conjunto de 'planos organizacionais' criados para a gestão das metas coletivas do SMA. Nos diagramas seguintes, estes planos são representados pelos retângulos na parte superior de cada diagrama.

Figura 31 – Diagrama de Visão Geral do Agente - *Planner*.



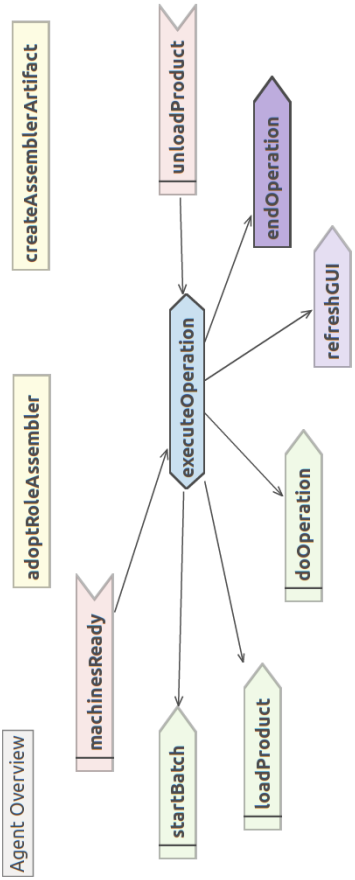
A Figura 32 apresenta a visão geral do agente *Configurator* que tem como plano *prepareMachines* e ainda emite uma mensagem *machinesIDLE*. As ações e percepções já são conhecidas.

Figura 32 – Diagrama de Visão Geral do Agente - *Configurator*.



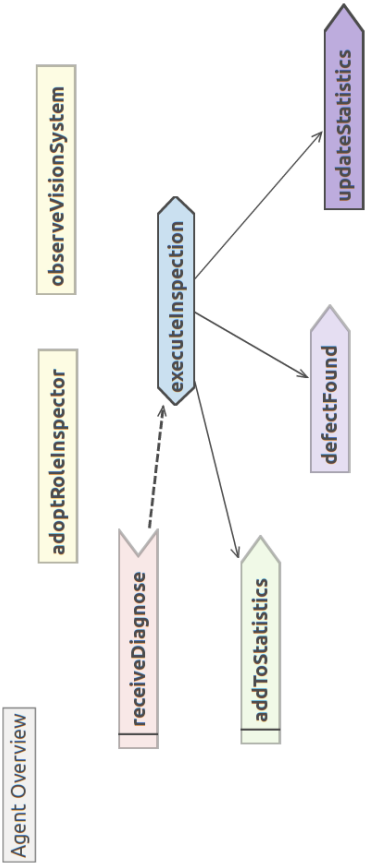
Já o agente *Assembler* (Figura 33) tem o plano de *executeOperation* (que representa realizar uma tarefa de montagem por determinada máquina) e emite a mensagem *endOperation* que libera o artefato para outro item do lote do produto e o item do lote passa para a próxima operação na linha de produção.

Figura 33 – Diagrama de Visão Geral do Agente - *Assembler*.



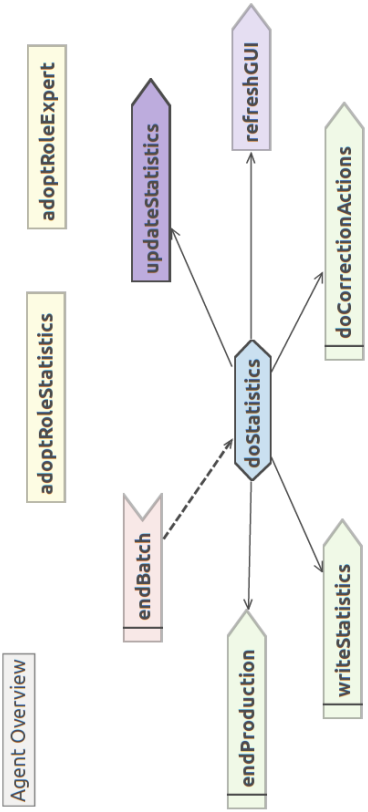
Destaca-se no agente *Inspector* (Figura 34) o plano de *executeInspection*, que representa o acesso ao sistema de inspeção visual para avaliar a qualidade do produto e a mensagem para *updateStatistics* que envia novos dados do processo com informações da produção.

Figura 34 – Diagrama de Visão Geral do Agente - *Inspector*.



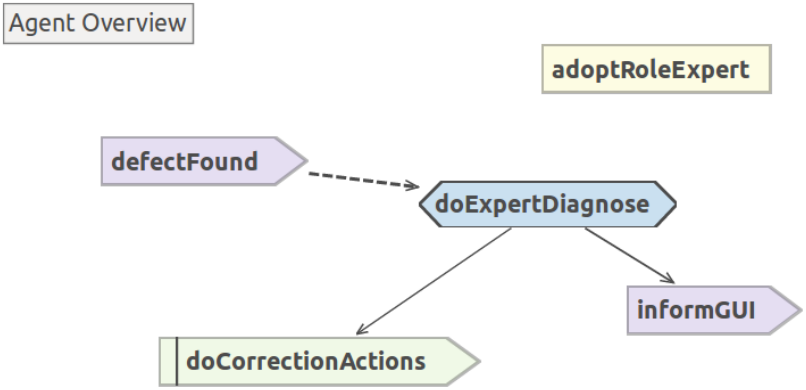
O agente *Statistic* (Figura 35) é responsável por produzir as estatísticas da produção do lote, gravar os defeitos detectados e definir as causas dos defeitos.

Figura 35 – Diagrama de Visão Geral do Agente - *Statistic*.



O agente *Expert* (Figura 36), baseado na lista de defeitos detectados pelo artefato *VisionSystem* que são enviados ao agente *Inspector* e ao agente *Statistic*, tem a função de definir quais as causas prováveis para os defeitos e sugerir ações corretivas para que os defeitos sejam evitados.

Figura 36 – Diagrama de Visão Geral do Agente - *Expert*.



Se houver necessidade, o diagrama de Visão Geral da Habilidade demonstra quais são as sub-habilidades, crenças e planos que fazem parte de uma habilidade. As habilidades são utilizadas para permitir a modularização dos agentes e pode ser vista como uma biblioteca de planos. Numa habilidade são informados planos, crenças ou outras habilidades que são utilizadas pelos agentes aos quais a habilidade está relacionada (pode ser mais de um agente). Este diagrama também é definido com base nos diagramas Normativo, Estrutural, Visão Geral do Sistema e Visão Geral do Ambiente, além de utilizar informações do Diagrama de Visão Geral dos Agentes.

A principal vantagem da modelagem empregando a metodologia Prometheus AEOlus, que segue o *framework* JaCaMo, foi transformar o ambiente complexo de configuração e monitoramento da PPS em uma série de diagramas especializados e simplificados. A partir destes diagramas a fase de implementação pode ser iniciada com maior esclarecimento sobre como deve ser desenvolvido o SMA.

4.5.1.4. Implementação

Essa é a última fase, o objetivo é o refinamento de alguns elementos do sistema para permitir a geração de código, de forma que o código gerado seja o mais completo possível. Por isso, a fase de implementação supõe que os agentes são desenvolvidos utilizando-se o *framework* JaCaMo e o refinamento dos elementos será feito tendo-se

como base esse *framework*. É importante notar que a maneira como a fase de implementaçãao é conduzida pode ser alterada se os agentes forem implementados utilizando-se outra plataforma de agentes.

O refinamento dos elementos consiste, basicamente, no preenchimento dos descritores de cada elemento. Por exemplo, o refinamento do agente *Planner*, apresentado na Tabela 12, visa definir a quantidade de instâncias do agente que são iniciadas quando o sistema for executado. Essa informação é apresentada no campo Cardinalidade. As tabelas descritoras de todos os agentes estão no APÊNDICE A – Descritores dos Agentes.

Para a descrição dos planos, percepções, ações, crenças e mensagens, deve-se levar em conta a sintaxe utilizada para o desenvolvimento de agentes utilizando a plataforma *Jason*. Explicar em detalhes o desenvolvimento utilizando essa plataforma não cabe neste escopo, por isso somente algumas informações básicas, necessárias para o entendimento dos descritores, são apresentadas. Vale lembrar que as variáveis devem iniciar com letras maiúsculas e não devem ser utilizados espaços em branco para descrever as ações, crenças, mensagens ou percepções. Uma descrição detalhada sobre a plataforma de programação de agentes *Jason* pode ser encontrada em (BORDINI et al., 2006).

Tabela 12 – Refinamento do agente *planner*.

Agente *planner*	
Descrição:	este agente recebe o produto a ser produzido da interface com o usuário a partir do início da operação do Sistema Multiagente. Após receber o produto, o agente deverá criar uma lista ordenada dos links entre a máquinas da linha de produção para fabricar o lote do produto. Ele também é responsável pela criação da organização e da interface com o usuário na inicialização da plataforma multiagente.
Cardinalidade:	1 (propõem-se que cada linha possua um agente *Planner*)

O refinamento do plano envolve a definição do contexto no qual esse plano deve ser executado. O contexto determina em qual estado o ambiente deve estar para que o plano possa ser iniciado. Por exemplo, o plano *executeInspection* será executado quando a percepção *receiveDiagnose* for recebida. Na Tabela 13 o plano *createOrganization* é refinado. O contexto é descrito, nesse caso, utilizando-se a sintaxe da plataforma *Jason*. As tabelas descritoras dos demais planos estão no APÊNDICE B – Descritores dos Planos.

Tabela 13 – Refinamento do plano *createOrganization*.

Plano *createOrganization*	
Descrição:	este plano busca criar a organização conforme o arquivo XML que foi construído com base no resultado da modelagem realizada em Prometheus AEOlus. O arquivo XML é construído seguindo o *framework* MOISE.
Contexto:	
Agente:	*Planner*

O plano *receiveProduct* usa como evento *trigger* a percepção *product_received*. Para o refinamento da percepção se deve definir qual a informação que a percepção carrega (Tabela 14). É importante notar que a percepção *product_received* só será utilizada como *trigger* do plano se tiver vindo de um artefato do ambiente chamado `Interface`. Isso evita que o plano seja iniciado caso o agente receba essa informação de outra fonte. As tabelas descritoras de outras percepções estão no APÊNDICE C – Descritores das Percepções.

Tabela 14 – Refinamento da percepção *product_received*.

Percepção *product_received*	
Descrição:	recebe a informação sobre qual a linha que deve ser utilizada para a fabricação do lote e o tamanho do lote. Esta percepção provém da interface com o usuário.
Informação:	[numDaLinha,tamanhoDoLote]
Agente:	*Planner*

Os planos para atender as metas organizacionais executam ações do tipo *goalX*. O refinamento das ações consiste em informar a função que cada ação executa, juntamente com seus parâmetros (Tabela 15). Essa função pode ser interna ou externa, quando consiste na execução de uma operação em um artefato (externa) ou dentro do próprio contexto do agente. As tabelas descritoras de outras ações estão no APÊNDICE D – Descritores das Ações.

Tabela 15 – Refinamento da ação *goalX*.

Ação *goalX*	
Descrição:	várias das ações dos agentes seguem a nomenclatura definida para as metas apresentadas no arquivo XML do *framework* MOISE. Exemplo: <goal id="gX" min="X" ds="receive product to production"/> ▪ g1: receber o produto para produzir ▪ g2: selecionar a lista de máquinas ▪ g3: selecionar e criar a lista de máquinas e links ▪ g4: realizar o setup das máquinas para o produto ▪ g5: carregar os produtos a serem produzidos ▪ g6: acompanhar a produção do lote ▪ g7: final do lote ▪ g8: divulgar as estatísticas do lote ▪ g9: divulger a lista de defeitos do lote
Função:	como especificado no arquivo da organização
Agente:	*Assembler, Configurator, Expert, Inspector, Planner, Statistic*

O agente *Planner* possui uma crença que cria a lista de máquinas *machineList*. O refinamento dessa crença visa definir qual é a informação que cada crença armazena. Isso é feito no campo Informação do descritor, conforme apresentado na Tabela 16. As tabelas descritoras de outras crenças estão no APÊNDICE E – Descritores das Crenças.

Tabela 16 – Refinamento da crença do agente *Planner*.

Crença *machineList*	
Descrição:	lista de máquinas a serem criadas baseadas nos artefatos do ambiente
Informação:	machineList([machine("L1","artifacts.machines.Loader"), machine("PP1","artifacts.machines.PastePrinter"), machine("PaP1","artifacts.machines.PickAndPlaceStation"), machine("U1","artifacts.machines.Unloader"), machine("VS1","artifacts.machines.VisionSystem"), machine("RO1","artifacts.machines.ReflowOven")]).
Agente:	*Planner*

O refinamento das mensagens consiste em informar o propósito e o conteúdo da mensagem enviada. O propósito indica qual a performativa utilizada para envio da mensagem. A performativa fornece

uma indicação do que deve ser feito pelo agente que receber a mensagem, conforme descrito em (BORDINI et al., 2006). Por exemplo, a Tabela 17 demonstra o refinamento da mensagem *prepareMachines* enviada pelo agente *Planner* para o agente *Configurator*. As tabelas descritoras de outras mensagens estão no APÊNDICE F – Descritores das Mensagens.

Tabela 17 – Refinamento da mensagem *prepareMachines*.

Mensagem *prepareMachines*	
Descrição:	mensagem do *Planner* para o *Configurator* no qual ele envia o identificado do produto a produzir, a lista de máquinas e os links entre as máquinas para formar a linha de produção para o lote
Propósito:	*achieve*
Conteúdo:	prepareMachines(idenficadorProduto,listaLinks,listaMáquinas)
Agente:	*Planner*

Por fim, o refinamento dos artefatos do ambiente define quais operações são disponibilizadas pelo artefato para serem utilizadas pelos agentes, bem como as propriedades observáveis e eventos observáveis que são gerados pelo artefato e podem ser 'sentidos' pelos agentes. A Tabela 18 apresenta o refinamento do artefato *Machine* para o Modelo Genérico de Modelagem e Implementação do SMA para configurar e monitorar a PPS. As tabelas descritoras de outros artefatos estão no APÊNDICE G – Descritores dos Artefatos.

Esses descritores foram adaptados do método original do Prometheus para o Prometheus AEOlus.

O método Prometheus AEOlus ainda não conta com uma ferramenta que permita a geração de todo o código fonte com base nos *work products* criados durante o seu uso. Desta forma, a geração de código dos agentes é feita manualmente com base nos diagramas de Visão Geral do Sistema, Visão Geral do Agente e *Capabilities* (quando se aplica) e também baseada nos descritores de Ações, Percepções, Mensagens, Planos, Crenças e Agentes. O código referente ao ambiente é gerado com base no diagrama de Visão Geral do Ambiente, no descritor dos Artefatos e refinamento realizado no diagrama de Classe – do método UML – modelado.

O método Prometheus AEOlus utiliza *work products* originados no Prometheus e, durante a modelagem aqui apresentada, esses aspectos foram apresentados. Uma descrição completa do uso do método

Prometheus AEOlus não está no escopo deste trabalho de pesquisa e para encontrá-la leia Uez e Hübner (2014) ou acesse o site do método[15].

Tabela 18 – Refinamento do artefato *Machine*.

Artefato *Machine*	
Descrição:	Esse artefato é a generalização dos recursos não autônomos da linha de produção. O artefato Machine pode representar qualquer equipamento da linha de produção, a partir dele são especializados cada equipamento. Desta forma, um único artefato é representado na Arquitetura de Referência, já no modelo de implementação o artefato geral é especializado nos equipamentos da linha de produção a ser controlada pelo SMA.
Operações:	init() setup(int batchSize, int productID) run() stop() pause() stateMachine() readyToLoad() notReadyToLoad() isUnloaded() returnFromPause() enableLoad() getProduct(OpFeedbackParam<Product> v) load() unload() operate() pauseOp() stopOp()
Parâmetros:	enum status; int batchSize; int productID; Product currentProduct;
PropriedadesObserváveis:	status
Eventos Observáveis:	

4.5.2. Modelagem dos Recursos do Ambiente com UML

O Diagrama Visão Geral do Ambiente (Figura 29) apresenta o relacionamento dos agentes com um *workspace* Production onde estão dispostos os recursos não autônomos – artefatos e outros recursos – da

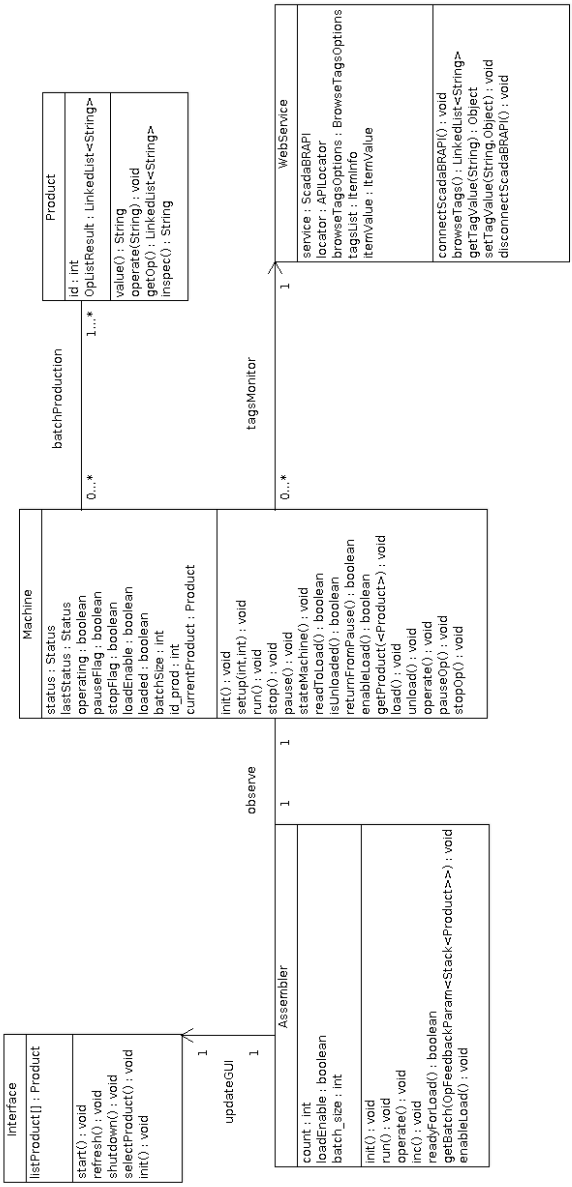
[15] Site do método Prometheus AEOlus: http://www.uez.com.br/aeolus/metodo.html

linha de produção de pequenas séries. O Diagrama Visão Geral do Ambiente não detalha como este *workspace* é modelado – isto seria uma boa contribuição futura para o método Prometheus AEOlus – modelagem dos recursos do ambiente. Por isso, opta-se pela modelagem de um ambiente utilizando o método UML do paradigma orientado a objetos, uma vez que, os artefatos podem ser representados como objetos – Java – que representam os recursos do ambiente.

O objetivo desta modelagem é apresentar um diagrama de classes geral e genérico o suficiente para representar os artefatos e demais recursos de qualquer linha de PPS. A partir deste diagrama de classe geral e genérico, especializa-se o diagrama para atender a uma linha em particular. Mas este processo deve ser simples, então se optou pelo intenso uso das técnicas de modelagem de orientação a objetos para apresentar um diagrama de classes onde os recursos específicos – qualquer um – é resultado da especialização de um recurso geral e genérico. O *workspace* para uma linha PPS específica é montado baseado na sequência dos recursos presentes – primeiro o buffer de entrada que se liga no forno que envia para a máquina de inspeção que descarrega no buffer de saída. Esta linha é 'criada' no SMA durante a etapa de planejamento da produção e a comunicação com os equipamentos reais é definida na criação do ScadaDAQ – que cria o mecanismo de comunicação *Web Service* entre a interface do ScadaDAQ com os equipamentos reais na linha.

O Diagrama de Classes Geral e Genérico apresenta uma classe Machine (Figura 37) que é capaz de representar qualquer recurso não autônomo de uma linha de produção. A partir dele se criam os recursos através da técnica de especialização da classe-mãe (Machine). Fazem parte do ambiente ainda o artefato Interface que é responsável pela interação com a camada superior da Arquitetura de Referência – a Inteface com o Usuário pode ser um aplicativo Java integrado com a plataforma multiagente ou ser a API com um outro aplicativo. Outro artefato presente é o Assembler que auxilia o agente Assembler no cumprimento dos seus objetivos e dois recursos – Product e WebService – o primeiro para representar cada item do lote que está na linha de produção e o segundo é o recurso para realizar a comunicação com o sistema SCADA da camada mais inferior – Camada de Interface com o Ambiente da Produção ou abreviadamente ScadaDAQ.

Figura 37 – Diagrama de Classes Genérico do Ambiente.



4.5.3. Modelagem da Interface com o Ambiente da Produção

A API do ScadaBR é um serviço web (*Web Service*) utilizando a tecnologia SOAP (*Simple Object Access Protocol*)[16]. Isto permite estender o ScadaBR através da arquitetura cliente-servidor, onde o ScadaBR age como um servidor, e um módulo externo (desenvolvimento customizado) atua como cliente (Figura 19).

O ScadaBR e o recurso *Web Service* presente no *workspace* do CArtAgO podem estar executando na mesma máquina, ou em computadores separados, desde que conectados e acessíveis via rede IP.

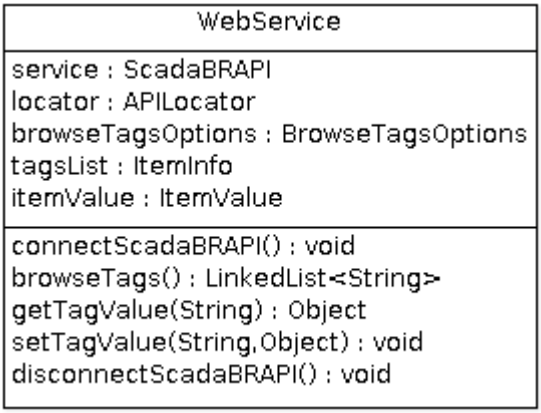
Um recurso *Web Service* (classe Java) foi criado para realizar a comunicação entre o ScadaDAQ – responsável pelo acesso aos equipamentos da linha de produção – e o *workspace* do *framework* CArtAgO que modela o ambiente da linha de produção no contexto do SMA. O recurso *Web Service* executa uma série de etapas necessárias para estabelecer a comunicação e após permitir a leitura e escrita de *tags*. A Figura 38 apresenta o diagrama da classe *Web Service* – usando ArgoUML – que representa o recurso no ambiente. Nela é possível observar métodos responsáveis pela criação do serviço *Web Service* (SOAP); busca da lista de *tags* do ScadaBR; seleção da *tag* desejada; leitura do valor da *tag* selecionada; modificação do valor (escrita) da *tag* selecionada; controle de erros de leitura ou escrita da *tag*.

4.5.4. Implementação do Modelo Genérico de SMA

O interesse de se apresentar um modelo genérico para a implementação do SMA é gerar um guia para a implementação já realizando uma separação dos elementos de composição e gestão da Arquitetura de Referência. O modelo genérico utiliza os elementos conceituais da Arquitetura de Referência, os diagramas da modelagem e os direcionam para elementos implementáveis.

[16] SOAP (*Simple Object Access Protocol*, em português Protocolo Simples de Acesso a Objetos) é um protocolo para troca de informações estruturadas em uma plataforma descentralizada e distribuída. Ele se baseia na Linguagem de Marcação Extensível (XML) para seu formato de mensagem, e normalmente baseia-se em outros protocolos da Camada de aplicação, mais notavelmente em Chamada de Procedimento Remoto (RPC) e Protocolo de Transferência de Hipertexto (HTTP), para negociação e transmissão de mensagens. Fonte: http://pt.wikipedia.org/wiki/SOAP

Figura 38 – Diagrama de classe recurso *Web Service* - por ArgoUML.



4.5.4.1. Agentes

O código fonte de cada agente é descrito em um arquivo de extensão .asl e tem como base o diagrama de Visão Geral do Agente, o diagrama de Visão Geral do Sistema e o descritor de cada elemento do sistema. Os agentes são definidos no *Jason* através de suas crenças iniciais e dos planos que podem executar. Todo plano vai ter um nome, especificado no arquivo do *Jason* pelo caractere @. As crenças iniciais do agente são definidas no diagrama de Visão Geral do Agente e não estão ligadas a nenhum plano. Essas crenças são incluídas no código fonte que define o agente utilizando o campo informação que foi especificado no descritor da crença.

Os planos são formados por três partes: evento `trigger : contexto <- corpo`. O evento *trigger*, no *Jason*, pode ser uma adição de crença ou uma inclusão de objetivo. No Prometheus AEOlus, os eventos *trigger* podem ser uma percepção ou uma mensagem recebida. Quando o evento *trigger* especificado no Prometheus AEOlus for uma percepção, o código gerado no *Jason* deverá ser uma adição de crença. Quando for uma mensagem recebida, duas alternativas são possíveis: se a mensagem utilizar a performativa ACHIEVE – definida no descritor da mensagem - o *trigger* será uma adição de objetivo. Caso contrário, será tratado como adição de crença. Todo plano deve, obrigatoriamente, possuir um evento *trigger*, que especifica quando esse

plano será ativado. O contexto, por sua vez, é definido no descritor do plano mas não é obrigatório.

Para a geração do código de cada agente foram utilizados os diagramas do método que modelam o agente (Visão Geral do Agente, por exemplo) e também os diagramas que presentam a relação do agente com os outros agentes (Visão Geral do Sistema), com o Ambiente (Visão Geral do Ambiente) e com a organização (Missões, por exemplo). Os descritores de refinamento dos agentes, das ações, das percepções, das mensagens, dos artefatos também são importantes ferramentas para auxiliar na codificação do agente (.asl). Estes descritores estão apresentados nos APÊNDICE A – Descritores dos Agentes até APÊNDICE G – Descritores dos Artefatos. Na codificação dos agentes também ocorre a criação do *workspace* dos artefatos.

A implementação dos agentes seguiu fortemente a modelagem, porém onde foi possível, adequações foram feitas com o objetivo de melhorar o modelo de implementação.

4.5.4.2. Ambiente

Para geração do código do ambiente, utiliza-se o diagrama de Visão Geral do Ambiente, o descritor dos Artefatos e o diagrama de Classe da UML. Como pode ser visto no Algoritmo 1, o artefato *Machine* é codificado em linguagem Java. Três pacotes, apresentados nas linhas 3 até 5 do código fonte, que fazem parte do *framework* CArtAgO devem ser importados: `cartago.Artifact`; `cartago.OPERATION` e `cartago.ObsProperty`. O artefato *Machine* é, portanto, uma classe Java que extende o pacote `cartago.Artifact` (linha 11) e define alguns métodos públicos, entre eles: o método `init` e o método `setup`, que definem as operações que podem ser executadas sobre esse artefato. No método `init` são definidas as propriedades observáveis do artefato, que já foram informadas no campo Propriedades do descritor de Artefatos. Para cada uma, um comando `defineObsProperty` é criado. Esse comando exige dois parâmetros: o nome da propriedade e o valor inicial da mesma que, aqui, são definidos com a mesma informação. Já o método `setup`, que define uma das operações a ser executada no artefato, é definido com base no que foi descrito no campo Operações do descritor de Artefatos. No caso, o método `setup` recebe como parâmetros o tamanho do lote a ser produzido e o identificador do produto selecionado para produzir. Com o identificador do produto cada

máquina pode carregar o arquivo de configuração para aquele produto. Defini-se que os *setups* armazenados nas máquinas seguem o `idProduct` o que facilita no carregamento da configuração no equipamento real. Esta informação é enviada do SMA para o sistema SCADA que este acessa a máquina física para carregar o arquivo correto.

Algoritmo 1 – Fragmentos de código do artefato *Machine*.

```
1   package artifacts.machines;
2   import resources.Product;
3   import cartago.Artifact;
4   import cartago.OPERATION;
5   import cartago.ObsProperty;
6   @ARTIFACT_INFO(
7   outports = {
8   @OUTPORT(name = "out-1"),
9   @OUTPORT(name = "in-1") }
10  )
11  public class Machine extends Artifact {
12  public enum Status {
13  STOPPED, IDLE, WAIT, LOADED, READY, PAUSE, DEFECT }
14  ...
15  void init() {
16  status=Status.STOPPED;
17  defineObsProperty("status", "STOPPED"); }
18  ...
19  @OPERATION
20  void setup(int _batchSize,int idp){
21  ObsProperty stats = getObsProperty("status");
22  await_time(1000);
23  batchSize = _batchSize;
24  id_prod = idp;
25  if(status==Status.STOPPED){
26  status=Status.IDLE;
27  stats.updateValue("IDLE");}
28  }
29  ...
```

4.5.4.3. Organização

Para implementação da organização são utilizados os diagramas de objetivos, estrutural, de missões e normativo. Um arquivo .XML é gerado com as informações referentes a organização do SMA. A estrutura principal do arquivo é apresentada no Algoritmo 2. O arquivo é composto de três partes: a especificação estrutural (linhas 4 a 28), a especificação funcional (linhas 29 a 40) e a especificação normativa (linhas 41 a 44).

A especificação estrutural é gerada com base no diagrama Estrutural e apresenta a definição dos papéis que fazem parte do SMA e dos grupos dos quais esses papéis participam. Entre as linhas 5 e 12 se ilustra a geração da tag `<role-definitions>` que especifica os papéis que fazem parte da organização. Papéis que herdam características de outro papel, como o papel *rPlanning* apresentado na linha 8, do papel-pai *rSmallSeriesManager* deve ser informado através da *tag* `extends`.

A especificação dos grupos que compõem a organização envolve definir os papéis que fazem parte do grupo, as ligações entre os papéis, o subgrupos que formam o grupo e as restrições de compatibilidade que existem no grupo. A *tag* `group-specifications` informa o `id` do grupo e a *tag* `roles` define os papéis que farão parte do mesmo. Para cada papel são informadas as cardinalidades mínima (`min`) e máxima (`max`) de agentes que podem adotar aquele papel no grupo. Para que os agentes possam assumir um papel na organização, este papel deve fazer parte de um grupo. A *tag* `links` define as ligações entre os papéis do grupo. Para cada ligação deve ser informado o papel de origem (`from`), o papel de destino (`to`), o tipo de ligação (`type`), o escopo e se a ligação continua sendo válida dentre dos subgrupos. As ligações podem assumir os tipos *authority*, *acquaitance* e *communication* e o escopo pode ser `inter-group` ou `intra-group`. A *tag* `sub-groups` define os subgrupos que fazem parte do grupo e a *tag* `formation-constraints` é usada para especificar as restrições de compatibilidade existentes entre os papéis. Entre as linhas 13 e 27 se ilustra o uso dessas tags.

A especificação funcional é gerada com base nos diagramas de objetivos e de missões. O esquema é definido com base no diagrama de objetivos e é especificado pela *tag* `scheme`. No fragmento apresentado o objetivo `smallSeriesProductionControl` foi refinado nos subobjetivos presentes no diagrama de objetivos. Cada objetivo é informado com uma `id` única que será utilizada para se referir ao objetivo posteriormente no arquivo .asl e uma descrição (`ds`). A *tag* (`plan`) indica que os subobjetivos de um objetivo podem ser atingidos em paralelo (`operator=parallel`), ou se a decomposição do objetivo utilizar o operador *AND* e for informada precedência entre os subobjetivos, estes serão atingidos sequencialmente (`operator=sequence`) e, caso utilizem o operador *OR*, somente um dos planos precisa ser atingido (`operator=choice`).

Após informar os objetivos, tendo como base o diagrama de missões, são informadas as missões que podem ser atribuídas aos papéis. Um exemplo de geração da *tag* `mission` é apresentado entre as linhas 34 e 38. Essa *tag* especifica o `id` da missão e a lista de planos que a compõem. Após as missões serem especificadas, a *tag* `scheme` deve ser finalizada.

Por fim, com base no diagrama normativo, é gerada a especificaçãao normativa. Aqui são especificadas as normas que relacionam os papéis as missões. Para isso é utilizada a *tag* `norm`, onde é informado o tipo da norma (`type`), o papel (`role`) e a missão (`mission`). As normas podem ser do tipo *obligation*, quando o papel tem obrigação de se comprometer com a missão, ou *permission*, quando o papel tem permissão de se comprometer com a missão. Um exemplo de especificação normativa pode ser encontrado entre as linhas 41 e 44 do Algoritmo 2.

Algoritmo 2 – Resumo do arquivo MOISE para organização.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <organisational-specification
3   ...
4   <structural-specification>
5   <role-definitions>
6   <role id="rSmallSeriesManager"/>
7   <role id="rSmallSeriesShopFloor"/>
8   <role id="rPlanning">
9   <extends role="rSmallSeriesManager"/>
10  </role>
11  ...
12  </role-definitions>
13  <group-specification id="grCompany">
14  <links>
15  ...
16  </links>
17  <subgroups>
18  <group-specification id="grManagers" min="1">
19  <roles>
20  <role id="rPlanning" min="1" max="1"/>
21  <role id="rStatistic" min="1" max="1"/>
22  <role id="rExpert" min="1" max="1"/>
23  </roles>
24  </group-specification>
25  ...
26  </subgroups>
27  </group-specification>
28  </structural-specification>
29  <functional-specification>
```

```
30 <scheme id="production">
31 <goal id="smallSeriesProductionControl" min="1">
32 ...
33 </goal>
34 <mission id="leadUp" min="1">
35 <goal id="g1"/>
36 <goal id="g2"/>
37 ...
38 </mission>
39 </scheme>
40 </functional-specification>
41 <normative-specification>
42  <norm id="n1" type="obligation" role="rPlanning" mission="leadUp"/>
43 ...
44 </normative-specification>
45 </organisational-specification>
```

Uma descrição detalhada de como criar o arquivo está em Uez e Hübner (2014).

4.6. CONSTATAÇÕES

Neste capítulo foi apresentada a proposta de uma Arquitetura de Referência para SMA para configuração e monitoramento da PPS. Um Modelo de Referência foi selecionado (JaCaMo) para servir como base para a apresentação da Arquitetura de Referência. A arquitetura proposta cumpre devidamente os Requisitos Desejáveis. Vale ressaltar, que a estratégia de propor a Arquitetura de Referência surgiu da pesquisa das propostas de SMA para manufatura estudadas. Normalmente os sistemas baseados em agentes para a manufatura propõem um solução para cada caso e não apresentam ao desenvolvedor a possibilidade de enquadrar o SMA proposto às suas próprias necessidades a outras linhas de produção.

Outra questão em aberto encontrada estava relacionada à interoperabilidade. Normalmente um grande esforço é dispensado na integração do sistema baseado em agentes com os sistema legados dos equipamentos. A opção pela criação de uma dimensão – Interface com o Ambiente de Produção – transformou o ambiente de comunicação heterogêneo em um ambiente homogêneo onde o SMA e o SCADA utilizam uma tecnologia – *Web Service* – para promover a interoperabilidade.

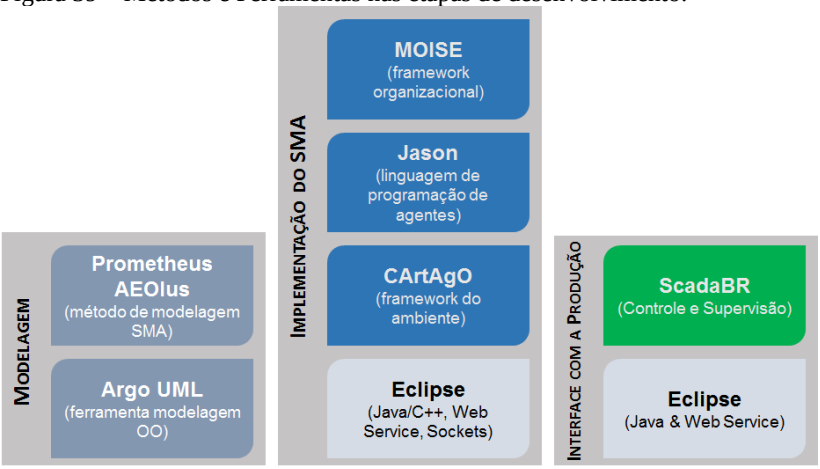
Esta Arquitetura de Referência implementada em um Modelo Genérico de Modelagem e Implementação permitiu ao desenvolvedor de um SMA para PPS customizá-lo para o seu cenário de aplicação.

Destaca-se também que Arquitetura de Referência e os Métodos e as Ferramentas atenderam aos Requisitos Desejáveis (Tabela 19). A Figura 39 posiciona em qual etapa do desenvolvimento cada Método ou Ferramenta contribuiu na proposta do MAS4SSP.

Tabela 19 – Cumprimentos dos Requisitos Desejáveis.

Requisito	Modelo ou ferramenta	Método
Atender a PPS	*Framework* JaCaMo, ScadaBR, *Web Service*	Prometheus AEOlus e UML
Padrões Abertos e Atuais	JaCaMo, ScadaBR, *Web Service* (SOAP/XML)	Prometheus AEOlus e UML
Definição de objetivos dinâmica	*Framework* Moise+	Prometheus AEOlus
Modelar qualquer linha PPS	*Framework* CArtAgO, Java	Prometheus AEOlus e UML
Processo unificado de modelagem e implementação	*Framework* JaCaMo, ScadaBR	Prometheus AEOlus e UML
Distribuído	*Framework* JaCaMo, *Web Service,* ScadaBR	Prometheus AEOlus
Todo o potencial da programação orientada multiagente	*Framework* JaCaMo	Prometheus AEOlus

Figura 39 – Métodos e Ferramentas nas etapas de desenvolvimento.



Contudo, ainda há que se avaliar o custo das customizações que serão necessárias para que o SMA faça realmente o que o desenvolvedor deseja, ou necessite. Algumas customizações são responsabilidade do próprio desenvolvedor – como a criação do sistema no ScadaBR que realize a aquisição e a supervisão dos dados da linha de PPS.

Com esta nova abordagem, as empresas podem ser incentivadas a desenvolver seus SMA para configuração e monitoramento da PPS pois existe agora uma Arquitetura de Referência a partir da qual o desenvolvedor foque seus esforços nas customizações para o seu cenário.

De forma complementar, pode-se afirmar que este capítulo apresentou uma justificativa da utilização de SMA para atender aos Requisitos Desejáveis para um sistema de configuração e monitoramento da PPS, aliado com as características e vantagens da programação orientada a multiagente.

Como continuidade deste trabalho de pesquisa e como uma das formas de avaliação da abordagem, a seguir é apresentada a implementação de uma instância da Arquitetura de Referência para um SMA para configuração e monitoramento da PPS no contexto de uma linha de produção de Placas de Circuito Impresso (PCI) com as características de uma PPS.

5. INSTANCIAÇÃO DA ABORDAGEM - EXPERIMENTO

Uma das formas de verificação da abordagem proposta neste trabalho de pesquisa é a implementação de um protótipo baseado na Arquitetura de Referência. Busca-se com isto demonstrar a viabilidade técnica da sua utilização para a criação de um SMA para configuração e monitoramento da PPS que esteja em consonância com os Requisitos Desejáveis. O objetivo do experimento é avaliar se a abordagem proposta é uma solução técnica viável. Não foi o objetivo deste trabalho de pesquisa fazer uma comparação com outras abordagens ou realizar um levantamento científico de indicadores de desempenho. Estes dois temas são sugeridos como trabalhos futuros desta pesquisa.

O protótipo aqui apresentado toma como referência uma linha de PPS de Placas de Circuito Impresso. É relevante citar que aqui é implementada uma instância simulada para verificação e testes. Contudo, ela é representativa porque segue todos os requisitos definidos pela proposta. Se outras instâncias forem implementadas, mesmo que utilizando cenários de implementações diferentes, mas seguindo as definições da proposta, estima-se que elas também serão viáveis tecnicamente.

Este capítulo inicia com a apresentação do cenário do experimento. Após, a implementação do SMA é apresentada tendo como base a Arquitetura de Referência apresentada no capítulo anterior. Aqui serão apresentadas as instanciações – do Modelo Genérico de Modelagem e Implementação – e as customizações necessárias. Alguns trechos do código fonte que merecem destaque são também apresentados. O detalhamento da modelagem e da implementação estão nos Apêndices deste documento.

5.1. DELIMITAÇÃO DO EXPERIMENTO

O protótipo do SMA para configuração e monitoramento da PPS utilizou como cenário de aplicação o Laboratório-Fábrica LabElectron da Fundação CERTI na cidade de Florianópolis/SC (Figura 40). O mercado do LabElectron é formado principalmente por empresas cujos pedidos são pequenos e ocorre uma grande variação nos modelos de produtos a serem fabricados (normalmente em cada novo lote é uma nova versão do produto a ser produzida). O sistema de produção do LabElectron é caracterizado como uma PPS. Neste contexto, o LabElectron busca constantemente por tecnologias apropriadas ao seu

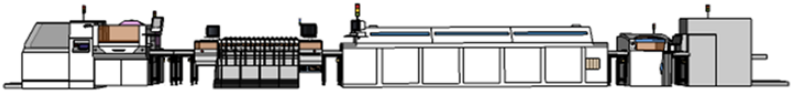
cenário de produção e vários trabalhos já foram desenvolvidos para melhoria do processo e da qualidade na produção de PCIs (DORO, 2009; STEMMER *et al.*, 2013; STEMMER *et al.*, 2014; VARGAS, 2012).

Figura 40 – Linha de produção do LabElectron.

Fonte: www.labelectron.org.br (Acesso em: 02/05/2014)

A linha de montagem do LabElectron foi simplificada para a representada na Figura 41 onde o número de equipamentos foi reduzido. Contudo, os equipamentos fundamentais da linha de produção foram mantidos (Tabela 84). O objetivo de simplificar a linha foi reduzir redundância e facilitar a compreensão do processo de produção. Também no experimento, foram utilizados 4 (quatro) tipos de produtos para simular a produção.

Figura 41 – Linha SMT utilizada no experimento.

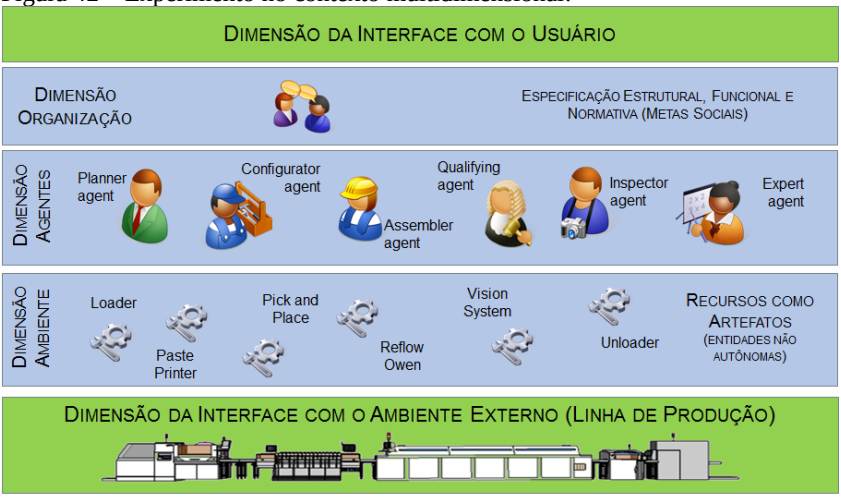


O ciclo de operação da linha de produção é o seguinte: 1) no início do turno de trabalho o responsável pela linha recebe as ordens de produção do turno em conjunto com um planejamento da produção; 2) o operador avalia o planejamento que considera o tempo de fabricação, o tamanho do lote, a similaridade entre os componentes, a disponibilidade de matéria-prima; 3) o operador define o lote a ser produzido; 4) com o tipo da placa ele busca as placas ‘nuas’ no estoque e carrega a *Loader* com *n* unidades; 5) em seguida realiza o *setup* da SPI, da Insersora, do Forno, da AOI e prepara o magazine da *Unloader* para receber as placas acabadas; 6) os equipamentos após o *setup* passam para um estado de

READY; 7) o operador inicia a produção (*RUN*); 8) o operador monitora os equipamentos, na ocorrência de alguma falha, a linha pára, o equipamento com falha passa para o estado *FAIL* e os demais permanecem em um estado de pausa (*PAUSE*) até a falha ser solucionada; 9) continua a produção até a finalização do lote; 10) as placas finalizadas são retiradas da *Unloader* e enviadas para o setor de testes e posterior expedição. O ciclo se inicia com o próximo lote.

Compreendido o ciclo de operação e coletado as informações pertinentes sobre os equipamentos da linha de produção, passa-se a fase de revisão da modelagem e customização de aspectos peculiares ao experimento. Após a etapa de revisão da modelagem se passa a fase de customização da implementação. Assim, a partir do Modelo Genérico de Modelagem e Implementação apresentado no capítulo anterior, as customizações necessárias para atender ao experimento selecionado são realizadas tanto a nível do SMA, como do ScadaBR e na definição da interface com o usuário (Figura 42).

Figura 42 – Experimento no contexto multidimensional.



5.2. TECNOLOGIAS DE IMPLEMENTAÇÃO

A partir da delimitação do experimento, retorna-se ao Modelo Genérico de Modelagem e Implementação que é customizado.

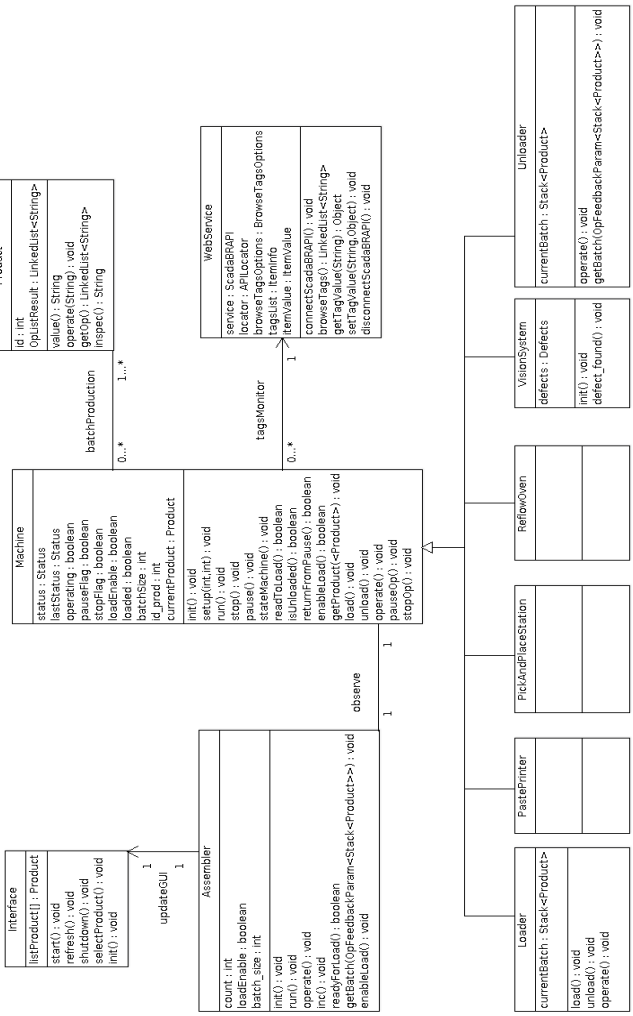
5.2.1. Revisão da Modelagem

A revisão da modelagem do SMA do modelo genérico sugere que os artefatos do ambiente do experimento devem ser criados baseados no artefato genérico e geral. A partir do Diagrama de Classes Geral e Genérico do Ambiente (Figura 37) foram derivados os artefatos que representam a linha de PPS de placas de circuito impresso do experimento (Figura 41), resultando no diagrama da Figura 43. Os demais artefatos (`Interface` e `Assembler`) e recursos (`Product` e `WebService`) mantêm-se como os do Modelo Genérico de Modelagem e Implementação.

A Figura 43 apresenta os artefatos derivados da classe do Modelo Genérico de Modelagem e Implementação – `Machine` – que se torna a classe-mãe. É importante destacar que apenas os artefatos `Loader` e `UnLoader` tiveram alguma customização de implementação. O motivo é a adequação a como é realizado o carregamento/descarregamento dos produtos na linha de produção. Neste experimento, as máquinas `Loader` e `UnLoader` possuem magazines[17] com capacidade de receber as placas nuas e acabadas do lote – não é carregado um item do lote no equipamento, como modelado no artefato *Machine* e sim um lote inteiro. Ou seja, a produção somente inicia quando o lote completo de placas nuas está na `Loader` e finaliza quando todas as placas acabadas estiverem na `UnLoader`. Por isso, tanto o artefato `Loader` como o artefato `UnLoader` tiveram uma reimplementação dos métodos de `load()`, `unload()` e `operate()` no artefato `Loader` e dos métodos `operate()` e `getBatch()` no artefato `UnLoader` para adequação à linha de PCIs. Devido a forma de implementação do modelo genérico do artefato `Machine` os demais artefatos não necessitam de qualquer customização dos seus métodos e/ou propriedades observáveis.

[17] recipiente da máquina onde são armazenados componentes ou ferramentas em canais apropriados para serem utilizados durante a operação do equipamento.

Figura 43 – Diagrama de classe dos artefatos do experimento.



Outra customização relevante é apresentada no Algoritmo 3 que é um fragmento do código fonte do artefato `Loader` do seu método `unload()` que foi reimplementado para monitorar quando todos os

itens do lote foram enviados para a próxima máquina, quando isto ocorre, o status da máquina volta para o estado `WAIT` que significa que está configurada mas sem produtos carregados, senão, o estado é `LOAD` indicando que ainda possui produtos no *magazine*. Esta customização (linhas 6 a 13) foi necessária pois a `Loader` necessita informar a linha que ela já cumpriu sua atividade neste lote.

Algoritmo 3 – Fragmentos de código do artefato *Loader*.

```
1   void unload(){
2       ObsProperty stats = getObsProperty("status");
3       try{
4               execLinkedOp("out-1","enableLoad");
5               await("isUnloaded");
6               if(!currentBatch.isEmpty()){
7                       status=Status.LOADED;
8                       stats.updateValue("LOADED");
9               }else{
10                      log("Loader empty!!!");
11                      status=Status.WAIT;
12                      stats.updateValue("WAIT");
13              }
14      } catch (Exception ex){
15              log("{operate}@ call link enableLoad");
16      }
17  }
```

Já o Algoritmo 4 destaca o método `operate()` do artefato `UnLoader` que também deve monitorar se todos os itens do lote já chegaram na máquina `UnLoader`. Quando todos os itens do lote estão na `UnLoader` esta passa para o estado de `WAIT` para informar a linha que a produção do lote chegou ao fim (linhas 7 a 13).

Algoritmo 4 – Fragmentos de código do artefato *Unloader*.

```
1   void operate(){
2       ObsProperty stats = getObsProperty("status");
3       await_time(50);
4       currentBatch.add(currentProduct);
5       log("batch size="+currentBatch.size());
6       loaded=false;
7       if(currentBatch.size()==batchSize){
8               status=Status.READY;
9               stats.updateValue("READY");
10      }else{
11              status=Status.WAIT;
12              stats.updateValue("WAIT");
13      }
14  }
```

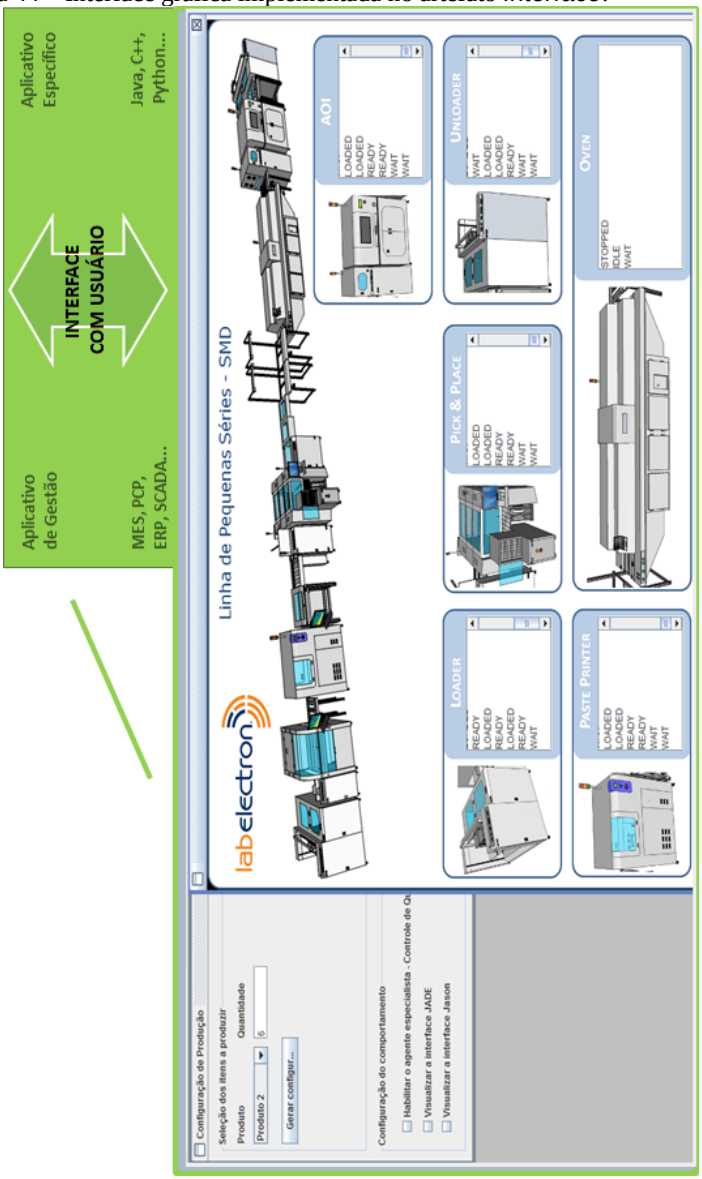
Para este experimento, nenhuma outra customização relevante do ponto de vista da Arquitetura de Referência foi necessária. Pequenas customizações aconteceram no modelo de implementação que foi proposto no capítulo anterior no momento de codificação. Neste experimento existe apenas um sistema de inspeção da qualidade e por isso a cardinalidade do Diagrama Estrutural (Figura 26) permanece inalterada, contudo, havendo mais de um sistema de inspeção, a cardinalidade deverá ser alterada.

5.2.2. Codificação

A codificação se iniciou partindo do Modelo Genérico de Modelagem e Implementação e as customizações no código ocorreram a nível de agentes e do ambiente – principalmente na customização das *tags* que são recebidas e enviadas pelo ScadaBR – na dimensão de Interface com o Ambiente da Produção. Para o projeto do SCADA foi utilizada uma cópia do sistema SCADA da linha de produção do LabElectron – que sofreu customizações para o experimento. O sistema ScadaBR que supervisiona a linha de produção do LabElectron atualmente recebe informações sobre o estado dos equipamentos (`STOPPED`, `IDLE`, `WAIT`, `LOADED`, `READY`, `PAUSE`, `DEFECT`) e ainda não há possibilidade de atuar nos equipamentos, as alterações são informadas aos operadores da linha que as executam manualmente no SW ou HW de cada equipamento. Neste experimento, o sistema ScadaBR projetado insere a possibilidade de alteração automática dos equipamentos da linha, desta forma o ScadaBR realmente configura e monitora a linha de produção de forma automática e com o SMA de forma autônoma.

O artefato *Interface*, este sim, teve um esforço de desenvolvimento de uma interface de usuário que atendesse ao experimento do LabElectron. Uma interface simples porém moderna em Java foi desenvolvida (Figura 44). A interface é composta por várias abas desde uma de apresentação da linha, passando por uma de configuração, e uma aba de operação onde estão os equipamentos da linha se deseja monitorar com os seus parâmetros. Na Figura 44 são apresentados a Printer (`PastePrinter`), a Insersora (`PickAndPlace`), o Forno (`ReflowOven`) e a AOI (`VisionSystem`) que são a forma como os equipamentos – artefatos – são conhecidos no Labelectron.

Figura 44 – Interface gráfica implementada no artefato *Interface*.



Dois recursos que foram modelos provenientes da Arquitetura de Referência também estão presentes – *Product* e *WebService*. O recurso *Product* (Algoritmo 5) é um objeto Java que representa cada item do lote na linha de produção, este objeto armazena o resultado de cada operação dos artefatos *Machine*. E também armazena o resultado da inspeção que ocorrendo algum defeito (linhas 8 a 15), este fica registrado no item do lote até o final da produção do lote. O objeto também permite que seja rastreado por quais artefatos o produto passou, já que cada operação (`operate()`) (linhas 2 a 4) de cada artefato é armazenada (linha 5 a 7). Desta forma, é possível rastrear em tempo de operação cada item do lote.

Algoritmo 5 – Fragmento do recurso *Product.*

```
1   ...
2   public void operate(String res){
3       OpListResult.add(res);
4   }
5   public LinkedList<String> getOp(){
6       return this.OpListResult;
7   }
8   public String inspec(){
9       String ret=new String();
10      Iterator<String> itr=this.OpListResult.iterator();
11      while(itr.hasNext()){
12              ret=ret+itr.next()+"|";
13      }
14      return ret;
15  }
```

Já o recurso *WebService* é responsável neste experimento pela comunicação com o sistema ScadaBR. O Algoritmo 6 apresenta fragmentos do processo de instanciação do serviço pelo recurso *WebService* com a interface de programação (API) do ScadaBR (linhas 2 e 3), buscam-se as *tags*[18] disponíveis no ScadaBR (linhas 5 e 7), nas linhas 9 a 14 é configurado o mecanismo de leitura de uma determinada *tag* – neste caso o sensor de entrada do forno, já entre as linhas 16 e 29 é descrito o processo de escrita de um novo valor para uma *tag* – neste caso a temperatura do forno é alterada na zona 1 para 123.00 graus Celsius (linha 24).

[18] Definir TAG

Algoritmo 6 – Fragmento do recurso *WebService.*

```
1   ...
2   service = (APISoapBindingStub) locator.getAPI();
3   System.out.println("recebeu o serviço");
4   ...
5    BrowseTagsOptions browseTagsOptions = new BrowseTagsOptions();
6   ...
7    browseTagsResponse = service.browseTags(browseTagsParams);
8   ...
9   ReadDataOptions readDataOptions = new ReadDataOptions();
10  ReadDataParams readDataParams = new ReadDataParams();
11  readDataParams.setOptions(readDataOptions);
12  String[] itemPathList = {"Machines.Oven.Oven_InputSensor"};
13  readDataParams.setItemPathList(itemPathList);
14   ReadDataResponse readDataResponse = new ReadDataResponse();
15  ...
16  WriteDataOptions writeDataOptions = new WriteDataOptions();
17  writeDataOptions.setReturnItemValues(false);
18  ItemValue itemValue = new ItemValue();
19  String pathWriteData = "Machines.Oven.Oven_TemperatureZone1";
20  itemValue.setItemName(pathWriteData);
21  itemValue.setTimestamp(Calendar.getInstance());
22  itemValue.setQuality(QualityCode.GOOD);
23  itemValue.setDataType(DataType.FLOAT);
24  Object novoValor = 123.00;
25  itemValue.setValue(novoValor );
26  ItemValue[] itemValueList = new ItemValue[1];
27  itemValueList[0] = itemValue;
28  WriteDataParams writeDataParams = new WriteDataParams();
29  writeDataParams.setItemsList(itemValueList);
30  ...
```

As implementações que são utilizadas para fins de interfaceamento entre a *framework* CArtAgO e o ScadaBR, nesta instância de implementação, foram projetadas para serem menos intrusivas possível.

Constata-se que assim como a implementação dos agentes ficou limitada a lógica multiagente, a implementação dos artefatos e recursos do ambiente ficou limitada a esta dimensão. O foco de cada implementação está na definição do artefato em termos de operações, propriedades e eventos observáveis. Os diagramas de Visão Geral do Ambiente e UML e o Descritor do Ambiente facilitam também a codificação. As vantagens desta abordagem no que tange a dimensão do ambiente são: foco na implementação de artefatos e recursos; especialização das funções; criação de uma solução flexível – pode

atender a vários cenários e dinâmica – alterações no ambiente podem ser realizadas em tempo de operação.

5.2.2.1. Organização

O arquivo XML, resultande da modelagem em Prometheus AEOlus e sua implementação seguindo o framework MOISE, permaneceu inalterado. As metas organizacionais são as mesmas das propostas no Modelo Genérico de Modelagem e Implementação.

5.2.2.2. ScadaBR – Interface de Configuração e Monitoramento do Processo

O ScadaBR permite que além de ler as configurações e estados dos sensores/atuadores é possível alterá-los por meio da interface do ScadaBR (Figura 45). A Figura 45 apresenta uma visão geral da linha de produção de PCI estudada e também uma área reservada para cada um dos equipamentos do ambiente de produção. Cada um dos equipamentos recebeu um nome de acordo com a sua função e um estudo detalhado do seu funcionamento e da interface de comunicação com cada equipamento foi realizado. Deste estudo, foram selecionados os sensores, atuadores e arquivos de configuração necessários para o *setup* e operação de cada equipamento. A Tabela **20** apresenta cada equipamento com o seu nome fictício, o fabricante e modelo do real na linha e uma descrição das entradas e saídas supervisionadas e controladas (*tags*) pelo ScadaBR DAQ. Basicamente, todo equipamento da linha possui um sensor que indica a presença de um item do lote na entrada (*input sensor*), outro sensor que indica que o equipamento está realizando uma operação sobre o item do lote (*process sensor*) e um sensor que indica que o item do lote está na saída do equipamento (*output sensor*). Há também um indicador do estado do equipamento (`STOPPED`, `IDLE`, `WAIT`, `LOADED`, `READY`, `PAUSE`, `DEFECT`) e o nome do arquivo de configuração do equipamento (*config file*). Normalmente este aquivo possui o nome do produto a ser produzido, criando desta forma o relacionamento entre o nome do produto e qual *setup* deve ser carregado. Na primeira vez que o produto será produzido o operador baseado na experiência seleciona uma configuração de um produto que possui as características semelhantes e posteriormente realiza os ajustes para o novo produto – salvando a configuração do equipamento com o nome do produto. Doro (2009) criou um sistema automatizado para auxiliar nesta operação.

Figura 45 – Interface gráfica do ScadaBR DAQ

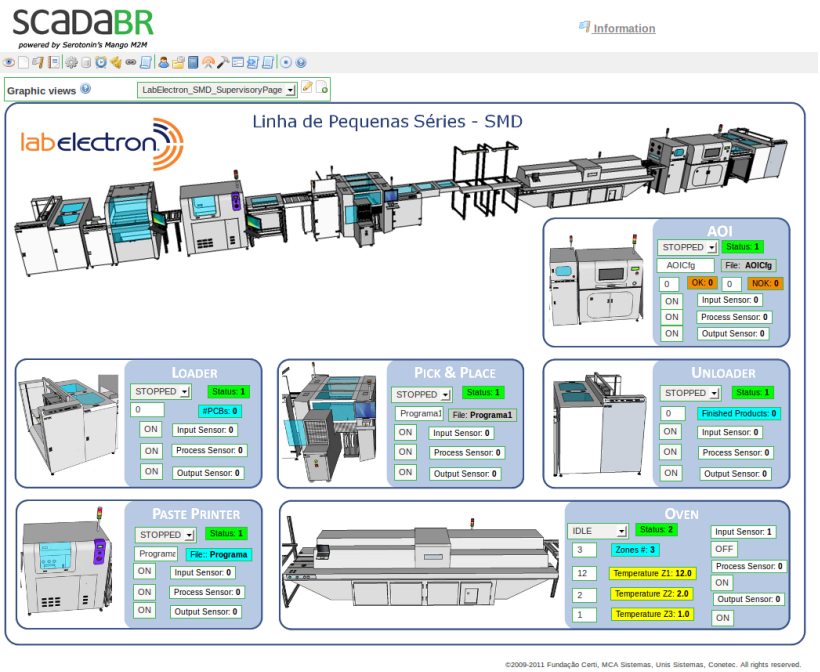


Tabela 20 – Linha de produção simulada do Labelectron.

Nome	Fab./Modelo	Variáveis monitoradas (tags)
Loader	NUTEK NTE 2200LX	Status; Qtde. de PCI no magazine – placas nuas; Sensores: entrada, processo, saída.
Paste Printer (SPI)	TRI TR7066	Status; Sensores: entrada, processo, saída. Arquivo de configuração da máquina.
Pick&Place (insersora)	SIEMENS SiPlace SX1	Status; Sensores: entrada, processo, saída. Arquivo de configuração da máquina.
Oven (Forno)	ERSA HOT FLOW 2/12	Status; Sensores: entrada, processo, saída. Quantidade de zonas de temperatura ativas (até 3); Temperatura de cada zona ativa.
AOI	TRI TR7500	Status; Sensores: entrada, processo, saída. Arquivo de configuração da máquina. Resultado da inspeção – OK/NOK; Quantidade de aprovados e reprovados.
Unloader	NUTEK NTE 2200LX	Status; Qtde. de PCI no magazine - finalizados; Sensores: entrada, processo, saída.

5.3. ANÁLISE DO EXPERIMENTO

Os resultados obtidos com a implantação do *MAS4SSP* no simulador do LabElectron mostraram que todos os elementos contidos na Arquitetura de Referência foram aplicados na prática. A proposta de um Modelo Genérico de Modelagem e Implementação baseado na Arquitetura de Referência para guiar o desenvolvimento e acumular *expertise* – a fim de serem aplicadas em outras linhas PPS – foi validada tecnicamente.

A operação do experimento também se mostrou amigável e fácil do ponto de vista do operador. A simulação acontece em 4 etapas, onde na etapa 1 o sistema ScadaBR que simula as variáveis (tags) do processo é iniciado (Figura 46); na etapa 2 o operador acessa a interface com o usuário onde seleciona o tipo do produto (entre 4 tipos de produtos) e o tamanho do lote (Figura 47); na etapa 3 o SMA é iniciado e realiza a configuração e monitoramento da linha de produção observando a evolução dos estados dos equipamentos (Figura 48); a evolução da produção do lote é apresentada na interface com o usuário através da visualização dos estados dos equipamentos, até o final do lote (Figura 49).

Figura 46 – Experimento Labelectron – Etapa 1

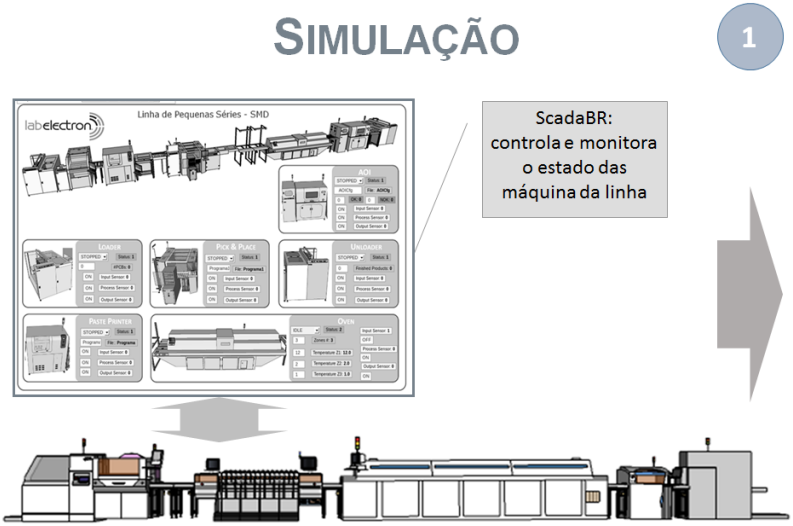


Figura 47 – Experimento Labelectron – Etapa 2

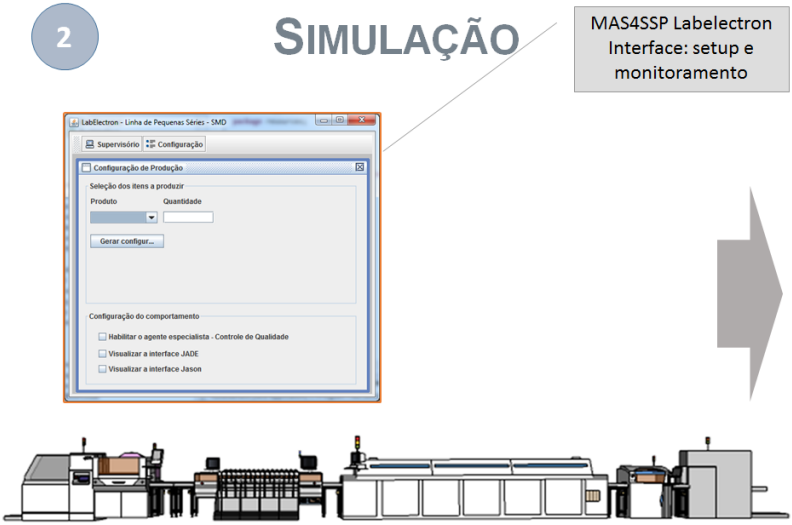


Figura 48 – Experimento Labelectron – Etapa 3

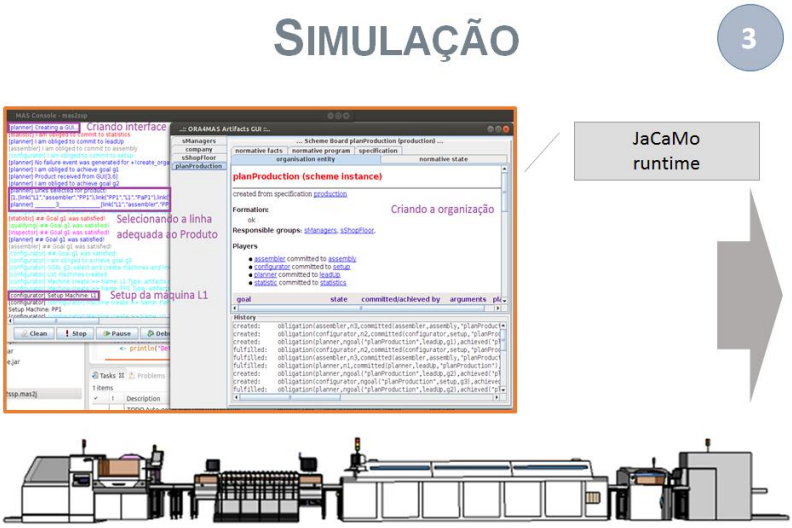
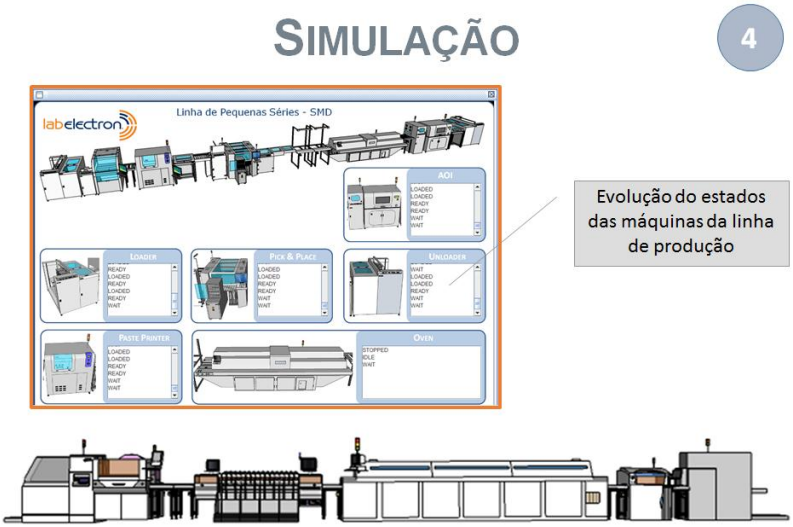


Figura 49 – Experimento Labelectron – Etapa 4



Para a PPS de montagem de placas eletrônicas o uso do *MAS4SSP* mostrou ser uma ferramenta útil e viável tecnicamente para a configuração e o monitoramento da produção. Os resultados mostraram que embora a implantação do *MAS4SSP* no LABelectron aconteceu de forma simulada, ele se mostra uma alternativa para o objetivo traçado pelo Labelectron para que na sua linha de produção - *o planejamento da produção, o setup, o controle de qualidade (AOI), o sistema de rastreabilidade e outros sejam integrados a um sistema automático de configuração e monitoramento da produção*. A capacidade do *MAS4SSP* de atuar autonomamente oferece à empresa a oportunidade de passar para um estágio mais avançado de supervisão da produção na qual será capaz de atuar de forma preventiva e atingir a qualidade desejada desde a primeira vez.

Em termos gerais, o experimento do *MAS4SSP* foi considerado válido pela engenharia da empresa como uma alternativa para configurar e monitorar a produção. O experimento do *MAS4SSP* teve suporte do LabElectron, através da disponibilização de informações do processo que permitiram planejar e definir melhor a estratégia de implementação.

De uma forma geral, implantando o *MAS4SSP*, a empresa possuirá a disposição informações das causas de defeitos no produto

oriundas do processo e das ações necessárias para evitá-las, tudo isto de forma autônoma no contexto do SMA. Com o objetivo de realizar uma comparação superficial entre o tempo de configuração e o tempo de monitoramento dos equipamentos, é apresentada a Tabela 21.

Tabela 21 – Comparação entre os tempos simulados e da linha de produção.

Objetivo	Tempo do LabElectron	Tempo do experimento
Tempo de *setup* da Loader	+/- 2 min	< 1 s
Tempo de *setup* da SPI	+/- 1 min	< 1 s
Tempo de *setup* da Insersora	+/- 1 min	< 1 s
Tempo de *setup* do Forno	+/- 2 min	< 1 s
Tempo de *setup* da AOI	+/- 5 min	< 1 s
Tempo de *setup* da Unloader	+/- 2 min	< 1 s
Tempo de setup completo da linha	+/- 12 min	< 10 s
Tempo para monitorar o estado equip.	+/- 2 min	< 1 s

A viabilidade técnica demonstrada neste experimento é importante para justificar a Arquitetura de Referência proposta em um cenário industrial de pequenas séries. Percebe-se que para outras aplicações serão necessárias customizações do Modelo Genérico de Modelagem e Implementação – nas mesmas características do experimento, mas que o esforço necessário será menor que do projeto, modelagem e implementação de um SMA desde o princípio. Percebe-se também um ganho no tempo de desenvolvimento comparado ao método seguindo o paradigma orientando a agentes – onde o número de agentes tende a ser maior devido a agentificação de todos os recursos (autônomos e não-autônomos). Constatação esta da equipe de desenvolvedores que foram entrevistados – próxima seção.

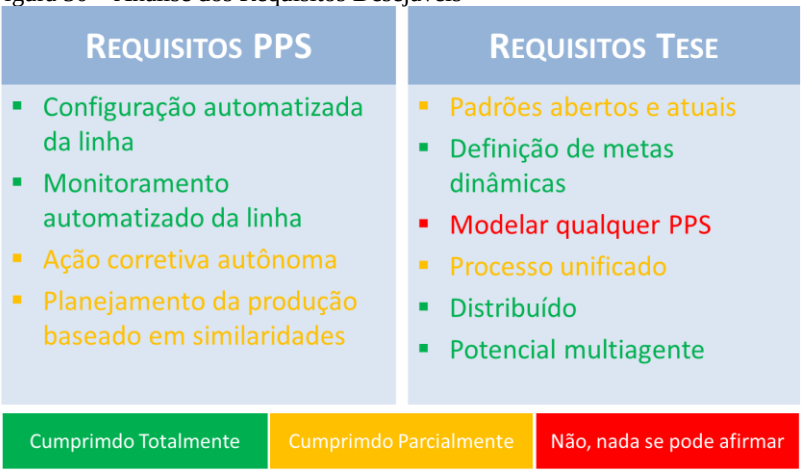
Do ponto de vista do trabalho de pesquisa a Tabela 22 apresenta algumas considerações sobre o cumprimento ou não dos Requisitos Desejáveis propostos para a Arquitetura de Referência após o experimento.

Observando-se os Requisitos Desejáveis e realizando uma comparação com os resultados do experimento se percebe que a abordagem é viável tecnicamente, pois atende satisfatoriamente aos Requisitos Desejáveis como se observa na classificação realizada na Figura 50.

Tabela 22 – Considerações sobre os Requisitos Desejáveis.

Requsito	Atendimento	Considerações
Atender a PPS	Parcial	O experimento mostrou que SMA proposto atende à necessidade de realizar vários *setups* para cada novo lote a ser produzido. Também mostra que alterações na planta ou no processo são rapidamente absorvidas pelo SMA – alterando a linha de produção – artefatos – ou alterando as metas de produção – organização.
Padrões Abertos e Atuais	Sim, com ressalvas	As ferramentas utilizadas pela Arquitetura de Referência são todas atuais, ativas e em desenvolvimento. Mas existe um desafio, as ferramentas utilizadas estão em desenvolvimento, são utilizadas por comunidades restritas e especializadas e também ainda não há uma padronização internacional. Exemplo: a modelagem de SMA e o padrão *Web Service*.
Definição de objetivos dinâmica	Sim	Como o emprego do framework Moise – dimensão organização – as metas, a estrutura, as missões e normas de relacionamento entre os agentes podem ser alteradas em tempo de execução.
Modelar qualquer linha PPS	Nada se pode afirmar	O experimento tratou de uma linha de PCI que foi bem modelada a partir do Modelo Genérico de Modelagem e Implementação, mas não foram realizados outros experimentos para um conclusão.
Processo unificado de modelagem e implementação	Parcial	Houve avanço, no início do trabalho de pesquisa não havia um método para o Modelo de Referência, hoje existe. Contudo, ainda é necessário uma ferramenta funcional *start to end* de modelagem e também seria interessante a integração do diagrama de visão geral do ambiente com diagramas de classe e casos de uso do método UML.
Distribuído	Sim	O SMA e o SCADA são inerentes aos sistemas distribuídos.
Todo o potencial da programação orientada multiagente	Sim, para este caso	Também foi atendido pela Arquitetura de Referência que foi implementada no experimento baseado no modelo genérico. Contudo, em uma linha de produção maior, mais complexa, ou uma integração bottom-up da produção a dimensão de Interação (com seus protocolos e ontologias) será necessária.

Figura 50 – Análise dos Requisitos Desejáveis



Dois pontos frágeis merecem destaque: a ausência de padrões tanto dos SMA como *Web Service*; as ferramentas de SMA ainda estão em desenvolvimento e restritas à comunidades especializadas – normalmente da área de computação – o que para a indústria é uma fragilidade. Por outro lado, percebe-se pontos fortes, do ponto de vista da PPS: o SMA é um forte candidato como solução técnica para os desafios deste modelo de produção; as ferramentas para modelagem e implementação seguindo o paradigma orientado a multiagente estão tendo um grande avanço nos últimos anos, estima-se que uma ferramenta *start-to-end* com todas as dimensões está próxima; o emprego do ScadaBR integrado com o ambiente via *Web Service* contribui fortemente para a adoção desta abordagem; a implementação do SMA se torna mais clara, especializada e eficiente com o uso de todo o potencial dos agentes, do ambiente, da organização e da interação.

6. CONCLUSÃO E PERSPECTIVAS

Este capítulo está estruturado em 3 seções principais que iniciam com uma análise geral sobre a nova abordagem, em seguida se recuperam os objetivos geral e específicos e uma análise sobre o cumprimento destes objetivos é realizada. Por fim, o capítulo finaliza com as considerações finais sobre este trabalho de pesquisa passando pelas suas limitações e sugestões de trabalhos futuros.

6.1. ANÁLISE GERAL DA PROPOSTA

Com o objetivo de analisar a proposta de Arquitetura de Referência para SMA para configuração e monitoramento da PPS, esta seção apresenta dois procedimentos metodológicos aplicados, cada qual com sua importância específica:

- Entrevistas com os desenvolvedores: para isso foi apresentada a Arquitetura de Referência para as pessoas que atuaram como desenvolvedores diretos ou indiretos da abordagem proposta. Este grupo foi formado pelos bolsistas do grupo de pesquisa que atuaram na implementação do experimento do Labelectron e também atuaram na implementação de um sistema baseado em agentes na Alemanha.
- Publicação de artigos em eventos científicos: para avaliar o potencial da abordagem proposta durante o percurso e no final do trabalho de pesquisa.

Além disso, outro procedimento metodológico foi o experimento do Labelectron do capítulo anterior, no qual a partir da execução de uma instância de implementação criada a partir do Modelo de Referência e da Arquitetura de Referência, foram efetuados testes de implementação e do experimento foram extraídos dados da sua execução com o intuito de validar tecnicamente a proposta e o cumprimento dos Requisitos Desejáveis apresentados no capítulo anterior.

6.1.1. Questionário

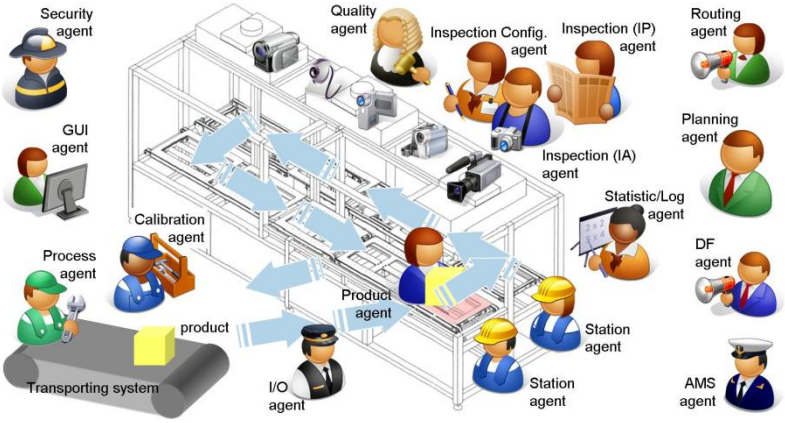
Realizar perguntas às pessoas sobre o que pensam sobre determinado assunto é uma forma de conduzir estudos empíricos de observação, experimento e *survey* (avaliação). A qualidade dos dados extraídos desse método de estudo pode trazer vantagens para o objeto de estudo de uma pesquisa.

A aplicação do questionário neste trabalho de pesquisa teve o objetivo de auxiliar na comprovação dos objetivos gerais e específicos.

6.1.1.1. Contexto do Questionário

O BRAGECRIM/COGMET 013/2010 projeto entre o DAS/UFSC e o instituto WZL/RWTH-Aachen teve o objetivo de *desenvolver uma nova geração de sistemas metrológicos de produção e de garantia de qualidade, com um maior grau de percepção cognitiva sobre o produto e o processo*. Na Alemanha o projeto implementou um sistema baseado em agentes (Figura 51) – paradigma oirentado a agentes – baseado na plataforma JADE e os agentes implementados em JAVA. O cenário de aplicação foi uma célula de inspeção de faróis automotivos (Figura 52).

Figura 51 – Esquemático do sistema agentificado do experimento alemão.



No Brasil, o experimento foi o descrito no capítulo anterior. O projeto foi financiado pela CAPES e CNPq no Brasil e pelo DFG na Alemanha entre 2010 e 2013

Figura 52 – Foto da linha de inspeção de faróis do WZL da RWTH-Aachen.

É importante salientar que o questionário foi aplicado com a equipe de desenvolvedores envolvidos no projeto BRAGECRIM/COGMET que participaram dos dois experimentos. Dessa forma, o questionário não serve diretamente como instrumento de comprovação científica/instrumento concreto, mas serve de auxílio de análise subjetiva do assunto.

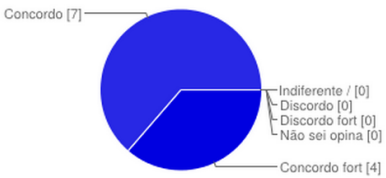
O questionário aplicado aos desenvolvedores objetivou apresentar a proposta MAS4SSP de forma completa e também recuperar aspectos relevantes do SMA implementado no parceiro da Alemanha seguindo o paradigma orientado a agentes. A apresentação teve um enfoque técnico e foi direcionada para apresentar ocorreu a implementação do experimento do MAS4SSP e também como foi implementada a solução na Alemanha. Todos os entrevistados tiveram atuação direta ou indireta no desenvolvimento dos dois experimentos. Das 12 (doze) pessoas que participaram do desenvolvimento – em algum período entre 2010 e 2013, 11 (onze) responderam o questionário com as questões apresentadas na Tabela 23.

As duas primeiras perguntas do questionário tem o objetivo de levantar o posicionamento da equipe quanto a contribuição do SMA de uma forma geral para a PPS. O Gráfico 1 é referente as respostas em relação a contribuição do SMA como solução técnica para tratar de um dos principais desafios da PPS – reduzir o tempo de *setup*. Quanto a essa questão, não houve nenhum entrevistado que tenha discordado, pelo contrário, todos os entrevistados concordam que o SMA é uma contribuição para reduzir o tempo de *setup* na PPS.

Tabela 23 – Questionário aplicado aos desenvolvedores.

Questionário aplicado aos desenvolvedores do projeto COGMET/Bragecrim	
1	Você acredita que a utilização de um SMA para configuração e monitoramento da PPS pode diminuir efetivamente o tempo de *setup* da linha de produção?
2	Você acredita que a utilização de um SMA para o configuração e monitoramento da PPS pode auxiliar efetivamente os operadores nas tarefas diárias na empresa?
3	Na sua opinião, a nova proposta de abordagem MAS4SSP pode ser considerada uma alternativa para a implementação de sistema de configuração e monitoramento da PPS?
4	Apesar de ser um protótipo, você acha que a abordagem é adequada para ser utilizada na PPS (LabElectron)?
5	Caso você tivesse o poder de decisão em uma empresa que trabalha com PPS, você cogitaria usar o MAS4SSP?
6	Caso você tivesse o poder de decisão em uma empresa que trabalha com PPS, se o MAS4SSP não fosse uma proposta ABERTA, você cogitaria pagar pelas funcionalidades de um SMA para configuração e monitoramento da PPS?
7	Você acredita que a proposta apresentada, é flexível para adaptar o SMA para configuração e monitoramento da PPS às mudanças na linha de produção?
8	Na sua opinião, a proposta apresentada pode ser considerada inovadora?
9	Você acredita que esta nova abordagem multiagente possui vantagem de modelagem e implementação frente a abordagem baseada em agentes implementada no WZL/RWTH-Aachen?
10	Assinale a seguir o que você considera VANTAGEM desta abordagem quando comparado com a proposta alemã.
11	Assinale a seguir o que você considera DESVANTAGEM desta abordagem quando comparado com a proposta alemã.
12	Por favor, utilize o espaço a seguir para fazer comentários, sugestões, críticas, e eventualmente propostas de trabalhos futuros. Em especial para casos de respostas "discordo" às perguntas.

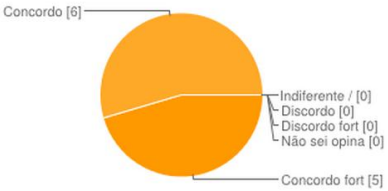
Gráfico 1 – Pergunta 1: O tempo de *setup* na PPS diminui?



Concordo fortemente	4	36%
Concordo	7	64%
Indiferente / não vejo correlação	0	0%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	0	0%

O Gráfico 2 mostra também que a maioria concorda quanto a contribuição do SMA no auxílio aos operadores nas tarefas diárias da linha de produção.

Gráfico 2 – Pergunta 2: O MAS4SSP auxiliará os operadores?

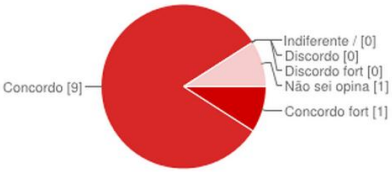


Concordo fortemente	5	45%
Concordo	6	55%
Indiferente / não vejo correlação	0	0%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	0	0%

As próximas perguntas do questionário eram específicas sobre a nova abordagem proposta. As perguntas foram classificadas em perguntas onde o entrevistado pode escolher apenas uma resposta, outras de múltipla escolha e no final uma pergunta aberta descritiva. A primeira pergunta questiona os entrevistados sobre se a proposta *MAS4SSP* pode ser considerada uma alternativa para implementar SMA para configurar e monitorar a PPS (Gráfico 3). O objetivo desta pergunta é respaldar o atendimento de um dos objetivos deste trabalho de pesquisa que é a verificação da viabilidade técnica da Arquitetura de Referência proposta. Dos entrevistados que participaram do desenvolvimento a maioria concorda que o *MAS4SSP* é uma solução técnica viável para configuração e monitoramento da PPS.

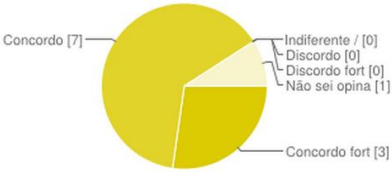
A pergunta 4 teve o objetivo de avaliar o experimento do LabElectron, se a solução proposta pode ser considerado uma solução técnica adequada. A maioria dos entrevistados concorda fortemente ou concorda com a afirmação (Gráfico 4).

Gráfico 3 – Pergunta 3: MAS4SSP é uma alternativa para a PPS?



Concordo fortemente	1	9%
Concordo	9	82%
Indiferente / não vejo correlação	0	0%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	1	9%

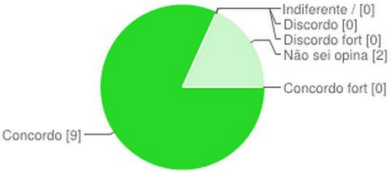
Gráfico 4 – Pergunta 4: MAS4SSP é uma solução para a PPS(LabElectron)?



Concordo fortemente	3	27%
Concordo	7	64%
Indiferente / não vejo correlação	0	0%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	1	9%

O Gráfico 5 tem o objetivo de avaliar o comprometimento dos desenvolvedores com a abordagem proposta. A pergunta se refere a possibilidade de utilizar a abordagem em uma aplicação no contexto da sua empresa atrelando a tomada de decisão a perdas e ganhos – ganho de promoção, perda do emprego. Aqui não havendo custos adicionais para a empresa além das horas da sua equipe de engenharia para customizar e implantar o *MAS4SSP*, ou pagamento de prestador de serviço externo. Esta questão também auxilia quanto ao interesse por uma solução aberta. As respostas apresentam bom comprometimento da equipe de desenvolvedores com a proposta.

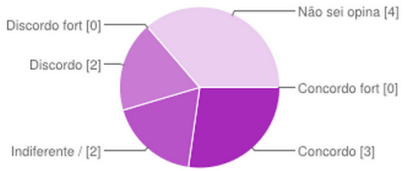
Gráfico 5 – Pergunta 5: Você usaria MAS4SSP na sua empresa?



Concordo fortemente	0	0%
Concordo	9	82%
Indiferente / não vejo correlação	0	0%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	2	18%

A pergunta 6 coloca um contraponto à questão 5, questionando sobre o interesse caso não se utilizassem soluções abertas na Arquitetura de Referência proposta. O comprometimento diminui, mas não inviabiliza (Gráfico 6).

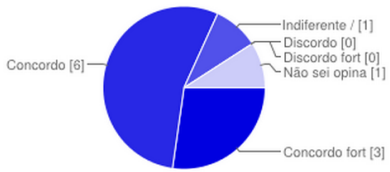
Gráfico 6 – Pergunta 6: Você usaria o MAS4SSP se fosse proprietário?



Concordo fortemente	0	0%
Concordo	3	27%
Indiferente / não vejo correlação	2	18%
Discordo	2	18%
Discordo fortemente	0	0%
Não sei opinar	4	36%

A pergunta 7 questiona sobre a flexibilidade da proposta, se alterações na linha da produção serão absorvidas pelo MAS4SSP. A maioria dos desenvolvedores aponta que a proposta é flexível o suficiente para atender as alterações constantes da linha de PPS (Gráfico 7).

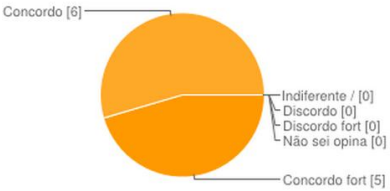
Gráfico 7 – Pergunta 7: MAS4SSP é flexível às mudanças da produção?



Concordo fortemente	3	27%
Concordo	6	55%
Indiferente / não vejo correlação	1	9%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	1	9%

O Gráfico 8 questiona os desenvolvedores sobre o ineditismo da proposta, aqui também há consenso que a Arquitetura de Referência proposta é considerada inovadora em alguma perspectiva – no contexto da PPS, no contexto do SMA ou ambos.

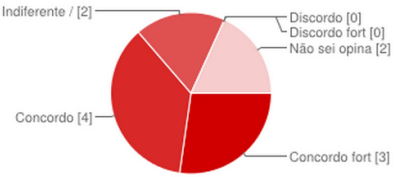
Gráfico 8 – Pergunta 8: A proposta é inovadora?



Concordo fortemente	5	45%
Concordo	6	55%
Indiferente / não vejo correlação	0	0%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	0	0%

A próxima pergunta faz uma comparação direta entre a abordagem proposta neste trabalho de pesquisa com a abordagem orientada a agentes implementada no parceiro da Alemanha. Questiona-se sobre qual é a melhor abordagem, a maioria dos desenvolvedores concorda com a proposta MAS4SSP (Gráfico 9).

Gráfico 9 – Pergunta 9: MAS4SSP é melhor que da Alemanha?



Concordo fortemente	3	27%
Concordo	4	36%
Indiferente / não vejo correlação	2	18%
Discordo	0	0%
Discordo fortemente	0	0%
Não sei opinar	2	18%

Na entrevista, duas perguntas apresentavam uma série de tópicos (Tabela 24) que foram avaliados durante o desenvolvimento do *MAS4SSP* e também quando comparado com o experimento implementado na Alemanha. Na primeira pergunta é pedido para o entrevistado selecionar quais são as vantagens do *MAS4SSP* quando comparado com a proposta seguindo o paradigma orientado a agentes implementada na Alemanha (Gráfico 10). A outra pergunta questiona sobre as desvantagens do MAS4SSP quando comparado com o experimento da Alemanha (Gráfico 11).

Tabela 24 – Tópicos avaliados pelos desenvolvedores.

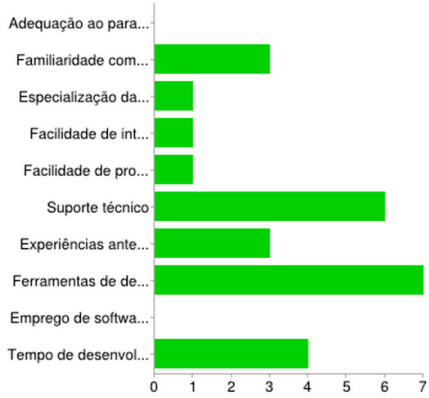
	Tópicos de comparação entre a abordagem do Brasil e da Alemanha
1	Adequação ao paradigma orientado a multiagente
2	Familiaridade com os paradigmas tradicionais de programação
3	Especialização das entidades (agente, organização, ambiente interno, ambiente externo)
4	Facilidade de integração dos frameworks, sistemas legados
5	Facilidade de programação
6	Suporte técnico
7	Experiências anteriores, grupos de desenvolvedores
8	Ferramentas de desenvolvimento maduras
9	Emprego de softwares livres
10	Tempo de desenvolvimento

A análise das respostas dos desenvolvedores a estas questões reforçou a constatação do trabalho de pesquisa que a Arquitetura de Referência baseada no Modelo de Referência JaCaMo integrado com o ScadaBR e *Web Service* tem como destaque (Gráfico 10): (i) a utilização ampla dos recursos do paradigma orientado a multiagente; (ii) a especialização do sistema complexo distribuído em dimensões adequadas as características das entidades participantes; (iii) e o emprego de tecnologias e padrões abertos que facilitam o uso e a integração com outros sistemas legados.

Gráfico 10 – Pergunta 10: Vantagens MAS4SSP frente à alemã

Quanto aos desafios (Gráfico 11) da Arquitetura de Referência MAS4SSP estão àqueles ligados normalmente à novidade: (i) dificuldade de adaptação ao novo paradigma orientado a multigante, em especial, a linguagem de programação orientada a agentes; (ii) e como ainda é um paradigma recente, as ferramentas de desenvolvimento ainda não estão completas e a comunidade de pesquisa e desenvolvimento ainda é inscipiente. Tanto estes desafios como os destaques foram percebidos durante a realização do trabalho de pesquisa pela maioria dos desenvolvedores.

Gráfico 11 – Pergunta 11: Desvantagens MAS4SSP frente à alemã



Para finalizar esta discussão, destacam-se dois comentários finais dos entrevistados que contribuem ainda mais para estas constatações:

> *A proposta MAS4SSP apresenta potencial frente as demais abordagens. O experimento do labelectron serviu para validar tecnicamente a proposta, contudo, novas implementações seriam necessárias para avaliar desempenho, eficiência e eficácia. A abordagem mostra ser um caminho promissor para o SMA na indústria!*
>
> Entrevistado #4

> *A utilização do paradigma de SMA ainda está suportada por ferramentas não-maduras e que possuem um baixo desempenho computacional. Acredito que para ficar mais explícita a vantagem do uso de SMA na indústria, seria necessária a presença de requisitos complexos e constantes mudanças no ambiente. Com isso, a implementação de um sistema baseado no paradigma de SMA poderia tornar menos oneroso o processo de implementação de um sistema flexível para automação/integração do chão de fábrica.*
>
> Entrevistado #5

6.1.2. Publicação de Artigos Científicos

A publicação de artigos para a comunidade científica é um fator de relevância para a evolução do trabalho em todo seu ciclo de definição, modelagem e projeto. Assim, procurou-se publicar regularmente em eventos relevantes nas temáticas do trabalho de pesquisa como uma forma de avaliação da proposta.

Foram publicados e apresentados 6 artigos em eventos internacionais sendo que 1 deles exclusivamente sobre o experimento no Labelectron. Foram também submetidos 2 artigos para períodicos internacionais, sendo 1 deles exclusivo sobre a Arquitetura de Referência proposta. As publicações são apresentadas no Tabela 25 e ordenados conforme data de publicação.

Esses artigos funcionaram também como uma avaliação preliminar dos resultados desenvolvidos ao longo trabalho de pesquisa e que se considerou como um tipo de aval da comunidade científica ao prosseguimento da pesquisa.

Tabela 25 – Lista de publicações em eventos científicos e revistas.

Título	Evento/Revista	Local	Ano	Tipo
Handling small series production inspection requirements through the use of cognitive and flexible metrology strategies	CIRP2010	Golfo de Nápoles, Itália	2010	Congresso Internacional
Cognitive Production Metrology: A new concept for flexibly attending the inspection requirements of small series production	MATADOR2010	Manchester, Inglaterra	2010	Congresso Internacional
Multiagent-based approach for the automation and quality assurance of the small series production	ETFA´2011	Tolouse, França	2011	Congresso Internacional
A Machine Vision System to Quality Assurance in a Small Series Production	ISMTII 2013	Aachen e Braunschweig, Alemanha	2013	Congresso Internacional
Artificial Intelligent Systems to Quality Assurance in Small Series Production	ISMTII 2013	Aachen e Braunschweig, Alemanha	2013	Congresso Internacional
Artificial Intelligent Systems to Quality Assurance in Small Series Production	Key Engineering Materials (Online)	Zurich-Durnten, Suiça	2014	Periódico Internacional
A Multi-Agent System for the Production Control of Printed Circuit Boards using JaCaMo and Prometheus AEOlus	INDIN´14	Porto Alegre, Brasil	2014	Congresso Internacional
MAS4SSP: a Multi-Agent Reference Architecture for the Small Series Production	IEEE/ASME – Transactions on Mechatronics	New Jersey, EUA	(2015)	Periódico Internacional

6.2. ATENDIMENTO AS EXPECTATIVAS DA PESQUISA

Esta seção traça um paralelo com a pergunta de pesquisa, objetivo geral e objetivos específicos apresentados no primeiro capítulo deste documento.

6.2.1. Pergunta de Pesquisa

A abordagem proposta é uma solução técnica para a configuração e o monitoramento da produção de pequenas séries?

Essa pergunta foi respondida com a implementação de um experimento baseado na Arquitetura de Referência proposta. A implementação seguiu um Modelo Genérico de Modelagem e Implementação que utilizou o experimento para a sua instanciação e customização ao cenário específico – PPS de PCI. Atendendo as necessidades do cenário do experimento, o MAS4SSP do Labelectron se adequou de forma efetiva e efetuando as tarefas definidas para o experimento.

6.2.2. Objetivo Geral

> O objetivo geral desta tese é propor uma nova abordagem para a implementação de sistema multiagente para a configuração e monitoramento da produção de pequenas séries.

No Capítulo 4 foi apresentada a Arquitetura de Referência para o SMA para configuração e monitoramento da PPS. Essa Arquitetura de Referência foi baseada em um Modelo de Referência – JaCaMo – criado com base paradigma VOGAL. Com a Arquitetura de Referência construída foi possível criar um Modelo Genérico de Modelagem e Implementação onde o usuário pode customizar o SMA para configurar e monitorar a PPS no cenário da sua aplicação, seja, placas de circuito impresso, faróis automotivos, cadeiras de dentista, refrigerados customizados, etc.

O SMA para configuração e monitoramento da PPS de PCI implementado fornece uma ferramenta eficiente para diminuir o tempo de *setup* da linha e também fornece auxílio aos operadores, assumindo certas atividades da produção, como o monitoramento do estado dos equipamentos.

6.2.3. Objetivos Específicos

> Definir o Modelo de Referência para o sistema multiagente para a configuração e monitoramento da produção de pequenas séries.

O Modelo de Referência para o SMA foi apresentado no capítulo 3. Este foi baseado na pesquisa de referências do estado da arte sobre plataformas multiagente que seguem o paradigma VOGAL e também

sobre os SMA aplicados na indústria. O paradigma VOGAL, o framework JaCaMo, as abordagens dos projetos baseados em agentes na indústria serviram como base para a criação da Arquitetura de Referência.

> Conceber uma Arquitetura de Referência para a configuração e o monitoramento da produção de pequenas séries.

Ainda no Capítulo 4, foi apresentada a Arquitetura de Referência do SMA para configuração e monitoramento da PPS, que utilizou o Modelo de Referência JaCaMo e os Requisitos Desejáveis – SMA como estilo arquitetural para propor a arquitetura MAS4SSP.

> Apresentar um modelo de implementação genérico baseado no Modelo de Referência e na Arquitetura de Referência.

No Capítulo 4 também foi apresentada a modelagem e a implementação do Modelo Genérico de Modelagem e Implementação baseado na Arquitetura de Referência.

> Experimentar a proposta em um protótipo computacional associado à Arquitetura de Referência concebida, para servir de instrumento de avaliação da proposta.

O Modelo Genérico de Modelagem e Implementação serviu de base para a implementação de um protótipo – experimento do Labelectron que foi apresentado e discutido do ponto de vista técnico no Capítulo 5. O Capítulo 6 no seu início apresenta uma análise geral da proposta e do experimento baseada em entrevista com os desenvolvedores e nas publicações científicas produzidas durante o trabalho de pesquisa.

Por fim, analisando a Arquitetura de Referência e o experimento implementado com base nesta arquitetura seguindo um Modelo Genérico de Modelagem e Implementação se conclui que a arquitetura implementa o paradigma orientado a multiagente de forma integrada e sinérgica; a modelagem possui um método que permite o projeto do início ao fim seguindo o paradigma orientado a multiagente – ainda necessita uma ferramenta start-to-end que implementa o método e realiza a integração automatizada com o método UML; a inserção da

camada SCADA proporciona a integração com os mais variados sistemas de chão-de-fábrica – integra o ScadaBR com o SMA via Web Service e usa drivers do SCADA para comunicação com os equipamentos; após o trabalho de pesquisa realizado se constata que aparentemente esta arquitetura poderia ser utilizada em outros sistemas de produção além da PPS – carece de experimentos; o procedimento de seleção de um Modelo de Referência (JaCaMo) que contribui para a criação de uma Arquitetura de Referência (MAS4SSP) que gera um modelo de implementação genérico se mostra válido após a implementação do experimento do Labelectron; e finalmente, esta nova abordagem proposta se mostra uma alternativa válida para a implementação de SMA na indústria.

6.3. CONCLUSÕES

Em um cenário de sistemas multiagente para a configuração e o monitoramento PPS, este trabalho de pesquisa apresentou uma nova abordagem para a implementação de tais SMA na configuração e monitoramento da PPS baseado em uma Arquitetura de Referência que foi proposta a partir de um Modelo de Referência e Requisitos Desejáveis.

Para a proposição da Arquitetura de Referência foram efetuadas pesquisas do estado da arte na áreas de SMA e PPS. Essas pesquisas forneceram bases suficientes para a formulação de características necessárias para se trabalhar na pesquisa. A pesquisa do estado da arte foi motivadora para a criação de uma nova abordagem para a implementação de SMA para configurar e monitorar a PPS.

Constata-se que a Arquitetura de Referência resolve, ou minimiza, uma aspectos relacionados a implementação de um sistema baseado em agentes em um ambiente de manufatura inteligente, como é o caso da PPS. Esta nova abordagem minimiza aspectos como: (i) a complexidade da programação do SMA baseado em agentes devido a quantidade de agentes gerados pela agentificação do cenário de estudo – são cenário diferentes mas a título de avaliação superficial, no experimento alemão, baseado na agentificação, foram implementados 25 agentes ao passo que no experimento brasileiro foram implementados 6 agentes interagindo com recursos e artefatos do ambiente e com metas organizacionais; (ii) apresenta um processo sinérgico e unificado de modelagem e implementação do SMA; (iii) especializa o SMA em dimensões com funções bem definidas; aumenta a flexibilidade da implementação do SMA para configuração e monitoramento da PPS

apresentando um ambiente genérico que pode ser especializado para o cenário de aplicação; (iv) integra o sistema SCADA como ferramenta para transformar o ambiente heterogêneo da linha de produção em um ambiente homogêneo do ponto de vista do SMA; (v) aplica a tecnologia de *Web Service* como ferramenta de comunicação como sistemas legados.

Entretanto, constata-se que a área de SMA para da PPS ainda possui desafios importantes a serem enfrentados, como o amadurecimento das tecnologias (como o *framework* JaCaMo), a padronização de fato da tecnologia *Web Service*, a finalização do método Prometheus AEOlus – com uma ferramenta de modelagem e geração de código completa – consolidando-o como uma ferramenta *start-to-end*, a promoção do emprego do ScadaBR como mecanismo de interação entre a linha de produção e o SMA e a disseminação do paradigma de programação orientado a multiagente tanto na comunidade científica como empresarial.

Quanto ao uso do SMA na manufatura, em especial na PPS, esta tecnologia ainda é considerada de fronteira no ambiente empresarial. Durante o trabalho de pesquisa, várias iniciativas foram estudadas e com raras exceções, os trabalhos se limitam ao ambiente de laboratório ou protótipos não integrados com a produção. Este cenário aparente de não interesse pela comunidade empresarial ocorre da mesma forma que ocorreu com outros paradigmas que demoraram décadas até servirem de fato, ao ambiente empresarial. O que fica claro com este trabalho e pesquisa e com os trabalhos estudados é que o SMA é uma alternativa técnica viável para a configuração e o monitoramento da PPS.

Definir o SMA como a solução para todos os problemas relacionados com os vários aspectos de um ambiente de manufatura adaptativo, flexível, interoperável, intercambiável e integrando os ambientes empresariais é uma proposta das mais complexas e, sob certas perspectivas, ainda não plenamente solucionável com o estado das tecnologias atuais (como o caso da dimensão de interação em um ambiente hirárquico e heterogêneo) e abordagens conceituais. Contudo, esta é uma excelente perspectiva de pesquisas futuras.

O trabalho de pesquisa aqui apresentado trouxe uma contribuição conceitual e de modelo de implementação de SMA para PPS que vai na direção dos Requisitos Desejáveis para um SMA para PPS. A premissa básica desta pesquisa é de que as empresas necessitam cada vez mais de sistemas de manufatura inteligentes que contribuam para viabilizar técnica e economicamente uma produção diversificada e com número de unidades tendendo a unidade – modelo *first time right on time*.

Este trabalho deve ser visto como uma contribuição para a área de SMA na configuração e monitoramento da PPS, buscando atender certos Requisitos Desejáveis identificados na literatura. A primeira contribuição a ser citada, é que não existe uma Arquitetura de Referência para a criação de SMA para configurar e monitorar a PPS. Existem implementações prontas ou direcionadas aos seus próprios requisitos de implantação. Não existe a flexibilidade de inserção de novas funcionalidades para esses SMA, não são baseados uma Arquitetura de Referência baseada no paradigma orientado a multiagente e não possuem um modelo de implementação genérico para esta Arquitetura de Referência. Assim, esse trabalho apresenta uma proposta de uma Arquitetura de Referência que possa servir de base para a criação de soluções que possam ser integradas e adaptadas aos diversos tipos de linhas de PPS. Arquitetura de Referência é flexível o suficiente para que ela possa atender as mudanças na linha de produção durante o tempo de operação e ainda ser interoperável com os ambientes e sistemas empresariais. Uma outra contribuição – da perspectiva de mercado – é que esta abordagem fornece um modelo de negócios a ser explorado: a criação de SMA para configurar e monitorar a PPS baseado em uma Arquitetura de Referência. Cada empresa que trabalha com PPS pode possuir sua própria implementação do Modelo Genérico de Modelagem e Implementação baseado na Arquitetura de Referência. Cada empresa pode integrar o MAS4SSP aos outros sistemas legados.

Quanto a avaliação dos resultados, três procedimentos metodológicos foram executados. O primeiro (Capítulo 5) foi a implementação de uma instância baseada na Arquitetura de Referência em cima de um experimento específico possuindo como referência uma linha de produção de placas de circuito impresso. O segundo procedimento (Capítulo 6) foi a apresentação da proposta e a aplicação de questionário para o grupo de desenvolvedores que participaram do desenvolvimento da abordagem. O terceiro procedimento (Capítulo 6) foi a publicação de artigos científicos para avaliação da proposta da abordagem durante o seu ciclo de vida, de modo a verificá-la frente a comunidade científica.

Em relação ao desenvolvimento de uma instância implementada baseada na Arquitetura de Referência proposta, foi gerado um protótipo de SMA para configuração e monitoramento da PPS que envolve todo o conjunto de tecnologias propostas. Mesmo sendo um protótipo, toda a complexidade de um ambiente de produção baseado em agentes está compreendida no experimento do LabElectron. Conclui-se que com o experimento foram atendidas as pecularidades de outros cenários de

PPS. Constata-se que para cada experimento serão necessárias certas customizações do Modelo Genérico de Modelagem e Implementação para o cenário onde será aplicado. Contudo, estas customizações serão do ponto de vista de algoritmo e não serão alterados modelos.

Com base nos resultados do teste de implementação da Arquitetura de Referência, verificou-se que esta instância se comportou conforme o que foi proposto e cumpriu corretamente os objetivo estabelecido – ser uma solução técnica para configuração e monitoramento da PPS. Embora esse não tenha sido o principal foco do trabalho, o sistema apresentou boa estabilidade quanto ao fato de ser uma arquitetura distribuída.

Outro ponto que merece destaque é que a instância implementada derivada do modelo se mostrou coerente com o Modelo de Referência selecionado e a Arquitetura de Referência concebida. Com a implementação realizada, mostrou-se que é possível realizar a derivação da linha de produção a partir de um modelo geral e genérico de recursos não autônomos (artefatos) e assim implementar um SMA orientado pelo paradigma VOGAL à linha de PPS, interoperando com subsistemas via SCADA, que trabalham dentro de uma arquitetura aberta e de intenso uso de padrões como descrito nesse trabalho.

Com base nas respostas dos questionários, baseados na forma de pesquisa qualitativa desse trabalho, pode-se observar uma boa aceitação da abordagem proposta.

Esta nova abordagem de implementação de SMA para configurar e monitorar a PPS aqui apresentada, além de apresentar uma abordagem inovadora baseada em uma Arquitetura de Referência, fornece uma base para o emprego da SMA em outros ramos da manufatura.

6.3.1. Limitações

Baseado em uma análise conjunta dos resultados tanto da parte do experimento como da etapa de avaliação da abordagem – entrevistas e artigos, algumas limitações foram identificadas. As limitações não são em relação à Arquitetura de Referência especificamente, mas a nível de modelagem e implementação do SMA para configuração e monitoramento da PPS. Estas limitações são discutidas a seguir:

- Modelagem: embora atualmente exista um método de modelagem proposto para o *framework* JaCaMo – Prometheus AEOlus – este método possui duas limitações: a primeira quanto a ausência de uma ferramenta de modelagem *start-to-end.* Atualmente o método utiliza o plugin PDT do

método original e os diagramas novos não são suportados. Além disso, não existe uma ferramenta de geração de código *user friendly*; a segunda se refere ao desacoplamento do diagrama de Visão Geral do Ambiente, do descritor de Artefatos do diagrama de Classes da UML. Sugeriu-se aos desenvolvedores adicionar a funcionalidade de refinar os artefatos e recursos do ambiente dentro de um diagrama de Classes UML. Com isso se cobriria quase que completamente o processo de modelagem, restando somente a definição das *tags* de comunicação do artefato *Web Service* do ambiente com a aplicação SCADA – ScadaBR – associada à linha de produção.

- Suporte: como o Modelo de Referência ainda está em desenvolvimento, assim como o sistema ScadaBR, a comunidade de desenvolvimento e de usuários ainda são inscipientes. O suporte às dúvidas é realizado normalmente pelos próprios criadores dos frameworks o que não deixa de ser interessante mas limita a disseminação do conhecimento concentrando-o em poucas pessoas.
- Ferramentas novas: como todos os modelos, *frameworks*, ferramentas e métodos apresentados neste trabalho de pesquisa tem sua origem na academia há resistência da comunidade empresarial em adotar sistemas que não possuam uma empresa responsável. Ainda mais o fato dessas ferramentas estarem em desenvolvimento, o ambiente industrial as considera imaturas para aplicações industriais.
- Software livre: atrelado os estigma da solução acadêmica está o estigma dos projetos de código aberto (software livre) que possuem resistência na indústria pelo mesmo fato da inexistência de uma entidade privada responsável. Além de pensamentos como: "se não custa nada é porque não funciona!".

6.3.2. Trabalhos Futuros

Tendo como base as limitações e certos pontos apresentados durante toda a extensão deste trabalho de pesquisa, alguns trabalhos futuros podem ser sugeridos como forma de gerar novas pesquisas subsequentes ou derivados a este trabalho em questão.

A proposta da tese se manteve no contexto da PPS, contudo, após validar tecnicamente a Arquitetura de Referência se levantou a hipótese

que esta nova proposta de abordagem de SMA poderia ser aplicada em outros sistemas de produção. Existem inúmeras iniciativas – algumas apresentadas no Capítulo 3 – que poderiam servir de cenário para a implementação de um novo experimento seguindo a Arquitetura de Referência e o modelo de implementação genérico.

Um outro trabalho que pode ser feito é quanto a inserção da dimensão Interação da Arquitetura de Referência no modelo de implementação. Um modelo conceitual de integração da Interação no framework JaCaMo foi proposto por Zatelli (2013), ainda necessita de amadurecimento para criar um mecanismo de alto nível de abstração assim como as demais dimensões. A dimensão de Interação também necessitaria ser integrada no método Prometheus AEOlus, desta forma, seria necessária também uma revisão do método. Com isto seria um interessante trabalho de pesquisa expandir a arquitetura MAS4SSP para os níveis hierárquicos superiores da empresa, chegando até mesmo a toda cadeia de fornecedores e clientes. A expansão do MAS4SSP para os níveis hierárquicos superiores aparentemente seria um desafio quando a dimensão de Interação – devido a existência de ínumeros e diversos protocolos de comunicação que exigiram um sistema ontológico de alto nível de abstração.

Outro trabalho de pesquisa está relacionado com a implementação de experimentos que considerem duas abordagens. Uma abordagem seguindo o paradigma orientado e agentes e outra abordagem seguindo a Arquitetura de Referência (MAS4SSP) baseada no paradigma orientado a multiagente. Com duas abordagens sobre um mesmo cenário de aplicação é possível realizar estudos mais detalhados sobre ganhos e perdas em cada uma. Além disso, é possível levantar indicadores com maior rigor científico para que além da viabilidade técnica se avaliem questões de desempenho de cada uma das abordagens.

Um problema constatado e já descrito é quanto o emprego do paradigma orientado a multiagente, um paradigma de ruptura, uma nova forma de pensar que para os desenvolvedores atuais acarreta um esforço grande – segundo o entrevistados – para alterar a forma de pensar o desenvolvimento de um sistema. Uma forma de acabar com este problema é a capacitação dos novos desenvolvedores neste novo paradigma e para os desenvolvedores atuais um ambiente de desenvolvimento que abstrai o paradigma seria uma alternativa. Sucintamente um ambiente onde o projetista desenha o que deseja e a ferramenta automatiza a modelagem e codificação do sistema.

REFERÊNCIAS

ALBUQUERQUE, P. U. B. de; ALEXANDRIA, A. R. de; REDES INDUSTRIAIS: **Aplicações em Sistemas Digitais de Controle Distribuído.** Fortaleza: Edições Livro Técnico, 2007.

AL-SALTI, M.; STATHAM A. **The Application of Group Technology Concept for Implementing SPC in Small Batch Manufacture**. International Journal of Quality & Reliability Management. V.11, Issue 4, p. 64-76. 1994

ARENAS L**. Cartas de control de corrida corta: estudio introductorio**. Neuquén, Noviembre 2005. Trabajo final de control estadístico de procesos. Facultad de Economía y Administración – U. N. del Comahue.

BABAOGLU, O.; et al. **Anthill: a framework for the development of agent-based peer-to-peer systems**. In Proceedings of the 22nd International Conference on Distributed Computing Systems, Vienna, Austria. Los Alamitos, CA: IEEE Computer Society Press, pp. 15–22, 2002.

BASS, L.; CLEMENTS, P. and KAZMAN, R. **Software architecture in practice.** 2nd ed. Addison-Wesley, 2003.

BEHRENS, T. M.; HINDRICKS, K. V.; DIX, J. **Towards an environment interface standard for agent platforms.** Annals of Mathematics and Artificial Intelligence, 61(4):261–295, 2011.

BELLIFEMINE, F.; POGGI, A.; RIMASSA, G. **Jade, A FIPA-compliant Agent Framework.** 4th International Conference on Practical Application of Intelligent Agents and Multi-Agent Technology, 2001.

BENSLIMANE, D.; DUSTDAR, S.; SHETCH, A. (2008). **Services Mashups: The New Generation of Web Applications.** IEEE Internet Computing 10 (5): 13–15. doi:10.1109/MIC.2008.110

BERGENTI, F.; GLEIZES, M-P.; ZAMBONELLI, F. (Org.) **Methodologies and Software Engineering for Agent Systems,** Kluwer Academic Publishers, 2004.

BERNON, Carole; COSSENTINO, Massimo; PAVON, Juan. **An Overview of Current Trends in European AOSE Research**. Informatica - An International Journal of Computing and Informatics, 29:379-390, 2005.

BLACK, J.T. **O Projeto da fábrica com futuro.** Porto Alegre, Editora Bookman, 2001, 627 p.

BLAZEWICZ, J., DOMSCHKE, W., PESCH, E. **The job shop scheduling problem: Conventional and new solutions techniques**. European Journal of Operational Research, v.93, p. 1-33, 1996.

BONABEAU, E.; et al. **Routing in telecommunications networks with ant-like agents**. In Proceedings of the 2nd International Workshop on Intelligent Agents for Telecommunication a Applications, Paris, France. London: Springer, pp. 60–71, 1998.

BOISSIER, Olivier; HÜBNER, Jomi F.; SICHMAN, J. S. **Organization oriented programming: From closed to open organizations**, in: O´HARE, G. M. P.; RICCI, A.; O´GRADY, M. J.; DIKENELLI, O. (Eds.), ESAW, Vol. 4457 of LNCS, Springer, 2006, pp. 86-105.

BOISSIER, Olivier; BORDINI, Rafael H.; HÜBNER, Jomi F.; RICCI, Alessandro; SANTI, Andrea. **Multi-agent oriented programming with JaCaMo**. Science of Computer Programming (2011), doi:10.1016/j.scico.2011.10.004, accepted 6 October 2011.

BORDINI, R. H.; DASTANI, M.; DIX, J.; FALLAH-SEGHROUCHNI, A. E. (Eds.). **Multi-Agent Programming: Languages, Platforms and Applications**, Springer, 2005.

BORDINI, R. H.; HÜBNER, J. F.; WOOLDRIDGE, M. **Programming Multi-Agent Systems in AgentSpeak Using Jason**. Wiley Series in Agent Technology. John Wiley & Sons, 2007.

BRATMAN, Michael E.; ISRAEL, David J.; POLLACK, Martha E.. **Plans and Resource-Bounded Practical Reasoning.** Appers in Computational Intelligence, 4(4):349-355, 1988.

BRAUBACH, L.; POKAHR, E.; LAMERSDORF, W. Jadex: **A BDI agent system combining middleware and reasoning.** In Ch. of Software Agent-Based Applications, Platforms and Development Kits, pages 143–168. Birkhaeuser, 2005.

BRECHER, Christian**. Integrative Production Technology for High-Wage Countries.** Springer. 1096 p. 2012. ISBN 978-3-642-21066-2

BRESCIANI, P, et al. **Tropos: An Agent-Oriented Software Development Methodology.** In: Journal of Autonomous Agents and Multi-Agent Systems, v. 8, n. 3, p. 203-236, 2004.

BROOKS, R. A. **A robust layered control system for a mobile robot.** In: IEEE Journal of Robotics and Automation, v. 2, n. 1, p.14-23, 1986.

BRUSSEL, H. V.; WYNS, J.; VALCKENAERS, P.; BONGAERTS, L. **Architecture for holonic manufacturing systems: PROSA**. Comput. Ind., vol. 37, no. 3, pp. 255–274, 1998.

BURGESS, T. **Quality management for manufacturers of short run semicustomized products**. The TQM Magazine, Volume 11, Number 4, 1999 , p. 24-25

CAIRE, G. et al.. **Agent-oriented analysis using Message/UML.** In: Proceedings of the 2nd International Workshop on Agent-Oriented Software Engineering (AOSE 2001), p. 101-107, 2001.

CASTELFRANCHI, C. **Commitments: From individual intentions to groups and organizations.** In: ISHIDA, T. (Ed.), Proceedings of the First International Conference on Multi-Agent Systems (ICMAS 95), p. 41–48, 1995.

CERVO, A. L. e BERVIAN, P. A. **Metodologia Científica**. São Paulo: Prentice Hall., 5a. Edição. 2002.

CHENG, C.S. **Group technology and expert systems concepts applied to statistical process control in small-batch manufacturing.** Arizona, 1989. 382p. PhD thesis, Arizona State University.

CHIRN, J.-L.; McFARLANE, D. **A holonic component-based approach to reconfigurable manufacturing control architecture**. in Proc. Int. Workshop Ind. Appl. Holonic Multiagent Syst., 2000, pp. 219–223.

CIANCARINI, Paolo; WOOLDRIDGE, Michael. **Agent-Oriented Software Engineering - The State of the Art,** Band 1957 der Reihe Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

COBURN, J. M. **JACK Intelligent Agents User Guide**, AOS Technical Report. 2001. Available on: <http://www.agent-software.com>. Accessed: 05/2012

COSSENTINO, M. **For Requirements to Code with the PASSI Methodology**. In: HENDERSON-SELLERS, B., GIORGINI, P. (Ed.), Agent-Oriented Methodologies, Idea Group Publishing, p. 79-106, 2005.

COULOURIS, G.F.; DOLLIMORE, J.; KINDBERG, T. **Distributed systems: concepts and design.** Addison Wesley Longman, 2005.

DASTANI, M.; GROSSI, D.; MEYER, J.-J.; TINNEMEIER, N. **Normative Multi-Agent Programs and Their Logics**, in: KRAMAS-08, Proceedings, 2008.

DELOACH, S. A.; GARCÍA-OJEDA, J. C. O-MaSE: a customisable approach to designing and building complex, adaptive multi-agent systems. In: International Journal Agent-Oriented Software Engineering, v. 4, n. 3, p. 244-280, 2010.

DELOACH, S. A.; VALENZUELA, J. L. **An agent-environment interaction model.** In Proc. of AOSE, pages 1–18, Berlin, Heidelberg, 2006. Springer.

DEMAZEAU, Y. **From interactions to collective behaviour in agent-based systems**, in: Proceedings of the 1st. European Conference on Cognitive Science. Saint-Malo, pp. 117-132, 1995.

DIGNUM, V. **A model for organizational interaction: based on agents, founded in logic.** Dissertation Series n..2004-1, 2004, 270 p. Thesis (PhD) – University of Utrecht, Utrecht, 2004.

DORNELAS, J. C. A. **Empreendedorismo – Transformando Idéias em Negócios**. 4a. Edição Revisada e Ampliada. Editora Elsevier, 2011.

DORO, Marcos Marinovic. **Solução integrada para auxiliar na garantia da qualidade na produção em pequenos lotes**. Tese de Doutorado. Florianópolis: Programa de Pós-Graduação em Engenharia Mecânica, Universidade Federal de Santa Catarina, 2009.

ELAM, M. E. **Control Charts for Short Production Runs**. Encyclopedia of Statistics in Quality and Reliability, March 2008.

ESTEVA, M., PADGET, J., SIERRA, C. **Formalizing a language for institutions and norms.** In: MEYER, J.-J. C.; TAMBE, M. (Ed.), Intelligent Agents VIII: 8th International Workshop, ATAL 2001, Seattle, WA, USA, August 1-3. Revised Papers. Berlin Heidelberg: Springer, (LNAI, v. 2333), p. 348–366 2002.

EVANS, R.; KEARNEY, P.; CAIRE, G.; GARIJO, F.; GOMEZ SANZ, J.; PAVON, J.; LEAL, F.; CHAINHO, P.; MASSONET, P. **MESSAGE: Methodology for Engineering Systems of Software Agents.** EURESCOM, EDIN 0223-0907 (2001)

FERBER, Jacques**. Multi-Agent Systems: An Introduction to Artificial Intelligence.** Addison-Wesley. ISBN 0-201-36048-9. 1999.

FERBER, J.; MICHEL, F.;BAEZ, J. AGRE: **Integrating environments with organizations.** In: WEYNS, D., PARUNAK, H.; MICHEL, F. (Ed.), Environments for Multi-Agent Systems, (E4MAS 2004), Lecture Notes in Computer Science (LNCS), Springer- Verlag, Berlin, v.3374, p. 48–56, 2005.

FERGUSON, I. A. Integrated control and coordinated behavior: a case for agent models. Intelligent Agents. In Lecture Notes in Computer Science Volume 890, pp 203-218, 1995.

GARCIA-OJEDA, J. C. et al**. O-Mase: A Customizable Approach to Developing Multiagent Development Process.** In: Agent-Oriented Software Engineering VIII: 8th International Workshop on Agent-oriented Software Engineering at AAMAS 2007, Lecture Notes in Computer Science (LNCS), Springer-Verlag, Berlin Heidelberg, v. 4951, p.1-15, 2008.

GESSER, Carlos E. **Uma abordagem para a integração dinâmica de serviços web em portais.** Eng. Elétrica - UFSC. Dissertação. 2006.

GIL, A.C. **Como elaborar projetos de pesquisa.** 5ª edição. São Paulo: Atlas. 2010.

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

GÖHNER, Peter; WAGNER, Paulo; WAGNER, Thomas. **Softwareagenten - Einführung und überblick Über eine alternative Art der Softwareentwicklung.** Teil III: Agentensysteme in der Automatisierungstechnik: Aufbau, Struktur und Implementierung an einem Anwendungsbeispiel. atp - Automatisierungstechnische Praxis, 46(2):42-51, 2004.

GOMEZ-SANZ, J.; GERVAIS, M.P. AND WEISS, G. **A survey on Agent-Oriented Oriented Software Engineering Research.** In: BERGENTI,V; GLEIZES, M. P.; ZAMBONELLI, F.(Ed.), Methodologies and software engineering for agent systems, Kluwer Academic Publishers, p. 33-62, 2004.

GRAND, Stephen; CLIFF, Dave; MALHOTRA, Anil. **Creatures: artificial life autonomous software agents for home entertainment.** In: AGENTS '97, Proceedings of the first international conference on Autonomous agents, Pgs. 22-29, 1997.

HINDRIKS, K. V**. Programming rational agents in GOAL.** Multi-Agent Programming: Languages and Tools and Applications, pages 119–157, 2009.

HITOMI, K. **Manufacturing Systems Engineering: A Unified Approach to Manufacturing Technology**. Production Management and Industrial Economics. 2nd edition, CRC Press, 1996. 536 p.

HOFMEISTER, C.; NORD, R.; SONI, D. **Applied Software Architecture.** Addison Wesley, 2000.

HOWDEN, N.; RONNQUIST, R.; HODGSON, A.; LUCAS, A. **JACK intelligent agents—Summary of an agent infrastructure.** In Proc. 5th ACM Int. Conf. Auton. Agents, 2001, pp. 251–257.

HOWELL, J.; YOUNG, R.C. **Reasoning about correlation-based inspection of small-batch-manufactured objects.** Robotics and Computer-Integrated Manufacturing, Volume 12, Number 3, September 1996 , pp. 243-249.

HÜBNER, J. F.; BOISSIER, O.; KITIO, Rosine; RICCI, Alessandro**. Instrumenting multi-agent organisations with organisational artifacts and agents: "Giving the organisational power back to the agents"**. Auton. Agent Multi-Agent Syst. (2010) 20:369–400. DOI 10.1007/s10458-009-9084-y

HÜBNER, J. F.; BOISSIER, O.; BORDINI, R. H. **From organisation specication to normative programming in multi-agent organisations**, in: CLIMA XI, pp. 117-134, 2010b.

HÜBNER, J. F.; SICHMAN, J.S.; BOISSIER, O**. MOISE+ Tutorial.** 2008. Avalable on <http://moise.sourceforge.net/doc/tutorial.pdf>.

HÜBNER, J. F.; SICHMAN, J.S.; BOISSIER, O**. Developing organised multi-agent systems using the MOISE+ model: Programming issues at the system and agent levels.** International Journal of Agent-Oriented Software Engineering, 1(3/4), 370–395. 2007.

HÜBNER, J. F. **Um Modelo de Reorganização de Sistemas Multiagentes.** Tese de Doutorado da Escola Politécnica da Universidade de São Paulo, 2003.

HUHNS, M. N. **Interaction-oriented programming,** in: First international workshop, AOSE 2000 on Agent-oriented software

engineering, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001, pp. 29-44.

HUHNS, M. N.; STEPHENS, L. M. **Multi-Agent Systems and Societies of Agents.** G. Weiss (ed.), Multi-agent Systems, ISBN 0-262-23203-0, MIT press (1999)

JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. **The Unified Software Development Process,** Addison-Wesley, 1999. 463 p.

JADHAV, S. A. **Setup Approval and Self Starting Schemes for Short Production Runs Pune**, 2005, 103 p. MEng thesis, University of Pune.

JENNINGS, N. **On agent-based software engineering.** Artificial Intelligence, 117(2):277-296, March, 2000.

JURAN, J. M.; GODFREY, A. B.; HOOGSTOEL, R. E.; SCHILLING, E. G. **Juran's Quality Handbook.** 5rd Edition, USA: McGraw-Hill, Inc, 1998. 1730 p.

LEITÃO, P.; COLOMBO, A.W.; RESTIVO, F. ADACOR: **A collaborative production automation and control architecture.** IEEE Intell. Syst., vol. 20, no. 1, pp. 58–66, Jan./Feb. 2005.

LEITÃO, P. **Agent-based Distributed Manufacturing Control: A State-of-the-art Survey.** Engineering Applications of Artificial Intelligence 22(7), 979–991, 2009.

LEITÃO, P.; VRBA, P. **Recent Developments and Future Trends of Industrial Agents.** In: MARIK, V.; VRBA, P. & LEITÃO, P. (Eds.): HoloMAS 2011, LNAI 6867, pp. 15–28, 2011.

LEITÃO, P. & RODRIGUES, N. **Multi-Agent System for On-demand Production Integrating Production and Quality Control.** In: MARIK, V.; VRBA, P. & LEITÃO, P. (Eds.): HoloMAS 2011, LNAI 6867, pp. 84–93, 2011.

LEITÃO, P.; BARBOSA, J.; VRBA, P.; SKOBELEV, P.; TSAREV, A.; KAZANSKAIA, D. **Multi-agent System Approach for the Strategic Planning in Ramp-up Production of Small Lots.** 2013 IEEE

International Conference on Systems, Man, and Cybernetics. DOI 10.1109/SMC.2013.807.

LEITÃO, P.; MARÍK, V.; VRBA, P. **Past, Present, and Future of Industrial Agent Applications.** IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 9, NO. 4, NOVEMBER, 2013.

LEMAITRE, C.; EXCELENTE, C. B. Multi-**agent organization approach.** In: The Second Iberoamerican Workshop on Distributed AI and MAS, Toledo, Spain, 1998.

LESPERANCE, Y. et al. **Foundations of a logical approach to agent programming.** In: WOOLDRIDGE, M.; MULLER, J. P.; TAMBE, M. (Ed.), Intelligent Agents II, Springer-Verlag: Berlin, Germany, LNAI v. 1037, p. 331-346. 1996.

LI, G.; WANG, H. **Research on Quality Control Technology of Capacity Flexibility Oriented to Multi-Specification & Small Batch**. in: Wireless Communications, Networking and Mobile Computing. WiCOM 08. 4th International Conference on. 12-14 Oct. 2008 p. 1-5.

LIAN, Z.; COLOSIMO, B. M. CASTILLO DEL, E. **Setup adjustment of multiple lots using a Sequential Monte Carlo method**. Technometrics, 2006, Vol. 48, Number 3, pp. 373-385.

LIN, S.; LAI, Y.; CHANG, S. **Short-run statistical process control: Multicriteria part family formation.** Quality and reliability engineering international, 1997, Vol. 13, No. 1, p 9-24.

LUCERO A. G. R. **Um método de otimização para a programação da manufatura em pequenos lotes**. Florianópolis, 2001, 113 f. Dissertação (Mestrado em Engenharia Mecânica), Universidade Federal de Santa Catarina.

MAES, Pattie. **Artificial life meets entertainment: lifelike autonomous agents**. Communications of the ACM, 38(11), 1995.

MAGEDANZ, T.; ROTHERMEL, K.; KRAUSE, S. **Intelligent agents: an emerging technology for next generation telecommunications?** In: Proceedings of IEEE INFOCOM '96. Conference on Computer Communications, Seiten 464{472, San Francisco, 1996. IEEE

Comput. Soc. Press.

MAHONEY, R. M. **High-Mix Low-Volume Manufacturing**. New Jersey: Pertince Hall, 1997. 222 p.

MARIK, César A.; MÖNCH, Lars; LEITÃO, Paulo; VRBA, Pavel; KAZANSKAIA, Daria; CHEPEGIN, Vadim; LIU, Liwei; MEHANDJIEV, Nikolay. Conceptual Architecture Based on Intelligent Services for Manufacturing Support Systems. 2013 IEEE International Conference on Systems, Man, and Cybernetics. DOI 10.1109/SMC.2013.808

MARIK, V.; LAZANSKY, J. **Industrial Applications of Agents Technologies.** Control Engineering Practice 15, 1364–1380, 2007.

MATURANA, F.; SHEN, W.; NORRIE, D. H. **MetaMorph: An adaptive agent-based architecture for intelligent manufacturing**. Int. J. Prod. Res., vol. 37, no. 10, pp. 2159–2174, 1999.

MILLER, T.; McBURNEY P. **Using constraints and process algebra for specification of first class agent interaction protocols.** In Proc. of ESAW VII, pages 245–264, Berlin, Heidelberg, 2007. Springer.

MILLER, T.; McGINNIS J**. Amongst first-class protocols.** In Proc. of ESAW VIII, pages 208–223. Springer-Verlag, Berlin, Heidelberg, 2008.

NANCY, Ruiz; GIRET, Adriana; BOTTI, Vicente. **Using an Agent-Supported Simulation Environment for Intelligent Manufacturing Systems.** Lecture Notes in Computer Science, 5696:124-134, 2009.

NWANA, S.; NDUMU, D.T.; LEE, L.C.; COLLIS, J.C. **Zeus: A Toolkit for Building Distributed Multi-Agent Systems**. 3th International Conference on Autonomous Agents, Seattle, WA, USA (1999)

OMICINI, A.; ZAMBONELLI, F.; KLUSCH M.; TOLKSDORF, R. (Eds.), **Coordination of Internet Agents: Models, Technologies, and Applications.** Springer-Verlag, 2001.
URL http://www.springer.com/computer/communications/book/978-3-540-41613-5

PADGHAM Lin; WINIKOFF, Michael. **Prometheus: A methodology for Developing Intelligent Agents**. 3th Agent-Oriented Software EngineeringWorkshop, Bologna, Italy, 2002.

PARUNAK, V.; **The AARIA Agent architecture: from manufacturing requirements to agent-based system design.** Integrated Computer-Aided Engineering 8(1), 2001.

PAVIM, Alberto Xavier. **Gestaltung von selbstoptimierenden Inspektionssystemen zur qualitativen Absicherung von wirtschaftlichen Kleinserien.** Doctoral thesis, University of Aachen - RWTH-Aachen, 2011.

PAVON, J.; GOMEZ-SANZ, J.J.; FUENTES, R. **The Ingenias Methodology and Tools.** In: HENDERSON-SELLERS, B., GIORGINI, P. (Ed.), Agent-Oriented Methodologies, Idea Group Publishing, p. 236-276, 2005.

PECHOUCEK, M.; MARIK, V**. Industrial Deployment of Multi-agent Technologies: Review and Selected Case Studies.** Autonomous Agents and Multi-agent Systems 17(13), 397–431, 2008.

PFEIFER, T.; SCHMITT, R.; PAVIM, A.; STEMMER, M.; ROLOFF, M. L.; SCHNEIDER, C.; DORO, M. M. Cognitive Production Metrology: A new concept for flexibly attending the inspection requirements of small series production. Proceedings of the 36th International MATADOR Conference 2010, pp 359-362. Print ISBN: 978-1-84996-431-9. Online ISBN: 978-1-84996-432-6. Publisher: Springer London. Copyright Holder: Springer-Verlag London. DOI: 10.1007/978-1-84996-432-6_81

PFEIFER, T.; MERGET, M. **Methodology to estimate the feasibility of Inspection tasks in Concurrent Engineering (CE).** In: Proceedings of the International Conference on TQM and Human Factors, Linköping: Centre for Studies of Humans, Technology and Organization, 1999, 15-17 Juni S. 194-199.

PICARD, G.; HUBNER, J.F.; BOISSIER, O.; GLEIZES, M-P. **Réorganisation et autoorganisation dans les Système Multi-agents.** In: GUESSOUM, Z. ; HAMAS, S. (Ed.), Journées Francophones sur les

systèmes multi-agents - Génie Logiciel Multi-agent (JFSMA09), France, Cepadués Editions, p.89-97, 2009.

POSLAD, S.; BUCKLE, P.; HADINGHAM, R. **The FIPA-OS Agent Platform: Open Source for Open Standards**, 2000. Available on: <http://fipaos.sourceforge.net/docs/papers/FIPAOS.pdf>. Accessed: 03/2012.

PYNADATH, D. V.; TAMBE, M.; CHAUVAT, N.; CAVEDON, L. **Toward team-oriented programming,** in: JENNINGS, N. R.; LESPÉRANCE, Y. (Eds.), ATAL, Vol. 1757 of LNCS, Springer, 1999, pp. 233-247.

PYZDEK, T. **Process Control for Short and Small Runs.** Quality Progress, Pgs. 51-60, 1993.

RAO, A. S.; GEORGEFF, V. **An abstract architecture for rational agents.** In: RICH, C.; SWARTOUT, W.; NEBEL, B. (Ed.), Proceedings of Knowledge Representation and Reasoning, p.439-449, 1992.

RICORDEL, P.; DEMAZEAU, Y. **Volcano, a vowels-oriented multi-agent platform.** In: From Theory to Practice in Multi-Agent Systems, Second International Workshop of Central and Eastern Europe on Multi-Agent Systems (CEEMAS 2001), LNCS v. 2296, Springer, p-253-262, 2002.

RICCI, A.; PUNTI, M.; VIROLI, M**. Environment programming in multiagent systems - an artifact-based perspective,** in Autonomous Agents and Multi-Agent SystemsSpecial Issue on Programming Multi-Agent Systems. DOI:10.1007/s10458-010-9140-7. p. 158-192 (2011)

RICCI, A.; VIROLI, M.; OMICINI, A. **CArtAgO: A framework for prototyping artifact-based environments in MAS**, in: WEYNS, D.; PARUNAK, H. Van D.; MICHEL, F. (Eds.), Environments for MultiAgent Systems III, Vol. 4389 of LNAI, Springer, 2007, pp. 67-86, 3rd International Workshop (E4MAS2006), Hakodate, Japan, 8 May 2006. Selected Revised and Invited Papers. URL http://www.springerlink.com/content/d1047836893j14q6/

RUSSEL, S.; NORVIG, P. **Artificial Intelligence – A Modern Approach.** 2nd Edition, Prentice Hall, 2003.

SAUTER, J.; PARUNAK, V. **ANTS in the supply chain.** In Proceedings of the Workshop on Agent-Based Decision Support Managing Internet-Enabled Supply Chain, Seattle, WA, pp. 1–9, 1999.

SCHUH, G.; KLOCKE, F.; BRECHER, C.; SCHMITT, R. **Excellence in Production**. 1st edition, Aachen: Apprimus-Verlag, 2007.

SETCHI, R. M. & LAGOS, N. **Reconfigurability and reconfigurable manufacturing systems: state-of-the-art review.** 2nd IEEE International Conference on Industrial Informatics INDIN 2004, p. 529 – 535, June 2004.

SHOHAM, Y. **Agent-Oriented Programming**. Artificial Intelligence. pp. 51–92. 1993.

SHEN, Weiming; HAO, Qi; YOON, Hyun Joong; NORRIE, Douglas H. **Applications of agent-based systems in intelligent manufacturing: An updated review.** Advanced Engineering Informatics, 20(4):415-431, Oktober 2006.

SICHMAN, J. S.; DEMAZEAU, Y. **On social reasoning in multi-agent systems.** In: Revista Iberoamericana de Inteligencia Artificial, v. 13, p. 68–84, 2001.

SILVA, V. T.; LUCENA, C. J. P. From **a Conceptual Framework for Agents and Objects to a Multi-Agent System Modeling Language.** In: Autonomous Agents and Multi-Agent Systems, n. 9, Kluwer Academic Publishers, p. 45–189, 2004.

SILVA, E. L. D.; MENEZES, E. M. **Metodologia da Pesquisa e Elaboração de Dissertação.** Editora da UFSC, 2005.

SINGH, M. P. **Information-driven interaction-oriented programming: BSPL, the blindingly simple protocol language.** In Proc. of AAMAS, pages 491–598, 2011.

SLACK, N., CHAMERS, S., et al. **Administração da Produção.** 1.ed. São Paulo: Atlas. 726p. 1997.

SPANOUDAKIS, N. ; MORAITIS, P. **Model-Driven Agents Development with ASEME.** In: GLEIZES, M-P.; WEYNS, D. (Ed.), Proceedings of the Eleventh International Workshop on Agentoriented Software Engineering (AOSE 2010) at AAMAS 2010, Toronto, Canada, p.49-60, 2010.

SPROVIERI, J. **Managing High-Mix, Low-Volume.** Assembly March 5, 2004. Disponível em: <http://www.assemblymag.com/CDA/Archives/0a26022c106c9010VgnVCM100000f932a8c0____> Acesso em: 14 nov. 2011.

STEELS, L. **Cooperation between distributed agents through self-organization.** In: DEMAZEAU, Y.; MULLER, J.-P. (Ed.), Decentralized AI: Proceedings of the First European Workshop on Modeling Autonomous Agents in a Multi-Agent World, Amsterdam, Elsevier Science Publishers B V, p. 175-196, 1990.

STEMMER, MARCELO RICARD ; DA COSTA, CAMILA PONTES BRITO ; VARGAS, JAQUELINE ; ROLOFF, MÁRIO LÚCIO . **Artificial Intelligent Systems for Quality Assurance in Small Series Production**. Key Engineering Materials (Online), v. 613, p. 279-287, 2014.

STEMMER, Marcelo R. ; SZYMANSKI, Charbel ; MELO, Daniel F. F.; ROLOFF, Mário L. **A Machine Vision System for Quality Assurance in Small Series Production.** In: 11th International Symposium on Measurement Technology and Intelligent Instruments (ISMTII), 2013, Aachen. 11th International Symposium on Measurement Technology and Intelligent Instruments (ISMTII), 2013.

STORCH, R. L.; LIM, S. **Improving Flow to Achieve Lean Manufacturing in Shipbuilding**. Production Planning and Control, vol. 10, no. 2, pp. 127–137, 1999.

STRATULAT, T.; FERBER, J.; TRAINER, J. **MASQ: towards an integral approach to interaction**, in: AAMAS (2009), pp. 813-820, 2009.

STURM, A.; SHEHORY, O. **A comparative evaluation of Agent-Oriented Methodologies.** In: BERGENTI, F.,

GLEIZES, M., ZAMBONELLI, F.(Ed.) Methodologies and Software Engineering for Agent Systems, Kluwer Academic Publishers, p. 127-147, 2004.

SYCARA, K.; PAOLUCCI, M.; VAN VELSEN, M.; GIAMPAPA, J. **The Retsina MAS Infrastructure**. Kluwer Academic Publishers. 2001.

TANG, P. F.; BARNETT, N. **A comparison of mean and range charts with pre-control having particular reference to short-run production.** 10(6):477-485, 1994.

TICHÝ, Pavel; KADERA, Petr; STARON, Raymond J.; VRBA, Pavel; MARÍK, Vladimír. **Multi-agent system design and integration via agent development environment.** Engineering Applications of Artificial Intelligence. Volume 25, Issue 4, June 2012, Pages 846–852, 2012. DOI: 10.1016/j.engappai.2011.09.021

TRAN, Q. N.; LOW, G. **Comparison of ten Agent-Oriented Methodologies.** In: HENDERSON-SELLERS, B., GIORGINI, P. (Ed.) Agent-Oriented Methodologies, Idea Group Publishing, p. 341-367, 2005.

TRENCANSKY, Ivan; CERVENKA, Radovan. Agent Modeling Language (AML): **A Comprehensive Approach to Modeling MAS.** Informatica - An International Journal of Computing and Informatics, 29:391-400, 2005.

UEZ, D. M.; HÜBNER, J. F. **Environments and organizations in multi-agent systems: From modelling to code**. In Fabiano Dalpiaz, Jürgen Dix, and M. Birna van Riemsdijk, editors, Proc. 2nd International Workshop on Engineering Multi-agent Systems (EMAS @ AAMAS 2014), pages 162-178, 2014.

VALCKENAERS, P.; BRUSSEL, H. V.; WYNS, J.; PEETERS, P.; BONGAERTS, L. **Multi-agent manufacturing control in holonic manufacturing systems.** In Human Management Systems. Amsterdam, The Netherlands: IOS Press, 1999, vol. 18.

VARGAS, J. **Sistema Especialista para a Produção de Pequenas Séries.** Dissertação de mestrado do Programa de Pós-Graduação em Engenharia de Automação e Sistemas, DAS, UFSC, 2012.

VERNADAT, F.B. **Enterprise modeling and integration: principles and applications.** London, Chapman & Hall. 1996.

WAGNER, Thomas. **Agentenunterstütztes Engineering von Automatisierungsanlagen.** PhD Thesis, Universität Stuttgart, 2008.

WEISS, Gerhard; JAKOB, Ralf**. Agentenorientierte Softwareentwicklung: Methoden und Tools.** Springer, 2005.

WEISS, G. (Org.) **Multiagent Systems: A modern approach to distributed artificial intelligence.** The MIT Press, 2001.

WEYNS, D.; PARUNAK, H. Van D.; MICHEL, F.; HOLVOET, T.; FERBER, J. **Environments for Multiagent Systems State-of-the-Art and Research Challenges**. E4MAS 2004, LNAI 3374, pp. 1–47, 2005. Springer-Verlag Berlin Heidelberg, 2005.

WHEELER, D. J. **Short Run SPC**. Knoxville, TN: SPC Press Inc, 1991. 62 p.

WOMACK, J. P.; JONES, D. T.; RODOS, D. **A mentalidade enxuta das empresas: elimine o desperdício e crie riquezas**. Rio de Janeiro : Campus, 1998. 427p.

WOOD, M. F.; DELOACH, S. A. **An overview of the Multiagent Systems Engineering Methodology.** In: CIANCARINI, P.; WOOLDRIDGE, M.(Ed.), Proceedings of the First International Workshop on Agent-Oriented Software Engineering, Lecture Notes in Computer Science (LNCS), Springer-Verlag, Berlin, v. 1957, p. 207-221, 2001.

WOOLDRIDGE, P. **An Introduction to Multiagent Systems.** Addison-Wesley, Reading, MA. 2000.

WOOLDRIDGE, P.; JENNINGS, N. R. Software Engineering with Agents, Pitfalls and Pratfalls. In:
IEEE Internet Computing, v. 3, n.3, p. 20-27, 1999.

ZAEH, Michael F.; REINHART, Gunther; OSTGATHE, Martin; GEIGER, Florian; LAU, Christian. **A holistic approach for the**

cognitive control of production systems. Advanced Engineering Informatics, 24(3):300-307, August 2010.

ZAMBONELLI, F.; JENNINGS, N. R.; WOOLDRIDGE, M. **Organisational Abstractions for the Analysis and Design of Multi-Agent Systems.** In: Proceedings of the First International Workshop on Agent-Oriented Software Engineering: (AOSE 2000), Limerick, Ireland, Springer Berlin / Heidelberg, v. 1957/2001, p. 407-422, 2001.

ZATELLI, M. R.; HÜBNER, J. F. **The Interaction as an Integration Component for the JaCaMo Platform**. AAMAS´14.

ZATTAR I. C. **Metodologia para implantação de um sistema de programação da produção com capacidade finita em empresas prestadoras de serviços.** Campinas, 2003, 201 f. Tese (Doutorado em Engenharia Mecânica), Sociedade Educacional de Santa Catarina.

ZHU; Y. D. **A framework of computer-aided short-run SPC planning system.** Singapore, 2005. MEng thesis, National University of Singapore.

APÊNDICE A – Descritores dos Agentes

Tabela 26 – Refinamento do agente *planner*.

Agente *planner*	
Descrição:	este agente recebe o produto a ser produzido da interface com o usuário a partir do início da operação do Sistema Multiagente. Após receber o produto, o agente deverá criar uma lista ordenada dos links entre a máquinas da linha de produção para fabricar o lote do produto. Ele também é responsável pela criação da organização e da interface com o usuário na inicialização da plataforma multiagente.
Cardinalidade:	1 (propõem-se que cada linha possua um agente *Planner*)

Tabela 27 – Refinamento do agente *configurator*.

Agente *configurator*	
Descrição:	este agente tem como objetivos: selecionar e criar as máquinas e os links entre elas para produzir o lote do produto selecionado; realizar o *setup* de cada máquina de acordo com o nome do produto. Para se comunicar com cada equipamento da linha de produção e realizar o *setup* de cada um de acordo com o arquivo de configuração de cada equipamento - que deve estar no equipamento relacionado com o nome do produto - este agente se comunicará com cada equipamento real (artefato) através do SCADA usando *Web Service*.
Cardinalidade:	1 (propõem-se que cada linha possua um agente *Configurator*)

Tabela 28 – Refinamento do agente *assembler*.

Agente *assembler*	
Descrição:	este agente tem como objetivo monitorar o processo de fabricação do item do lote do produto selecionado. Após todos os equipamentos estarem disponíveis para produção o lote de produtos pode ser carregado na linha de produção. Esta atividade é realizada pelo artefato *assembler*. O agente envia a ordem de carregamento do lote de produtos no primeiro equipamento e supervisiona a produção até o armazenamento de todos os produtos produzidos no último equipamento. A qualquer momento ele sabe o status de cada um dos equipamentos (livre, ocupado, falha).
Cardinalidade:	1 (propõem-se que cada linha possua um agente *Assembler*)

Tabela 29 – Refinamento do agente *inspector*.

Agente *inspector*	
Descrição:	o objetivo do agente inspector é receber da máquina de inspeção visual automática (que é um artefato) um diagnóstico - OK ou indicação de defeitos. De posse deste resultado o agente deve se comunicar com o agente de estatística que após avaliar os defeitos requisitará ao especialista que alterações no SETUP dos equipamentos devem ser realizadas para que os defeitos não ocorram novamente.
Cardinalidade:	n (sugere-se um agente para cada máquina de inspeção)

Tabela 30 – Refinamento do agente *statistic*.

Agente *statistic*	
Descrição:	este agente faz um relatório dos dados da produção: quantidades produzidas no lote; lista dos defeitos do lote, etc.
Cardinalidade:	1 (propõem-se que cada linha possua um agente *Statistic*)

Tabela 31 – Refinamento do agente *expert*.

Agente *expert*	
Descrição:	de posse do defeito encontrado pela máquina de inspeção, o agente de qualidade avalia as possíveis causas deste defeito e sugere alterações no SETUP dos equipamentos para que os defeitos não ocorram novamente.
Cardinalidade:	1 (propõem-se que cada linha possua um agente *Expert*)

APÊNDICE B – Descritores dos Planos

Tabela 32 – Refinamento do plano *createOrganization*.

Plano *createOrganization*	
Descrição:	este plano busca criar a organização conforme o arquivo XML que foi construído com base no resultado da modelagem realizada em Prometheus AEOlus. O arquivo XML é construído seguindo o *framework* MOISE.
Contexto:	
Agente:	*Planner*

Tabela 33 – Refinamento do plano *createGUI*.

Plano *createGUI*	
Descrição:	este plano cria a interface gráfica, que pode fazer parte da plataforma JaCaMo ou ser um sistema legado. O agente *Planner* recebe o produto a ser produzido da interface do usuário.
Contexto:	
Agente:	*Planner*

Tabela 34 – Refinamento do plano *receiveProduct*.

Plano *receiveProduct*	
Descrição:	este plano aguarda da interface gráfica o produto a ser produzido. O agente Planner recebe o produto a ser produzido da interface do usuário e passa a utilizar suas infos. no SMA.
Contexto:	
Agente:	*Planner*

Tabela 35 – Refinamento do plano *createLinksList*.

Plano *createLinksList*	
Descrição:	o plano cria uma lista de *links* que são necessários para criar a linha de produção para o produto a ser produzido, cada produto tem uma lista específica, ou seja, tem uma sequência de máquinas apropriada
Contexto:	
Agente:	*Planner*

Tabela 36 – Refinamento do plano *goalGX[atomic]*.

Plano *goalGX[atomic]*	
Descrição:	esses são os planos organizacionais que existem para acompanhar o comprometimento do agente *Planner* com as missões da organização. Quando as metas referentes a missão com a qual se comprometeu forem atingidas a missão é cumprida.
Contexto:	
Agente:	*Planner*

Tabela 37 – Refinamento do plano *adoptRoleConfigurator*.

Plano *adoptRoleConfigurator*	
Descrição:	Este é o plano organizacional do configurador. O plano indica com qual grupo o configurador se compromete e então recebe a missão com a qual deve se comprometer segundo a implementação no *framework* MOISE
Contexto:	
Agente:	*Configurator*

Tabela 38 – Refinamento do plano *prepareMachinesToProduction*.

Plano *prepareMachinesToProduction*	
Descrição:	baseado no ID do produto, no tamanho do lote e na especificação de quais equipamentos utilizar o agente *configurator* cria a linha de produção para o produto selecionado.
Contexto:	
Agente:	*Configurator*

Tabela 39 – Refinamento do plano *goalGX[atomic]*.

Plano *goalGX[atomic]*	
Descrição:	esses são os planos organizacionais que existem para acompanhar o comprometimento do agente *Configurator* com as missões da organização. Quando as metas referentes a missão com a qual se comprometeu forem atingidas a missão é cumprida.
Contexto:	
Agente:	*Configurator*

Tabela 40 – Refinamento do plano *updateGUI*.

Plano *updateGUI*	
Descrição:	Avisa a interface que o status da linha deve ser atualizado, as máquinas foram configuradas para produção do lote
Contexto:	
Agente:	*Configurator*

Tabela 41 – Refinamento do plano *createLine*.

Plano *createLine*	
Descrição:	este plano cria a linha de produção para o produto selecionado baseado nas especificações de tipo de máquinas e forma como devem ser ligadas para formar a linha de produção.
Contexto:	
Agente:	*Configurator*

Tabela 42 – Refinamento do plano *runMachine*.

Plano *runMachine*	
Descrição:	neste plano a máquina permanece em um estado onde já está apta para realizar a sua operação. Aguarda pelo carregamento de um item do lote para então entrar em operação.
Contexto:	
Agente:	*Configurator*

Tabela 43 – Refinamento do plano *linkMachinesArtifacts*.

Plano *linkMachinesArtifacts*	
Descrição:	neste plano os artefatos são criados no CArtAgO e viculados com a respectiva máquina via SCADA para aquisição dos dados da linha de produção.
Contexto:	
Agente:	*Configurator*

Tabela 44 – Refinamento do plano *createLinkBetweenMachines*.

Plano *createLinkBetweenMachines*	
Descrição:	este plano auxilia no cumprimento plano *linkMachinesArtifacts* cada máquina é ligada com a máquina anterior e posterior na linha especificada. No final a linha de produção está criada.
Contexto:	
Agente:	*Configurator*

Tabela 45 – Refinamento do plano *loadBatchOnLine*.

Plano *loadBatchOnLine*	
Descrição:	este plano carrega o número de itens do lote do produto selecionado no primeiro equipamento da linha construída.
Contexto:	
Agente:	*Assembler*

Tabela 46 – Refinamento do plano *watchProduction*.

Plano *watchProduction*	
Descrição:	este plano acompanha a produção do lote - fluxo dos itens do lote do produto selecionado pelas máquinas.
Contexto:	
Agente:	*Assembler*

Tabela 47 – Refinamento do plano *endBatch*.

Plano *endBatch*	
Descrição:	este plano acompanha a finalização do lote.
Contexto:	
Agente:	*Assembler*

Tabela 48 – Refinamento do plano *goalGX[atomic]*.

Plano *goalGX[atomic]*	
Descrição:	esses são os planos organizacionais que existem para acompanhar o comprometimento do agente *Assembler* com as missões da organização. Quando as metas referentes a missão com a qual se comprometeu forem atingidas a missão é cumprida.
Contexto:	
Agente:	*Assembler*

Tabela 49 – Refinamento do plano *adoptRoleAssembler*.

Plano *adoptRoleAssembler*	
Descrição:	plano para o agente assembler se comprometer com as metas organizacionais adotando o papel ao qual está destinado.
Contexto:	
Agente:	*Assembler*

Tabela 50 – Refinamento do plano *createAssemblerArtifact*.

Plano *createAssemblerArtifact*	
Descrição:	plano para a criação do artefato para auxiliar o agente *assembler* nas ações sobre o *workspace* PRODUCTION.
Contexto:	
Agente:	*Assembler*

Tabela 51 – Refinamento do plano *startBatch*.

Plano *startBatch*	
Descrição:	este plano inicializa a produção do lote – define a quantidade de itens do lote do produto selecionado a serem produzidos pelas máquinas da linha de produção.
Contexto:	
Agente:	*Assembler*

Tabela 52 – Refinamento do plano *observeMachineStatus*.

Plano *observeMachineStatus*	
Descrição:	observa o estado de uma determinada máquina da linha de produção criada.
Contexto:	
Agente:	*Assembler*

Tabela 53 – Refinamento do plano *adoptRoleInspector*.

Plano *adoptRoleInspector*	
Descrição:	plano organizacional para se comprometer com as missões definidas no *framework* MOISE.
Contexto:	
Agente:	*Inspector*

Tabela 54 – Refinamento do plano *observeVisionSystemArtifact*.

Plano *observeVisionSystemArtifact*	
Descrição:	caso o sistema de visão esteja presente o agente procura pelo artefato e foca nele esperando uma resposta sobre a inspeção do item do lote.
Contexto:	
Agente:	*Inspector*

Tabela 55 – Refinamento do plano *defectDetected*.

Plano *defectDetected*	
Descrição:	quando o artefato sistema de visão detecta uma falha no item do lote esta falha deve ser informada ao agente responsável.
Contexto:	
Agente:	*Inspector*

Tabela 56 – Refinamento do plano *updateDefectsStatitics*.

Plano *updateDefectsStatitics*	
Descrição:	identificado um defeito pelo inspector que avisa o statistic para atualizar a contagem e a lista de defeitos encontrados no lote.
Contexto:	
Agente:	*Statistic*

Tabela 57 – Refinamento do plano *writeBatchStatistics*.

Plano *writeBatchStatistics*	
Descrição:	divulgar os dados sobre a produção do lote, como a contagem e a lista de defeitos encontrados no lote.
Contexto:	
Agente:	*Statistic*

Tabela 58 – Refinamento do plano *writeBatchErrors*.

Plano *writeBatchErrors*	
Descrição:	divulgar a lista de defeitos encontrados no lote.
Contexto:	
Agente:	*Statistic*

Tabela 59 – Refinamento do plano *goalGX[atomic]*.

Plano *goalGX[atomic]*	
Descrição:	esses são os planos organizacionais que existem para acompanhar o comprometimento do agente *Assembler* com as missões da organização. Quando as metas referentes a missão com a qual se comprometeu forem atingidas a missão é cumprida.
Contexto:	
Agente:	*Statistic*

Tabela 60 – Refinamento do plano *adoptRoleStatistics*.

Plano *adoptRoleStatistics*	
Descrição:	plano organizacional para se comprometer com as missões definidas no *framework* MOISE.
Contexto:	
Agente:	*Statistics*

Tabela 61 – Refinamento do plano *adoptRoleQualifying*.

Plano *adoptRoleQualifying*	
Descrição:	plano organizacional para se comprometer com as missões definidas no *framework* MOISE.
Contexto:	
Agente:	*Expert*

Tabela 62 – Refinamento do plano *defectReceivedFromInspector*.

Plano *defectReceivedFromInspector*	
Descrição:	com o defeito recebido do agente inspector o agente pode indicar para o usuário as possíveis causas do defeito e que ações tomar para evitar que o defeito se repita.
Contexto:	
Agente:	*Expert*

APÊNDICE C – Descritores das Percepções

Tabela 63 – Refinamento da percepção *product_received*.

Percepção *product_received*	
Descrição:	recebe a informação sobre qual a linha que deve ser utilizada para a fabricação do lote e o tamanho do lote. Esta percepção provém da interface com o usuário.
Informação:	[numDaLinha,tamanhoDoLote]
Agente:	*Planner*

Tabela 64 – Refinamento da percepção *id_prod*.

Percepção *id_prod*	
Descrição:	recebe a informação sobre qual é o código do produto que será fabricado.
Informação:	[identificaçãoDoProduto]
Agente:	*Planner*

Tabela 65 – Refinamento da percepção *qtd_prod*.

Percepção *qtd_prod*	
Descrição:	recebe a informação sobre qual é o tamanho do lote.
Informação:	[tamanhoDoLote]
Agente:	*Planner*

Tabela 66 – Refinamento da percepção *idProduct*.

Percepção *idProduct*	
Descrição:	recebe a informação sobre qual é o código do produto que será fabricado.
Informação:	[identificaçãoDoProduto]
Agente:	*Configurator*

Tabela 67 – Refinamento da percepção *numero*.

Percepção *numero*	
Descrição:	recebe a informação sobre qual é o tamanho do lote.
Informação:	[tamanhoDoLote]
Agente:	*Configurator*

Tabela 68 – Refinamento da percepção *setup*.

Percepção *setup*	
Descrição:	percepção sobre a configuração de uma determinada máquina para o produto selecionado.
Informação:	[numeroMaquina, idProduto]
Agente:	*Configurator*

Tabela 69 – Refinamento da percepção *run*.

Percepção *run*	
Descrição:	carregamento dos itens do lote a produzir.
Informação:	[tamanhoDoLote]
Agente:	*Assembler*

Tabela 70 – Refinamento da percepção *termino*.

Percepção *termino*	
Descrição:	os itens do lote a produzir estão na última máquina.
Informação:	[flagIdentificaFimDoLote]
Agente:	*Assembler*

Tabela 71 – Refinamento da percepção *start*.

Percepção *start*	
Descrição:	começa a fabricação do lote.
Informação:	[tamanhoDoLote]
Agente:	*Assembler*

APÊNDICE D – Descritores das Ações

Tabela 72 – Refinamento da ação *gX*.

Ação *gX*	
Descrição:	várias das ações dos agentes seguem a nomenclatura definida para as metas apresentadas no arquivo XML do *framework* MOISE. Exemplo: <goal id="gX" min="X" ds="receive product to production"/> ▪ g1: receber o produto para produzir ▪ g2: selecionar a lista de máquinas ▪ g3: selecionar e criar a lista de máquinas e links ▪ g4: realizar o setup das máquinas para o produto ▪ g5: carregar os produtos a serem produzidos ▪ g6: acompanhar a produção do lote ▪ g7: final do lote ▪ g8: divulgar as estatísticas do lote ▪ g9: divulger a lista de defeitos do lote
Função:	como especificado no arquivo da organização
Agente:	*Assembler, Configurator, Expert, Inspector, Planner, Statistic*

APÊNDICE E – Descritores das Crenças

Tabela 73 – Refinamento da crença *machineList*.

Crença *machineList*	
Descrição:	lista de máquinas a serem criadas baseadas
Informação:	machineList([machine("L1","artifacts.machines.Loader"), machine("PP1","artifacts.machines.PastePrinter"), machine("PaP1","artifacts.machines.PickAndPlaceStation"), machine("U1","artifacts.machines.Unloader"), machine("VS1","artifacts.machines.VisionSystem"), machine("RO1","artifacts.machines.ReflowOven")]).
Agente:	*Planner*

Tabela 74 – Refinamento da crença *list_id_link*.

Crença *list_id_link*	
Descrição:	lista de links entre as máquinas da linha, o agente *Assembler* por meio de um artefato *Assembler* é que inicia e termina a lista pois este artefato (e o agente) é o responsável pelo carregamento do lote e acompanhar a sua conclusão na linha
Informação:	list_id_link([[001,[link("L1","assembler","PP1"), link("PP1","L1","PaP1"),link("PaP1","PP1","U1"), link("U1","PaP1","assembler"), link("assembler","U1","L1")]], [002,[link("L1","assembler","PP1"), link("PP1","L1","PaP1"),link("PaP1","PP1","VS1"), link("VS1","PaP1","U1"), link("U1","VS1","assembler"), link("assembler","U1","L1")]], [003,[link("L1","assembler","PP1"), link("PP1","L1","PaP1"),link("PaP1","PP1","RO1"), link("RO1","PaP1","VS1"), link("VS1","RO1","U1"), link("U1","VS1","assembler"), link("assembler","U1","L1")]]]).
Agente:	*Planner*

Tabela 75 – Refinamento da crença *termino*.

Crença *termino*	
Descrição:	não há nenhum item do lote no equipamento final, ou seja, nenhum produto finalizado do lote
Informação:	Flag: 0 = não finalizado; 1 = finalizado o lote;
Agente:	*Assembler*

APÊNDICE F – Descritores das Mensagens

A seguir estão descritas mensagens do MAS@SSP além das mensagens para a criação e gestão da organização.

Tabela 76 – Refinamento da mensagem *prepareMachines*.

Mensagem *prepareMachines*	
Descrição:	mensagem do *Planner* para o *Configurator* no qual ele envia o identificado do produto a produzir, a lista de máquinas e os links entre as máquinas para formar a linha de produção para o lote
Propósito:	*achieve*
Conteúdo:	prepareMachines(idenficadorProduto,listaLinks,listaMáquinas)
Agente:	*Planner*

Tabela 77 – Refinamento da mensagem *prepareMachines*.

Mensagem *prepareMachines*	
Descrição:	mensagem do *Planner* para o *Configurator* no qual ele envia o identificado do produto a produzir, a lista de máquinas e os links entre as máquinas para formar a linha de produção para o lote
Propósito:	*achieve*
Conteúdo:	prepareMachines(idenficadorProduto,listaLinks,listaMáquinas)
Agente:	*Planner*

Tabela 78 – Refinamento da mensagem *updateDefectsCount*.

Mensagem *updateDefectsCount*	
Descrição:	mensagem do *Inspector* para o *Statistic* após a detecção de um defeito pelo artefato *VisionSystem*
Propósito:	*achieve*
Conteúdo:	updateDefectsCount (tipoDefeito)
Agente:	*Inspector*

Tabela 79 – Refinamento da mensagem *defectFound*.

Mensagem *defectFound*	
Descrição:	mensagem do *Statistic* para o *Expert* sobre o defeito detectado no artefato *VisionSystem*, recebido do agente *Inspector*
Propósito:	*achieve*
Conteúdo:	defectFound(tipoDefeito)
Agente:	*Statistic*

Tabela 80 – Refinamento da mensagem *doCorrectionAction*.

Mensagem *doCorrectionAction*	
Descrição:	mensagem do *Expert* para o *Configurator* dizendo para alterar a configuração de uma máquina para evitar novos defeitos
Propósito:	*achieve*
Conteúdo:	doCorrectionAction(máquina,ação)
Agente:	*Expert*

APÊNDICE G – Descritores dos Artefatos

Tabela 81 – Descrição do artefato – *Interface*.

Artefato *Interface*	
Descrição:	Esse artefato é utilizado para visualizar o estado do SMA no controle da PPS. Na Arquitetura de Referência a interface com o usuário pode ser desenvolvida como um artefato Java dentro da própria plataforma JaCaMo, ou pode ser um sistema a parte que utiliza o artefato *Interface* como meio de comunicação com o SMA. O mecanismo para esta comunicação sugerido é o *Web Service*.
Operações:	selectProduct(Product aProduct) refreshGUI(String tag, double tagValue)
Parâmetros:	String tag, double tagValue
PropriedadesObserváveis:	tag tagValue product
Eventos Observáveis:	

Tabela 82 – Descrição do artefato – *Assembler*.

Artefato *Assembler*	
Descrição:	Esse artefato é um mecanismo para viabilizar as ações e percepções do agente *Assembler* no ambiente da linha de produção. Entre todos os agentes propostos, o agente *Assembler* tem a responsabilidade por realizar o início da produção de um lote e carregar o lote na primeira máquina da linha. O agente Assembler cria o link com a primeira e a última máquina da linha.
Operações:	init() run(int batchSize) operate() inc() readyForLoad() getBatch(OpFeedbackParam<Stack<Product> > v) enableLoad()
Parâmetros:	int count; boolean loadEnabled; int batch_size;
PropriedadesObserváveis:	count
Eventos Observáveis:	

Tabela 83 – Descrição do artefato – *Machine*.

Artefato *Machine*	
Descrição:	Esse artefato é a generalização dos recursos não autônomos da linha de produção. O artefato Machine pode representar qualquer equipamento da linha de produção, a partir dele são especializados cada equipamento. Desta forma, um único artefato é representado na Arquitetura de Referência, já no modelo de implementação o artefato geral é especializado nos equipamentos da linha de produção a ser controlada pelo SMA.
Operações:	init() setup(int batchSize, int productID) run() stop() pause() stateMachine() readyToLoad() notReadyToLoad() isUnloaded() returnFromPause() enableLoad() getProduct(OpFeedbackParam<Product> v) load() unload() operate() pauseOp() stopOp()
Parâmetros:	enum status; int batchSize; int productID; Product currentProduct;
PropriedadesObserváveis:	status
Eventos Observáveis:	

APÊNDICE H – Arquivo XML da Organização MAS@SSP

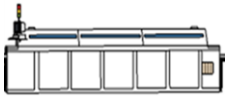
```
<?xml version="1.0" encoding="UTF-8"?>
<organisational-specification
    id="joj"
    os-version="0.6"
    xmlns='http://moise.sourceforge.net/os'
    xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
    xsi:schemaLocation='http://moise.sourceforge.net/os
    http://moise.sourceforge.net/xml/os.xsd'>
<structural-specification>
    <role-definitions>
        <role id="rSmallSeriesManager"/>
        <role id="rSmallSeriesShopFloor"/>
        <role id="rPlanning">
          <extends role="rSmallSeriesManager"/>
        </role>
        <role id="rStatistic">
          <extends role="rSmallSeriesManager"/>
        </role>
        <role id="rExpert">
          <extends role="rSmallSeriesManager"/>
        </role>
        <role id="rSetup">
          <extends role="rSmallSeriesShopFloor"/>
        </role>
        <role id="rAssembly">
          <extends role="rSmallSeriesShopFloor"/>
        </role>
        <role id="rInspection">
          <extends role="rSmallSeriesShopFloor"/>
        </role>
    </role-definitions>
    <group-specification id="grCompany">
        <links>
          <link from="rSmallSeriesManager"
           to="rSmallSeriesShopFloor" type="authority"
           scope="inter-group" extends-subgroups="true" bi-
           dir="false"/>
          <link from="rSmallSeriesShopFloor"
           to="rSmallSeriesShopFloor" type="communication"
           scope="inter-group" extends-subgroups="true" bi-
           dir="false"/>
          <link from="rInspection" to="rAssembly"
           type="authority" scope="intra-group" extends-
           subgroups="true" bi-dir="false"/>
          <link from="rSetup" to="rAssembly" type="authority"
           scope="intra-group" extends-subgroups="true" bi-
           dir="false"/>
          <link from="rExpert" to="rStatistic" type="authority"
           scope="intra-group" extends-subgroups="true" bi-
           dir="false"/>
        </links>
        <subgroups>
```

```
    <group-specification id="grManagers" min="1">
        <roles>
          <role id="rPlanning" min="1" max="1"/>
          <role id="rStatistic" min="1" max="1"/>
          <role id="rExpert" min="1" max="1"/>
        </roles>
    </group-specification>
    <group-specification id="grShopFloor" min="1">
        <roles>
          <role id="rSetup" min="1" max="3"/>
          <role id="rAssembly" min="1" max="10"/>
          <role id="rInspection" min="1" max="3"/>
        </roles>
    </group-specification>
    </subgroups>
    </group-specification>
</structural-specification>
<functional-specification>
    <scheme id="production">
        <goal id="smallSeriesProductionControl" min="1">
          <plan operator="sequence">
            <goal id="pp" ds="production_planning">
              <plan operator="sequence">
                <goal id="g1" min="1" ds="receive product to production"/>
                <goal id="g2" min="1" ds="select links for machines"/>
              </plan>
            </goal>
            <goal id="ps" ds="production_setup">
              <plan operator="sequence">
                <goal id="g3" min="1" ds="select and create machines and links"/>
                 <goal id="g4" min="1" ds="setup line machines to produce"/>
              </plan>
            </goal>
            <goal id="pa" ds="production_assembly">
              <plan operator="sequence">
                 <goal id="g5" min="1" ds="load itens on loader and start"/>
                 <goal id="g6" min="1" ds="watch out production"/>
                 <goal id="g7" min="1" ds="end batch"/>
              </plan>
            </goal>
            <goal id="ws" ds="write_production_statistics">
              <plan operator="parallel">
                 <goal id="g8" min="1" ds="write production statistics"/>
                 <goal id="g9" min="1" ds="write products errors list"/>
              </plan>
            </goal>
          </plan>
        </goal>
        <mission id="leadUp" min="1">
               <goal id="g1"/>
               <goal id="g2"/>
        </mission>
        <mission id="setup" min="1">
               <goal id="g3"/>
```

```
                <goal id="g4"/>
        </mission>
        <mission id="assembly" min="1">
                <goal id="g5"/>
                <goal id="g6"/>
                <goal id="g7"/>
        </mission>
        <mission id="statistics" min="1">
                <goal id="g8"/>
                <goal id="g9"/>
        </mission>
    </scheme>
</functional-specification>
<normative-specification>
    <norm id="n1" type="obligation" role="rPlanning" mission="leadUp"/>
    <norm id="n2" type="obligation" role="rSetup" mission="setup"/>
    <norm id="n3" type="obligation" role="rAssembly" mission="assembly"/>
    <norm id="n4" type="obligation" role="rStatistic" mission="statistics"/>
</normative-specification>
</organisational-specification>
```

APÊNDICE I – Equipamentos da Simulação

Tabela 84 – Descrição dos equipamentos do protótipo.

Equipamento	Descrição
	*Loader*: este equipamento recebe um lote de placas de circuito impresso sem componentes – placas nuas. As placas são armazenadas no seu *magazine*. Este equipamento é configurado recebendo o tipo de placa a ser produzida e a quantidade.
	*SPI – Solder Paste Insertion*: este equipamento recebe a placa sem componentes e aplica sobre a mesma uma camada de pasta de solda em toda a sua área. Este equipamento é configurado recebendo o tipo de placa a ser produzida para conhecer a área e a espessura de pasta de solda a ser aplicada.
	Insersora: a insersora posiciona os componentes um a um na sua posição (x,y) previamente definidas pelo arquivo de configuração da máquina. O *setup* da máquina é feito com a informação do tipo de produto a ser produzido e o seu respectivo arquivo com lista e posição dos componentes a serem inseridos é carregado pela insersora. A insersora também recebe os componentes a serem posicionados em seus *magazines*.
	Forno de refusão: o forno possui 3 zonas de temperatura que são configuradas para cada tipo de produto. O forno realiza a 'solda' do componente na posição.
	AOI – Automated Object Inspection: este equipamento faz a verificação da qualidade da placa após o forno. Busca por falhas de posicionamento, rotação, inversão, troca, etc. Ele é configurado recebendo o nome da placa a ser produzida.
	*Unloader*: este equipamento recebe as placas finalizadas e as armazena em um *magazine*. Sinaliza quando todas as placas foram recebidas então a fabricação do lote é encerrada.