ATIVIDADES DE CONTROLE E SUPERVISÃO ASSISTIDAS POR UM SIMULADOR DE FMS

Jean Marcelo Simão[1], Paulo Roberto O. da Silva[2], Paulo César Stadzisz[3], Luiz Allan Künzle[4]
Centro Federal de Educação Tecnológica do Paraná
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – Curitiba, PR
(1) simao@cpgei.cefetpr.br, (2) pros@ieee.org, (3) stadzisz@cpgei.cefetpr.br,
(4) kunzle@cpgei.cefetpr.br

Resumo. *O projeto e a implementação do Controle e Supervisão de Sistemas Flexíveis de Manufatura (FMS) são atividades complexas devido ao grau de automatização e flexibilidade requerida. Em razão dos riscos, custos e tempo envolvidos, estas atividades exigem ferramentas de apoio para análise e experimentação de soluções. Este artigo apresenta uma arquitetura de Controle e Supervisão de FMS integrante da ferramenta de simulação ANALYTICE-II. Esta ferramenta incorpora um simulador a eventos discretos que tem como primitivas equipamentos (e.g. tornos, fresas, esteiras), peças e paletes. Neste trabalho são consideradas as tarefas de monitoração do estado do sistema, coordenação das operações, detecção e diagnóstico de falhas e definição de estratégias de recuperação. A principal contribuição do trabalho é a integração da arquitetura de Controle e Supervisão ao simulador. Destacam-se a abordagem do Controle baseado em Regras e Agentes, o emprego da Rede de Petri Temporal Orientada a Objetos na modelagem da planta e a cooperação entre o raciocínio indutivo empírico e o rigor sistemático do raciocínio dedutivo nos diagnósticos formais.*

Palavras-chave: *FMS, controle, supervisão, simulação, diagnóstico.*

1. INTRODUÇÃO

Desenvolve-se no CEFET-PR, desde 1990, trabalhos que visam a capacitação em Sistemas Flexíveis de Manufatura (FMS), publicados em vários artigos e dissertações de mestrado. Atualmente desenvolve-se um novo simulador de FMS denominado ANALYTICE-II.

Este artigo trata das atividades de Controle e Supervisão em FMS e de sua integração ao ANALYTICE-II. Essas atividades são complexas e requerem ferramentas de apoio à concepção e à análise. A partir do estudo de diversas abordagens apresentadas na literatura, propõe-se uma arquitetura para integrar as diferentes visões, algoritmos e estruturas de dados envolvidos.

A arquitetura para o Controle de FMS se fundamenta em um Sistema Baseado em Regras (SbR) [Rich, 1991]. A base de fatos se constitui de agentes que representam abstrações de componentes do FMS (e.g. máquinas e peças) ou abstrações de hierarquia (estação, célula e

planta). As regras têm condições baseadas nos atributos dos agentes e ações que os modificam. A inferência é feita por encadeamento para frente, com base no algoritmo RETE [Rich, 1991].

A arquitetura de Supervisão trata da monitoração, do diagnóstico e das estratégias de recuperação. Considera que não é possível capturar em um único tipo de modelo uma compreensão plena de um sistema complexo, estando, portanto, baseada em múltiplos modelos formais dos componentes. Considera, também, que o comportamento normal dos componentes muda ao longo do tempo, ao contrário da maioria dos modelos formais. Conseqüentemente, a arquitetura permite a cooperação das abordagens baseadas em modelos empíricos, buscando um bom compromisso entre completude e complexidade.

Esse artigo está organizado da seguinte forma: a seção 2 apresenta o simulador ANALYTICE-II, as seções 3 e 4 descrevem a arquitetura de Controle e a modelagem e validação de estratégias de Controle. As seções 5 e 6 descrevem a problemática, a arquitetura e a modelagem dos componentes em Supervisão. A seção 8 apresenta as conclusões do trabalho.

2. ANALYTICE-II

ANALYTICE-II é uma ferramenta de simulação a eventos discretos que permite realizar experimentos para determinar parâmetros qualitativos e quantitativos através de seus principais módulos: modelo de equipamentos, ambiente físico, animação geométrica, núcleo e monitoração de simulação [Rosinha, 2000], [Koscianski, 2000].

As primitivas de modelagem são peças, paletes e equipamentos. Um equipamento é representado pelos modelos geométrico, cinemático e comportamental. O modelo geométrico pode ser importado de sistemas CAD. O modelo cinemático associa eixos de translação e rotação ao modelo geométrico, possibilitando sua animação tridimensional. O modelo comportamental representa a evolução dos estados do equipamento e apresenta uma interface para receber sinais de comando, como "abrir garra", e enviar sinais de resposta, como "fim de usinagem". Os modelos comportamentais são descritos por classes em linguagem de programação C++ cujas instâncias (objetos) são executadas e interagem com o núcleo do simulador durante a simulação.

3. ARQUITETURA DE CONTROLE PROPOSTA

O Controle de um FMS é dinâmico e orientado a eventos, onde as variáveis controladas são manipuladas como estados discretos [Künzle, 1990]. O Controle se comunica com os demais elementos de decisão (Planejamento, Escalonamento e Supervisão) com os quais forma um conjunto integrado. A arquitetura de controle proposta é estruturada por agentes representando a base de fatos e regras. Estes agentes realizam as inferências.

3.1 Base de fatos

Cada agente da base de fatos é um **Comando Ativo de Componente (CAC)** ou uma **Unidade de Controle (UC)** que concorre ou coopera com outros agentes na realização de tarefas.

CAC inclui habilidades de decisão e comando de um componente do FMS. São responsabilidades do CAC com relação ao componente controlado: (i) mapear seu comportamento para uma máquina de estados; (ii) estimar seus estados através de informações de monitoração; (iii) enviar comandos e receber respostas; (iv) armazenar e inferir informações (e.g. qual ferramenta usar numa operação). A integração do Controle ao

ANALYTICE-II se dá através dos CACs que interagem, não com componentes reais, mas com componentes simulados.

UC adiciona a habilidade de decisão e comando a níveis hierárquicos do FMS (e.g. planta, célula e estação). Uma UC agrega, no primeiro nível, um conjunto de CACs e regras e, em níveis superiores, um conjunto de CACs, UCs e regras. Cada UC mantém uma máquina de estados com base nos valores dos atributos seus e de seus agregados.

3.2 Agentes regra

Um Agente Regra (AR) é uma representação computacional, na forma de agente, de uma das regras que controlam o FMS. Cada AR é composto por dois agentes, o Agente Condição (AC) e o Agente Ação (AA) que possuem uma relação causal.

AC – faz o cálculo lógico para o AR que o possui. O AC é conectado a um ou mais Agentes Premissa (AP). Cada AP possui: (i) um valor booleano sobre si mesmo, (ii) um apontamento à um único atributo de um CAC ou de uma UC (chamado *Referência*) e (iii) um valor ou um limite de valores (chamado *Valor*). Cada AP faz comparações lógicas entre o *Valor* e a *Referência*, 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; então um outro AP subseqüente, neste mesmo AC, poderá usar esta variável atribuída como seu *Valor*, criando assim uma conexão entre os APs. O AC faz o cálculo lógico através da conjunção dos valores booleanos dos APs aos quais está conectado.

AA – é uma seqüência de Agentes de Ordem (AO) que comandam de forma assíncrona as UCs ou os CACs citados na respectiva AC, 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 verdadeiro.

São características gerais do AR: (i) ser criado por Regras de Formação (RF) que explicitam como serão os ARs; (ii) ser destruído quando um dos agentes referenciados no seu AC deixar de existir; (iii) o resultado verdadeiro da avaliação gerada pelo AC não é o suficiente para ter seu AA ativado, pois dois ou mais AR podem estar compartilhando uma premissa exclusiva, ou seja, um conflito; (iv) possuir modularidade de escopo, ou seja, os ARs de uma UC não enxergam os ARs de outra UC, o que facilita o Controle pois reduz a comunicação entre agentes.

3.3 Agentes de formação

Agentes de Formação (AF) são representações computacionais de RFs que criam os ARs. Um AF é mais genérico que um AR, pois suas premissas só analisam se um CAC ou uma UC é de uma determinada classe (e.g. CAC Torno ou UC Estação) sem considerar, a princípio, os valores dos seus atributos. Uma regra que especifica se uma classe de robôs pode pegar uma classe de peças é mais abrangente do que outra que especifica se um certo robô pode pegar uma certa peça. Um AR deriva de um AF, restringindo-se a uma combinação específica de agentes e considerando as restrições de seus atributos.

As premissas de um AF podem combinar diversos CACs ou UCs. Cada combinação cria um AR e conseqüentemente seus ACs e AAs. Após a criação do AC, são criados e conectados os APs necessários. Na criação de AP, lhe é conectada a *Referência* e passado o *Valor*, conforme o AF especifica. Já criado, o AP executa seu cálculo lógico e se auto atribui um valor booleano. Assim um AR pode, após ter seus APs conectados, saber seu valor booleano. À medida que mais ARs são criados e precisam de APs já existentes, é apenas feita uma conexão entre APs e ARs, evitando redundâncias e gerando um grafo de conexões. O AA, após criado, é conectado a uma seqüência de AOs, de acordo com o especificado no AF. As informações de ação do AF servem só para criar os AAs dos ARs.

Os AFs só analisarão as restrições aos atributos dos CACs ou UCs se isto for explicitado na premissa. Já os ARs sempre as analisam. Para que um AF analise restrições ou para que os ARs derivados deste AF as utilizem em sua composição estas restrições devem ser explicitamente representadas no AF.

3.4 Processo de inferência

Em SbR, uma inferência usual é ter a base de fatos como uma lista que é pesquisada cada vez que uma premissa deve ser avaliada. A avaliação completa analisa as premissas de todas as regras e é motivada por algum evento. A partir desta idéia criou-se avanços como o algoritmo RETE [inserir referência ao RETE] que serve de inspiração para inferência desta arquitetura.

A inferência inicia com a criação dos ARs e dos APs que, neste momento, fazem o cálculo lógico. A inferência continua através da conexão entre APs e atributos dos CACs ou das UCs. Quando um atributo muda de valor, os APs conectados a ele irão recalcular seu valor. Através do grafo de conexões, entre os APs e os ACs, a mudança de estado é expandida aos ARs que irão reavaliar sua condições. 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 o AP cujo cálculo lógico tenha mudado até os ACs, ARs e AAs logicamente dependentes do originador da mensagem, quando pertinente.

3.5 Conflitos entre agentes regras

Um conflito ocorre quando um AP é exclusivo e está sendo compartilhado entre os ARs de valores booleanos verdadeiros (ARs Elegíveis). Os conflitos dos ARs podem ser de mesma origem (quando são criados pelo mesmo AF) ou de origem distinta (quando são criados por AFs distintos). Para resolver o conflito, escolhe-se um só AR para ser ativado (o AR Eleito). Isto é feito com base em parâmetros de decisão que podem ser de várias origens. (e.g. de um escalonador dinâmico ou de políticas de controle).

4. MODELAGEM E VALIDAÇÃO DO CONTROLE POR REDES DE PETRI

As Redes de Petri (RdP) são ferramentas de especificação formal de sistemas discretos, permitindo analisar a dinâmica, de forma algébrica e gráfica, exprimindo paralelismos, concorrências e sincronizações. Outras vantagens deste formalismo são as propriedades estruturais e do modelo. RdP possuem extensões, como as RdPs à Objetos (RdPO). RdP e suas extensões são muito úteis em modelagem de Controle [Miyagi, 1996].

A arquitetura de Controle proposta usa regras como primitivas de descrição da dinâmica, mas isto não implica que a modelagem seja por regras. Pode-se elaborar um módulo, onde o Controle seja sintetizado numa RdPO e traduzido em conjunto de regras [Valette, 1991] que comporiam uma RdPO onde o jogador seria o motor de inferência [Cardoso, 1997].

A arquitetura permite a rápida formulação de partes do Controle, pois a criação de poucas regras é simples e pode ser feita por alguém que conheça empiricamente regras de Controle do FMS mas não conheça RdP. Um outro módulo na arquitetura poderia traduzir estas regras de Controle em uma RdPO e fazer uma análise formal [Nazareth, 1993], [Harhalakis, 1995].

5. SUPERVISÃO

As dificuldades da Supervisão Industrial são, principalmente, o grau variável de incerteza, as falhas transientes, o crescimento exponencial do número de hipóteses com o número de componentes [Stefik, 1995] e as dificuldades humanas diante desta atividade. Dentre as dificuldades humanas destacam-se: (i) a estruturação mental da complexidade e da

dinâmica dos sistemas; (ii) a necessidade de manter a atenção permanentemente concentrada; (iii) a dificuldade de conscientização da evidência de falhas não usuais, devido à economia cognitiva, (iv) a tendência de aceitação, que favorece a hipótese corrente, mesmo diante de contradições e (v) a opção por medidas de recuperação complexas ou equivocadas diante de estresse [Morrison, 1994].

O papel da Supervisão, na Teoria do Controle Supervisório [Mendes, 1995], é associar uma seqüência de eventos à saída do sistema. Em geral, o número de sensores disponível é limitado, pelo que cria-se modelos abstratos, capazes de monitorar indiretamente todos os estados a partir das observações disponíveis, como os modelos baseados em alarmes, em reconhecimento de padrões e em relações estruturais e comportamentais. Os modelos de monitoração baseados em alarmes de exceção são ineficientes, por falta de sensibilidade ao contexto. Uma das alternativas é projetar seqüências de testes, porém é difícil considerar todas as falhas possíveis. Outra possibilidade é o reconhecimento de padrões nos sinais observados, entretanto, é difícil explicitar o conhecimento dos especialistas. A Supervisão baseada nas especificações formais, permite incluir o diagnóstico no projeto dos sistemas e aprimorar a interação entre os engenheiros responsáveis pela especificação e os que estudam a manutenibilidade dos sistemas de manufatura.

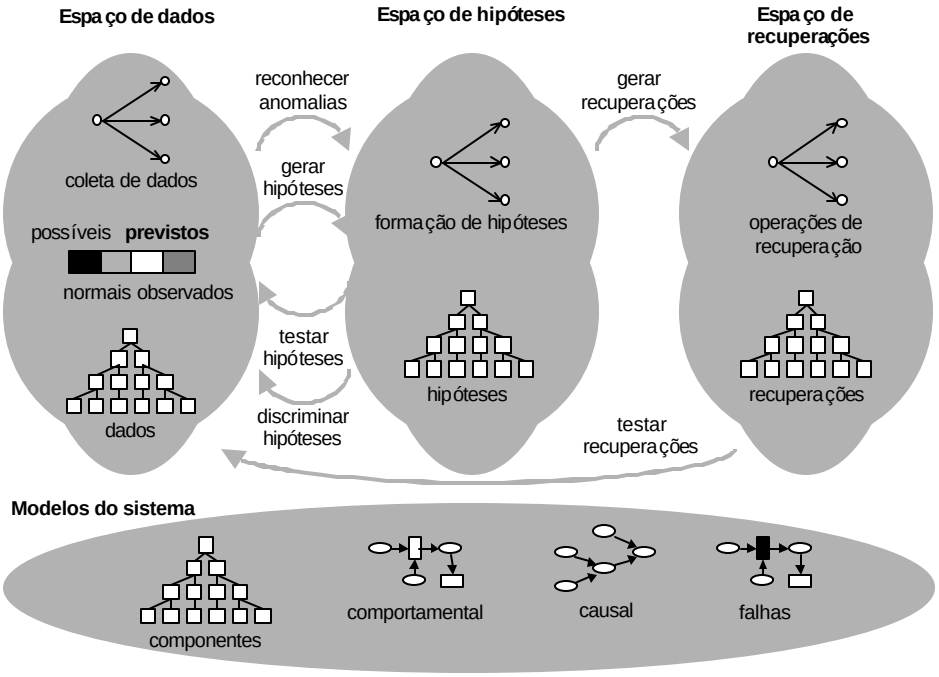

Figura 1 – Tarefas e espaços de estado na Supervisão (adaptado de [Stefik, 1995])

6. ARQUÉTIPO SUPERVISÓRIO GENÉRICO

O modelo proposto para supervisão em ANALYTICE II é baseado no trabalho de M. Stefik que apresenta um arquétipo supervisório genérico (Figura 1) [Stefik, 1995]. Nesse modelo, o **espaço de dados** corresponde a um conjunto finito de pontos de prova. O **espaço de hipóteses** representa hipóteses de diagnóstico. O **espaço de recuperações** representa o conjunto das ações possíveis para reparar o sistema. Os **modelos do sistema** descrevem sua estrutura e comportamento. Os três espaços de dados e os modelos do sistema podem apresentar múltiplos níveis de abstração e compartilhar procedimentos e estruturas de dados comuns.

Com base nestes fatos desenvolvem-se as principais tarefas da supervisão: (i) reconhecimento de anomalias, (ii) geração e teste de hipóteses, (iii) discriminação entre hipóteses e (iv) geração e teste da estratégia de recuperação.

6.1 Reconhecimento de anomalias

Para deduzir se uma observação é normal, pode-se executar uma simulação registrando as previsões não confirmadas como conflitos. Isso nem sempre é viável, pois a maioria das abordagens para a previsão de valores é logicamente incompleta. Recorre-se então às regras anticomportamentais para aumentar o número de previsões possíveis, mesmo que o sistema físico não possa operar ao reverso.

Havendo uma especificação formal do sistema, pode-se deduzir a ocorrência de falhas quando ocorre o rompimento das relações invariantes do modelo. Em sistemas complexos e auto-ajustáveis, porém, é difícil determinar a ocorrência de conflitos, através da propagação de restrições ou de simulação. Nesse caso, aplica-se a tarefa de classificação com base em modelos que não estabelecem o mapeamento da estrutura e do comportamento.

6.2 Geração de hipóteses

Algumas estratégias de diagnóstico consistem em executar seqüências sistemáticas de testes. A atual disponibilidade de recursos privilegia as técnicas de inferência ao invés da execução de testes que demandam custos e tempo maior. A inferência permite determinar o conjunto de componentes plausivelmente em estado de erro, antes de proceder qualquer medição.

As abordagens mais simples de diagnóstico se baseiam em tipos de faltas em componentes e assumem que o sistema tem uma única falta, ou que as faltas podem ser determinadas uma a uma, seqüencialmente. As estratégias mais robustas expandem sistematicamente os diagnósticos plausíveis, com base nas interações entre componentes e consideram combinações de faltas em grupo. A explosão combinatória dos diagnósticos compostos e a eficiência computacional requerem técnicas de busca hierárquica.

6.3 Discriminação entre hipóteses

Para discriminar hipóteses, a aquisição de dados inicia pela seleção do ponto de prova e termina quando as outras hipóteses plausíveis são descartadas. Uma das técnicas mais utilizadas é o “ponto de prova guiado” que começa onde a discrepância foi constatada, selecionando os componentes antecessores que apresentam saídas discrepantes. Se um antecessor recebe entradas válidas mas produz um resultado errado, ele deve estar em falha. Essa abordagem freqüentemente usa mais medições do que a seleção criteriosa do ponto a medir. Um dos critérios para seleção é a avaliação da entropia de Shannon que relaciona o ponto de prova com o ganho de informação [Stefik, 1995]. É um critério eficiente, porém, assume que o custo e o risco associados a cada ponto são uniformes, não considera a incerteza e atribui a mesma relevância a todas as hipóteses.

Quando o descarte de uma hipótese requer interação com o ser humano, a aplicação estrita de critérios de seleção pode gerar um diálogo aparentemente desconexo, sendo necessário ordenar a interrogação de forma aceitável para o ser humano.

6.4 Estratégias de recuperação

Diante de um diagnóstico, a estratégia de recuperação requer um plano de contingência para o modo de operação degradado, o que abrange tarefas como planejamento de rota, escalonamento e comando de máquinas, com base nos planos de processo em execução. Em

ANALYTICE II estas estratégias de recuperação ocorrerão por um diálogo entre supervisor e controlador.

6.5 Modelo dos componentes

Para dar suporte ao diagnóstico formal sistemático, o modelo da planta deve ser capaz tanto de simulação (prever as saídas a partir das entradas) quanto de inferência (inferir as entradas a partir das observações), o que não é possível na representação em C++ atual. Portanto, a arquitetura proposta aplica redes de Petri temporais orientadas a objetos [Esser, 1996] na modelagem comportamental. Ademais, o modelo comportamental é estendido para incluir regras que constituem o conhecimento empírico, como proposto em [Ayeb, 1995].

7. CONCLUSÕES

Neste artigo foram apresentadas as arquiteturas de controle e supervisão, a integrar à ferramenta de simulação ANALYTICE-II.

O Controle apresentado abstrai como primitivas os elementos de um SbR. As primitivas são consideradas agentes que possuem uma comunicação, inspirada em RETE, permitindo uma inferência rápida e precisa. As primitivas também permitem que o Controle seja expresso ou validado por RdP.

A arquitetura de Supervisão considerou as dificuldades humanas perante a atividade, permitindo a modelagem do sistema a partir de múltiplos modelos e a colaboração entre modelos de especificação formal com abordagens empíricas. Propõe-se, como linguagem de especificação comportamental dos componentes, o emprego de Redes de Petri Temporais Orientadas a Objetos.

REFERÊNCIAS

[Ayeb, 1995] Ayeb, B. el, 1995: Toward Systematic Construction of Diagnostic Systems for Large Industrial Plants: Methods, Languages, and Tools. IEEE Transactions on Knowledge and Data Enginering, vol. 6, no. 5, october.

[Cardoso, 1997] Cardoso, J. & Vallete, R., 1997: Redes de Petri. UFSC, Florianópolis, SC.

[Esser, 1996] Esser, R., 1996: An object oriented Petri net approach to embedded system design. Dissertation for the degree of doctor of technical sciences. Swiss Federal Institute of Technology Zurich.

[Harhalakis, 1995] Harhalakis, G. & Lin, C. P. & Mark, L. & Muro-Medrano, P. R., 1995: Structured representation of rule-based specifications in CIM using updated Petri nets. IEEE Transactions on Systems, Man and Cybernetics, Vol. 25, Nr.1.

[He, 1998] He[1], X. & Chu[2], W. C. & Yang[3], H. & Yang[4], S. J. H., 1998 : A new approach to verify rule-based systems using Petri nets. National Central. [1]North Dakota State University, U.S.A., [2]TungHai University, Taiwan, [3]De Montfort University, U.K. & [4]National Central University, Taiwan.

[Künzle, 1990] Künzle, L. A., 1990: Controle de Sistemas Flexíveis de Manufatura – Especificação dos Níveis Equipamento e Estação de Trabalho. Dissertação de Mestrado, CEFET/PR.

[Koscianski, 2000] Koscianki, A., 2000: Projeto e implementação de um simulador com animação gráfica para FMS. Dissertação de mestrado, CEFET/PR.

[Mendes, 1995] Mendes, R. S., 1995: Modelagem e Controle de Sistemas a Eventos Discretos. In Markus, M. e Pires, P. (organizadores): Manufatura integrada por computador. Belo Horizonte, Fundação CEFETMINAS.

[Miyagi, 1996] Miyagi, P. E., 1996: Controle programável – fundamentos do controle de sistemas a eventos discretos. Edgard Blücher.

[Morrison, 1994] Morrison, D. L. & Upton, D. M., 1994: Fault Diagnosis and Computer Integrated Manufacturing Systems. IEEE Transactions on Engineering Management. Jan.

[Nazareth, 1993] Nazareth, D. L., 1993: Investigating the applicability of Petri nets for rule-based system verification. IEEE Transactions on Knowledge and Data Engineering, Vol. 4 – Nr. 3.

[Rich, 1991] Rich, E. & Knight, K., 1991: Artificial Intelligence, McGraw-Hill.

[Rosinha, 2000] Rosinha, L. F. F., 2000: Proposta de uma arquitetura de simulação para sistemas flexíveis de manufatura e de modelagem de equipamentos industriais. Dissertação de mestrado. CEFET/PR, fevereiro.

[Stefik, 1995] Stefik, M., 1995: Introduction to knowledge systems, pp. 674. Morgan Kauffmann.

[Valette, 1991] Valette, R. & Bako, B., 1991: Software implementation of Petri nets and compilation of rule-based systems. Lecture Notes in Computer Science 524, Ed Rozenberg, G., Advances in Petri Nets 1991. Springer-Verlag.

CONTROL AND SUPERVISION ACTIVITIES ASSISTED BY A FMS SIMULATOR

Abstract. *The design and implementation of Control and Supervision of Flexible Manufacturing Systems (FMS) are complex activities, due to the automation degree and flexibility required. Given the risks, costs and time implied, these activities need tools to support solutions' analysis and experimentation. This article presents a Control and Supervision architecture to integrate the simulation suite ANALYTICE-II, a discrete event simulator whose primitives are equipments (lathers, milling machines, conveyors), parts and pallets. The system's states monitoring, operations' coordination, failure detection, failure diagnostic and recovering strategies definitions tasks are considered. The main contribution of this work is the Control and Supervision architecture integration to the simulator. Distinguished are the Rules and Agents Control based approach, the Object Oriented Temporal Petri Nets employment in the plant modeling and the cooperation between empirical inductive reasoning and systematic strictness of the deductive reasoning in formal diagnostic.*

Keywords: *FMS, control, supervision, simulation, diagnostic.*