ARQUITETURA DE SOFTWARE DE CONTROLE ORIENTADA A REGRAS E AGENTES PARA SISTEMAS AUTOMATIZADOS DE MANUFATURA

JEAN M. SIMÃO, PAULO R. O. DA SILVA, PAULO C. STADZISZ, LUIZ A. KÜNZLE

CEFET-PR, CPGEI, LSIP
Av. Sete de Setembro, 3165, 80230-901 Curitiba, PR, BRASIL
Tel.: 0.XX.41.310-4702 Fax.: 0.XX.41.310-4683
E-mails: {simao, pros, stadzisz, kunzle } @cpgei.cefetpr.br

Resumo— Neste artigo propõe-se uma arquitetura de *software* genérica, organizada na forma de um sistema baseado em regras (SBR), para projeto e implementação de Sistemas de Controle em Manufatura Automatizada. A implementação do SBR é feita por meio de agentes que englobam uma base de fatos e os processos de decisão e atuação a serem realizados com os conhecimentos expressos nas regras. O emprego de agentes possibilita um processo de inferência eficiente denominado RETE. Dada a complexidade, os riscos, custos e tempo envolvidos, a concepção do controle de Sistemas Automatizados de Manufatura requer ferramentas de apoio para análise e experimentação da solução proposta. Os ensaios de aplicação da arquitetura de controle proposta são realizados sobre a ferramenta de simulação ANALYTICE II. Esta ferramenta é um simulador a eventos discretos que oferece alta versatilidade de modelagem e separação explícita das partes controle e operativa. A principal contribuição do trabalho é a definição da uma arquitetura de controle flexível e abrangente.

Abstract— This paper presents generic software architecture, organized as a Rule Based System (RBS) for the design and implementation of the Control of Automated Manufacturing System. The RBS is implemented by means of agents than include the facts base and the decision processes. The use of agents allows an effective inference process referred as RETE. Taking into account the complexity, risks, costs and time required, the design of the Control System requires support tools for the analysis and test of the proposed solutions. Tests with the proposed architecture were done using ANALYTICE simulation tool. This is a discrete event simulator that offers high modeling flexibility and that explicitly separate the control and the production parts of a manufacturing system. The main contribution of this work is the definition of a flexible control architecture.

Keywords— Control, Discrete Event System, Automated Manufacturing System Agent, Rule Based System.

1 Introdução

O controle é um dos componentes de um Sistema Automatizado de Manufatura (SAM). Seu papel é a coordenação dos demais elementos constituintes da fábrica (e.g. tornos e robôs) de forma a que realizem processos de produção pré-estabelecidos. A coordenação da fábrica é realizada através : (i) da monitoração dos estados discretos de cada elemento da fábrica; (ii) da tomada de decisão com relação a estes estados e aos processos correntes; e (iii) do comando sobre elementos da fábrica, conforme a decisão tomada, através de ordens apropriadas e segundo protocolos específicos (Künzle, 1990) (Bongaerts, 1998).

Os sistemas de controle de SAM podem ser considerados como Sistemas a Eventos Discretos (SED). Esta categoria de sistema tem sido extensivamente estudada em centros de pesquisa e desenvolvimento. A literatura especializada apresenta várias abordagens e métodos para síntese e implementação de controle de SED (Chaar et al., 1993) (Miyagi, 1996) (Cury et al. 2000) (Langer et al., 2000).

Existem algumas características que fazem do controle de SAM um sistema computacional complexo, entre elas: o grau de automatização, a flexibilidade requerida e a extensão que pode atingir. Questões como riscos, custos e tempo necessário à concepção (inerentes a sistemas complexos) impõem, ao controle, a aplicação de métodos de síntese rigorosos e o emprego de ferramentas de suporte apropriadas para verificação e teste de funcionalidades (Bongaerts, 1998).

Neste artigo apresenta-se uma arquitetura para sistemas de controle de SAM permitindo projetos e implementações robustas e eficientes. A arquitetura apresentada é flexível permitindo implementações para diferentes plantas industriais e o desenvolvimento de diversos modelos de controle (e.g. hierárquico, heterárquico e holônico).

A arquitetura proposta se fundamenta no uso conjunto de duas técnicas : Sistema Baseado em Regras (SBR) (Rich et al., 1991) e Agentes (Franklin et al., 1996) (Ávila et al., 1998). O controle toma a forma de um SBR que é computacionalmente realizado por um conjunto de agentes. Neste SBR, a base de fatos consiste da representação dos estados (i.e. fatos) dos elementos que compõem o sistema de manufatura e as regras exprimem as relações causais que permitem a tomada de decisões em reação a mudanças na base de fatos.

A arquitetura tem sido utilizada em experimentos com o simulador ANALYTICE II (Simão, 2001), desenvolvido pelo grupo do Laboratório de Sistemas Inteligentes de Produção (LSIP) do CPGEI/CEFET-PR.

Esse artigo está organizado da seguinte forma: a seção 2 descreve sucintamente a arquitetura de controle; a seção 3 trata da monitoração e comando dos elementos do SAM; as seções 4 e 5 discutem, respectivamente, as questões da decisão (por regras e agentes) e o processo de inferência; a seção 6 trata do

conflito entre regras. Por fim, a última seção apresenta as conclusões do trabalho.

2 Visão geral da Arquitetura de Controle

O controle de um SAM é dinâmico e orientado a eventos, no qual as variáveis controladas são manipuladas como estados discretos (Künzle, 1990) (Mendes, 1995) (Miyagi, 1996). O controle se comunica com os demais elementos de decisão (Planejamento, Escalonamento e Supervisão) com os quais forma um conjunto integrado para gestão do processo produtivo.

Neste artigo, propõe-se uma arquitetura genérica de controle baseada em agentes. Um agente pode ser definido como um módulo de *software* com alto grau de coesão, com escopo bem definido, com autonomia e pertencendo a um certo contexto, onde seu comportamento atual pode e provavelmente influenciará sua existência futura (Franklin et al., 1996) (Ávila et al., 1998). A arquitetura computacional proposta é orientada a objetos e o conceito de agente é aplicado de forma limitada. De fato, pretende-se identificar entidades mais abstratas dentro da arquitetura através deste conceito de agente. A arquitetura, entretanto, não foi concebida na forma de um sistema multiagente (Yugeng et al., 1999).

Neste trabalho, a monitoração e a atuação sobre os elementos constituintes da fábrica são realizados por meio de agentes especializados. Os elementos do SAM podem ser componentes físicos (e.g. máquinas e peças), componentes de hierarquia (e.g. estação, célula e planta) e elementos informacionais (e.g. lote de peças, plano de produção e plano de processo).

O processo de decisão envolvido também é realizado por um conjunto de agentes. Estes agentes de decisão se orientam através dos estados discretos obtidos na monitoração, implementando o comportamento de regras condição/ação.

Os estados discretos (e.g. robô parado, robô em giro à esquerda, robô abrindo garra) observados na monitoração podem ser compreendidos como fatos. A decisão assume a forma de um conjunto de regras que se apoiam nos fatos. Desta forma, as "regras" interagem com a "base de fatos" avaliando estados e alterando seus valores.

O Controle proposto é, em essência, um SBR com um compromisso entre generalidade e aplicabilidade e cuja execução advém das atividades de agentes. Esta abordagem permite uma inferência avançada, com encadeamento para frente e realizada por associações entre os "agentes de monitoração" e "agentes de regra" seguindo o princípio de notificações introduzido pelo algoritmo RETE (Forgy, 1982) (Rich et al. 1991).

3 Monitoração e Comando dos Elementos do SAM

Todos os elementos do SAM são monitorados para gerar a base de fatos. Porém, devido à pluralidade de naturezas (física, informacional e de hierarquia), faz-se necessário adotar políticas específicas para cada classe.

3.1 Monitoração e Comando de Elementos Físicos

Cada elemento físico constituinte do SAM (e.g. robô, torno, PLC) possuirá um agente do tipo **Comando Ativo de Componente (CAC)** para representá-lo, monitorá-lo e comandá-lo.

Cada **CAC** age como um supervisor especializado (i.e. uma camada de *software* específica) sobre o componente físico. Fazem parte da atividade de monitoração sobre o componente físico : (i) receber sinais de *feedback* (e.g. sinais de sensores, respostas a comandos), (ii) inferir estados não explícitos (e.g. *watchdog*), (iii) mapear estas informações para uma máquina de estados e (iv) notificar a mudança de estados à elementos de decisão apropriados (i.e. "agentes de regra"). Já a atividade de comando envolve: (i) armazenar e inferir informações (e.g. qual ferramenta usar numa operação) e (ii) enviar comandos ao elemento representado, permitindo-o atuar.

Para melhor representar os elementos físicos, pode-se ter uma hierarquia de classes de CACs (e.g. CACs para equipamentos de armazenagem, de transporte ou de atuação) através da qual se especifica características um pouco mais especializada, porém ainda genéricas e aplicáveis à cada nível.

3.2 Monitoração e Comando de Níveis Hierárquicos

Para modularizar e melhor controlar o SAM, podem ser criados níveis hierárquicos. Eles agregam equipamentos formando estações de trabalho ou ainda agregam outros níveis hierárquicos formando células de produção.

Para representar, decidir, monitorar e comandar elementos de hierarquia, existe um tipo de agente denominado **Unidades de Controle (UC)**. A instância de uma **UC** agrega um conjunto de CACs ou outras UCs. Cada UC ainda pode agregar agentes responsáveis por elementos informacionais e por regras. As regras são as responsáveis pela decisão dentro da UC.

As UCs mantém, cada qual, uma máquina de estados com base nos valores de seus atributos (obtidos, a exemplo, por sensores ou *watchdog*) e de seus agregados (obtidos por comunicação). A cada mudança de estado, a UC notifica agentes de decisão relacionados com este evento.

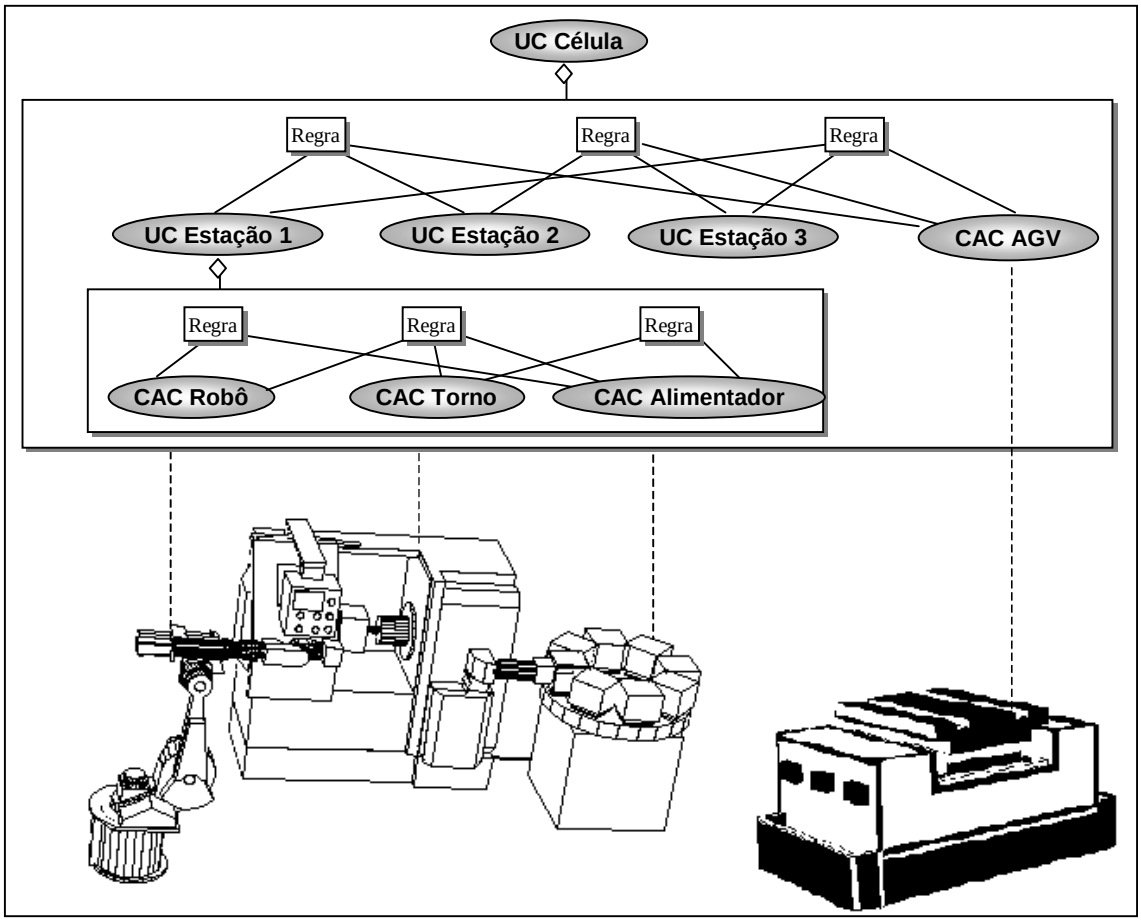

Figure 1 - Fábrica e o controle.

Uma UC pode receber comandos de regras de decisão de outras UCs, intitulados comandos compostos. Um comando composto é um comando de alto nível que induz a realização de um ou mais procedimentos no domínio da UC receptora. Os comandos compostos originam comandos atômicos que são distribuídos aos CACs ou comandos compostos que são enviados a outras UCs, conforme um plano de ações que é parte integrante da UC.

3.3 Monitoração e Comando de Elementos Informacionais

O processo de monitoração não visa somente os elementos físicos e os níveis de hierarquia. Ainda existem os elementos informacionais que surgem do relacionamento do controle com outros elementos de decisão. A exemplo: o "plano de processo" oriundo do *planejamento* e o "plano de produção" oriundo do *escalonamento*.

Na arquitetura proposta, os elementos informacionais são encapsulados em entidades chamadas **Agentes Abstratos (AB)**. Cada **AB** permite representar, manter e alterar os estados de um elemento informacional. A habilidade de alterar estados pode ser vista com um "comando" dado a uma entidade abstrata. Os ABs também possuem a capacidade de notificar agentes interessados, em decorrência de mudança de algum estado, isto é, de algum fato.

3.4 Padrão de Monitoração e Comando

Cada **CAC**, **UC** e **AB** é, de uma forma genérica, um **Agente da Base de Fatos (ABF)**. Os **ABFs** são gerados a partir de uma classe base que encapsula características comuns às suas especializações (i.e. CACs, UCs e ABs). Um exemplo de característica comum é a capacidade de notificação, após mudança de estado, ao conjunto de regras interessadas.

Todo **ABF** agrega dois conjuntos de agentes do tipo **Agente Atributo (AT)** e **Agente Método** (**AM**). Um AT é responsável por representar e manter os estados discretos de um atributo de um ABF (e.g. saber se uma determinada posição de um armazém está com peça ou não). Um AM é responsável por instigar um ABF a realizar alguma ação (e.g. ativar o atuador de um robô). Em última análise, os ATs e AMs são interfaces de saída e entrada de informações de cada ABF. Os ABFs são uma forma de expressão comum (i.e. um padrão de monitoração e comando) dos elementos que possam influenciar a decisão.

A Fig. 1 ilustra sinteticamente a composição de um controle de uma célula de manufatura hipotética.

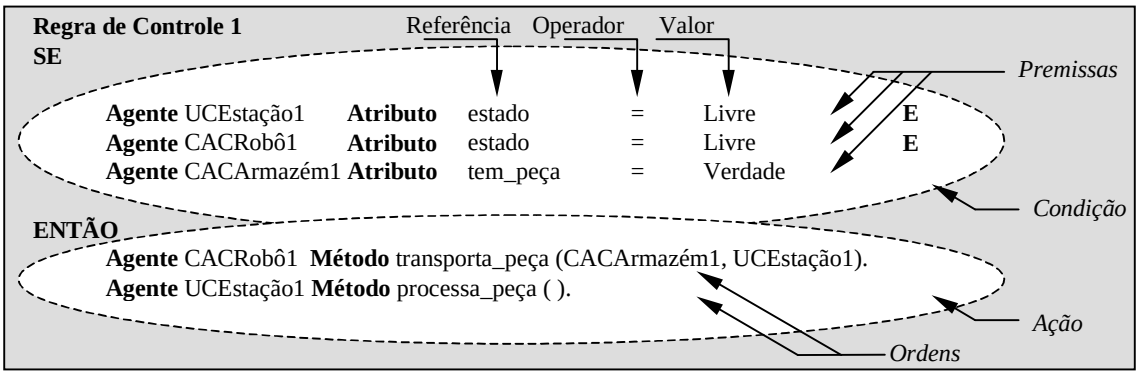

Figure 2 . Uma Regra de Controle e seus constituintes.

4 Decisão por Agentes e regras

Na arquitetura proposta, a decisão é realizada por um conjunto de regras que expressam, por meio de relações causais, como se dá a transição de estados no sistema.

A Fig. 2 demonstra uma Regra de Controle (RC), permitindo compreender a facilidade de se expressar o controle sob a forma de regras. As RCs são a forma de expressar o conhecimento necessário a decisão e ação. Cada RC é composta por uma *Condição* e uma *Ação*. Na condição encontra-se uma série de *Premissas* e na ação uma série de *Ordens*.

Sob o ponto de vista de representação computacional, cada RC é vista como um **Agente Regra (AR)**. Cada AR é composto por dois agentes, o **Agente Condição (AC)** e o **Agente Ação (AA)** que têm uma relação causal e representam, respectivamente, a *Condição* e a *Ação*.

AC – faz o cálculo lógico para o AR no qual está contido. O AC é conectado a um ou mais **Agentes Premissa (AP)** que são as representações das *Premissas*. Cada **AP** possui: (i) um valor booleano sobre si mesmo, (ii) um apontamento à um único atributo (AT) de um ABF (chamado ***Referência***), (iii) um operador lógico (chamado ***Operador***) e (iii) um valor ou um limite de valores (chamado ***Valor***). Cada AP faz comparações lógicas entre o *Valor* e a *Referência* utilizando o *Operador*, o que resulta no seu valor booleano. Outro recurso do AP é comparar o valor da sua *Referência* com uma variável vazia, implicando em uma atribuição. Desta forma, um outro AP, neste mesmo AC, poderá usar esta variável atribuída como seu *Valor*, criando assim uma conexão ou correlação entre os APs. O AC faz o cálculo lógico através da conjunção dos valores booleanos dos APs a ele conectados.

AA – é uma seqüência de **Agentes Ordem (AOs)** que, além de representar as *Ordens*, comandam assincronamente os ABFs citados no respectivo AC, através de seus AMs, de forma a realizarem operações. Cada AA é vinculado a um AC e só poderá ser passível de execução se a avaliação produzida pelo AC correspondente resultar verdadeira.

São características gerais de cada AR: (i) ter seu conhecimento expresso em uma RC; (ii) ser destruído quando um dos agentes referenciados no seu AC deixar de existir; (iv) ser passível de execução quando seu AC for provado; (v) possuir modularidade de escopo, ou seja, os ARs de uma UC não enxergam os ARs de outra UC; (vi) ter APs, conectados ao seu AC, compartilhados com ACs de outros ARs; e (vii) ter AOs, conectados ao seu AA, compartilhados com AAs de outros ARs. O compartilhamento de APs e AOS (i.e. de conhecimento) colabora no processo de inferência.

5 Processo de inferência

Um modelo usual de inferência em SBR é ter a base de fatos como uma lista que é pesquisada cada vez que uma premissa deve ser avaliada (Rich et al., 1991). O algoritmo RETE, que inspira a inferência nesta arquitetura, permite evitar a pesquisa em extensão tomando como base a notificação às regras somente quando um fato é modificado (Forgy, 1982).

A Fig. 3 ilustra o esquema de notificações. Uma notificação advém da ocorrência da mudança de estado em um determinado componentes da fábrica.

A inferência começa através de conexões entre APs e ATs dos ABFs. O ABF avisa ao AT que ele deve mudar de valor (e.g. n1 na Fig. 3). Quando o AT efetivamente muda de valor, ele notifica os APs conectados (e.g. n2 na Fig. 3) que recalculam seus valores lógicos. Se uma mudança de estado então ocorrer em um ou mais APs, elas são expandidas aos ARs (e.g. n3 e n4 na Fig. 3) através do grafo de conexões entre APs e ACs. Então cada AC reavalia seu próprio valor lógico. Os ARs em estado de "verdade" tornam seus AAs plausíveis de execução. Este grafo de conexões, criado pelo inter-relacionamento dos agentes, permite uma inferência rápida. Nesta solução não há buscas em listas e sim a propagação de mensagens desde os ATs, passando pelos APs e ARs, chegando aos AAs quando pertinente.

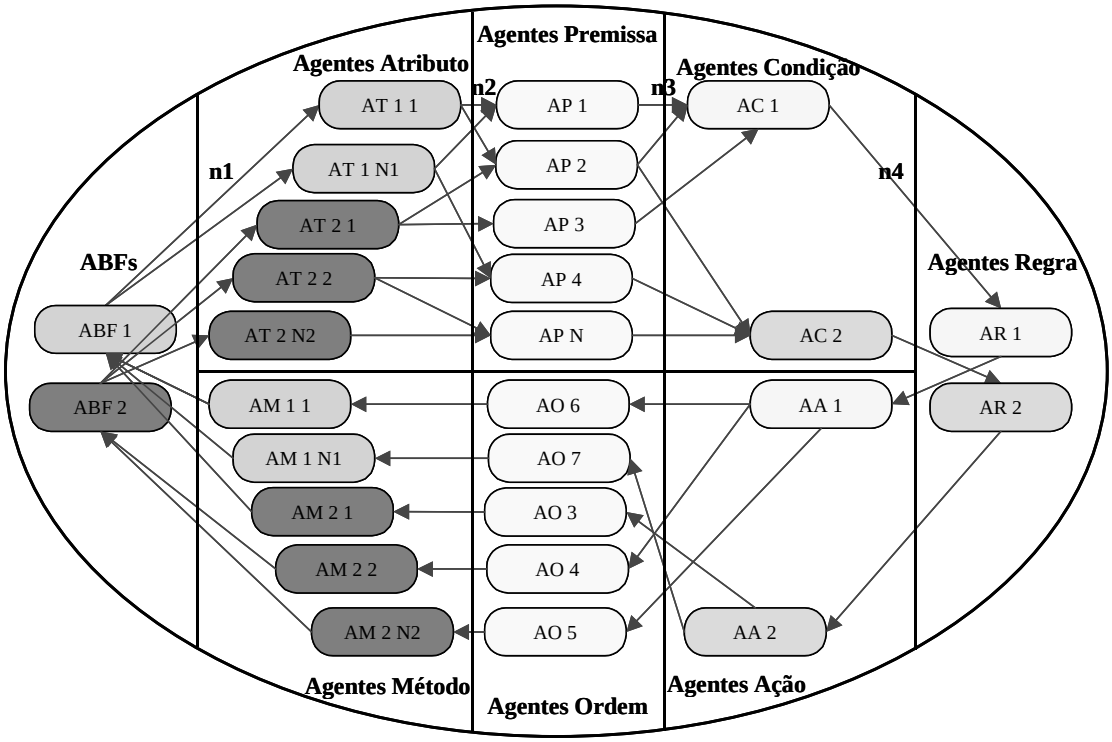

Figure 3. Mecanismo de notificação entre os Agentes.

6 Conflitos entre Agentes Regras

Para compreender o conceito de conflito, deve-se antes compreender o conceito de ABF exclusivo. Este conceito diz respeito a ABFs que representam recursos compartilhados, como por exemplo um robô que atende a duas estações de trabalho.

Um conflito ocorre quando um AP tem a *Referência* oriunda de um ABF exclusivo e está sendo compartilhado entre os ARs de valores booleanos verdadeiros (ARs elegíveis). Por exemplo: a *Premissa* "**Agente** CACRobô **Atributo** estado = Livre" pode colaborar para tornar duas RCs verdadeiras. Porém, se o CACRobô for exclusivo somente uma das RCs poderá ser executada ocorrendo, portanto, um conflito.

Para resolver o conflito, escolhe-se somente um AR para ser ativado (o AR eleito). Isto é feito com base em parâmetros de decisão que podem ser de várias origens. (e.g. um escalonador dinâmico ou políticas de controle). Após a escolha, os outros ARs envolvidos e até então elegíveis passam a ser considerados falsos.

7 Conclusão

Este artigo apresenta uma arquitetura de controle baseada em regras e agentes. O controle apresentado abstrai como primitivas os elementos de um SBR genérico. As primitivas são consideradas agentes que realizam uma inferência rápida e precisa, segundo o princípio de notificação de RETE. Existem, genericamente, dois grupos principais de agentes: (i) os responsáveis pelos processos de monitoração e atuação do controle, vistos como base de fatos; e (ii) os responsáveis pela decisão e comando, vistos como regras.

A arquitetura proposta traz uma abordagem promissora para uma importante área de engenharia, o controle discreto. Aplica-se uma lógica causal, sob o formalismo de um SBR genérico e realizado por agentes, permitindo inúmeras aplicações em instâncias de controle, tanto para simulações como para situações reais.

Os ensaios realizados sobre a ferramenta de simulação ANALYTICE II (Rosinha et al., 2000) demonstram a facilidade de criação do controle seguindo a arquitetura proposta e a robustez dos sistemas constituídos. A integração do controle ao ANALYTICE-II se dá através dos CACs que interagem, não com componentes reais, mas com componentes simulados. A arquitetura, em linhas gerais, abstrai-se da natureza simulada ou real do equipamento, o que sinaliza a independência funcional entre ela e os sistemas automatizados de manufatura.

Os trabalhos futuros incluem a expansão da arquitetura para abranger outros conceitos, como processamento distribuído e ferramentas para auxílio à elaboração da base de fatos e de regras. Outros trabalhos ainda incluem o desenvolvimento de métodos de síntese do controle usando Redes de Petri e técnicas de mapeamento destas soluções para a arquitetura proposta.

Referências Bibliográficas

Ávila, B. C., Abe, J. M. e Prado, J. P. de A. (1998). Inteligência Artificial Distribuída, Aspectos, Série Lógica e Teoria da Ciência, Instituto de Estudos Avançados USP.

Bongaerts, L. (1998). Integration of Scheduling and Control, In Holonic Manufacturing Systems. Ph. D. Thesis PMA / Katholieke Universiteit Leuven.

Chaar, J. K., Teichroew, D. and Volz, R. A. (1993). Developing Manufacturing Control Software: A Survey and Critique, The International Journal of Flexible Manufacturing Systems, Kluwer Academic Publishers. Manufactured in the Netherlands, pp. 053-088.

Cury, José E. R. e de Queiroz, M. H. (2000). Controle Modular de Sistemas de Manufatura Discretos, Anais do XIII Congresso Brasileiro de Automática, Florianópolis, SC, Brasil.

Forgy, C. L. (1982). RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, Artificial Intelligence.

Franklin, S. and Graesser, A. (1996). Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents, Proceedings of the 3th International Workshop on Agent Theories, Architectures and Languages, Springer-Verlag.

Künzle, L. A. (1990) Controle de Sistemas Flexíveis de Manufatura - Especificação dos níveis equipamento e estação de trabalho, Dissertação de Mestrado, CEFET/PR..

Langer, G., Sorensen, C., Schnell, J. and Alting, L. (2000). Design of a Holonic Shop Floor Control System for a Steel Plate Milling-Cell, In : 2000 International CIRP Design Seminar on Design with Manufacturing: Intelligent Design Concepts Methods and Algorithms, Haifa, Israel.

Mendes, R. S. (1995). Modelagem e Controle de Sistemas a Eventos Discretos - Manufatura integrada por computador, Belo Horizonte, Fundação CEFET-MG.

Miyagi, P. E. (1996). Controle Programável – Fundamentos do Controle de Sistemas a Eventos Discretos, Edgard Blücher, 1996.

Rich, E. and Knight, K. (1991) Artificial Intelligence, McGraw-Hill.

Rosinha, L. F., Koscianski, A., Stadzisz, P. C. and Künzle, L. A. (2000). Arquitetura Aplicada ao Projeto de FMS. XII Congresso Brasileiro de Automática, p.1012-1017, Florianópolis.

Simão, J. M. (2000). Proposta de uma Arquitetura para Sistemas Flexíveis de Manufatura Baseada em Regras e Agentes. Dissertação de mestrado. CPGEI/CEFET-PR.

Yufeng, L. and Shuzhen, Y. (1999). Research on the Multi-Agent Model of Autonomous Distributed Control System, In 31 International Conference Technology of Object-Oriented Language and Systems, IEEE Press, China.