

REDES DE PETRI E ORIENTAÇÃO A OBJETOS PARA O DESENVOLVIMENTO DE SISTEMAS HÍBRIDOS

Villani, Emilia

Dept. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica, USP
Av. Prof. Mello Moraes, 2201 São Paulo Brasil
emiliav@usp.br

Miyagi, Paulo Eigi

Dept. Eng. Mecatrônica e de Sistemas Mecânicos, Escola Politécnica, USP
Av. Prof. Mello Moraes, 2201 São Paulo Brasil
pemiyagi@usp.br

Valette, Robert

LAAS – CNRS (Laboratoire d'Analyse et d'Architecture des Systèmes)
7, Avenue du Colonel Roche, 31077 Toulouse Cedex 4 FRANCE
robert@laas.fr

Resumo. Este artigo propõe uma nova abordagem para projeto de sistemas de controle híbrido, onde “híbrido” indica a presença simultânea de características de sistemas a eventos discretos (sistemas dinâmicos com estados discretos e transições instantâneas) e de sistemas contínuos (sistemas dinâmicos com variáveis de estado e eventos contínuos). Partindo de uma proposta já existente de Rede de Petri associada a equações diferenciais, considera-se uma solução para modelagem de sistemas industriais complexos através de uma visão estruturada. Assim, esta nova abordagem introduz o conceito de orientação a objetos à utilização de Rede de Petri associada a equações diferenciais. A notação UML é utilizada para auxiliar a representação de diferentes aspectos do sistema. Através de um exemplo, verifica-se que a utilização em conjunto da UML e de redes Predicado Transição Diferenciais se complementam, possibilitando o enfoque das características híbridas do sistema.

Palavras chave: sistemas híbridos, orientação a objetos, redes de Petri, UML.

1. Introdução

Dentro do conceito de sistemas mecatrônicos, um aspecto importante é a integração de sistemas de diferentes tecnologias (Fraser et al., 1993). Esta integração tem sido objeto de diversos estudos e, em muitos casos, é um fator limitante para o aperfeiçoamento de sistemas de controle. Neste contexto, um problema frequentemente abordado, que é um resultado direto desta necessidade de integração, é o desenvolvimento de sistemas híbridos.

O conceito de “sistemas híbridos” se insere dentro do contexto de modelagem de sistemas, onde uma possível classificação refere-se aos tipos de variáveis encontradas no modelo. Neste sentido, são definidos os sistemas a eventos discretos (tempo e variáveis de estado são representados por números inteiros ou variáveis lógicas), os sistemas de variáveis contínuas (tempo e variáveis de estado são representados por números reais) e os sistemas híbridos, onde encontra-se a presença simultânea de características de sistemas a eventos discretos e de sistemas contínuos.

Em particular, no que se refere ao projeto de sistemas supervisórios híbridos, o aumento da demanda por produtividade e eficiência tem resultado no aumento da sofisticação e complexidade dos processos supervisionados. Associada à evolução da tecnologia de automação disponível, os profissionais devem assim lidar com tomadas de decisões cada vez mais complexas. A ausência de uma abordagem efetiva para o tratamento da complexidade conceitual relacionada ao projeto de sistemas supervisórios híbridos é apontada em [Nimmo, 1999] [Garcia, 1997].

Neste sentido, partindo de uma proposta já existente de rede de Petri associada a equações diferenciais para modelagem de sistemas híbridos, este trabalho busca uma solução para sistemas de elevada complexidade através de uma visão mais estruturada. Assim, apresenta-se uma nova abordagem onde introduz-se o conceito de orientação a objetos à utilização de rede de Petri associada a equações diferenciais.

Este trabalho está organizado da seguinte forma. No item 2 apresenta-se uma introdução sobre sistemas híbridos. O item 3 introduz a orientação a objetos e como os conceitos relacionados à orientação a objetos podem ser incorporados à teoria de redes de Petri. O item 4 apresenta os principais aspectos da abordagem proposta para sistemas híbridos. O item 5 apresenta algumas considerações finais e indica as principais direções nas quais este trabalho deve evoluir.

2. Sistemas Híbridos

Enquanto a teoria de controle clássico dispõe de técnicas e ferramentas para análise e síntese de sistemas contínuos, e abordagens equivalentes apresentam soluções similares para sistemas a eventos discretos, não existe uma abordagem formalizada e consolidada para a análise de sistemas com comportamento dinâmico discreto e contínuo simultaneamente, isto é, as soluções são abordadas de modo *ad hoc*, evidenciando a necessidade de desenvolvimento e consolidação de uma abordagem sistemática para o tratamento de sistemas híbridos [Lemmon, He & Markovskiy, 1999].

Um ponto de importância crucial no desenvolvimento de sistemas de controle é a decisão de qual modelo adotar. A escolha da linguagem de modelagem influencia diretamente as possibilidades de análise, verificação e validação e tem uma profunda influência em como um problema é abordado e como sua solução é emoldurada.

No que se refere a modelagem de sistemas híbridos, muitos formalismos têm sido propostos. Para uma revisão bibliográfica detalhada ver [Guéguen & Lefebvre, 2000], [Champagnat, 1998]. De uma forma geral, algumas abordagens consistem em extensões de modelos contínuos, como equações diferenciais ordinárias nas quais são incluídas variáveis cujo valor pode ser modificado de forma descontínua no tempo [Antsaklis & Nerode, 1998]. Outras abordagens consistem na modificação de técnicas de modelagem utilizadas em sistemas a eventos discretos, onde são introduzidos novos elementos para a representação da dinâmica contínua do sistema, como nas Redes de Petri Híbridas [Alla & David, 1998]. Existem também abordagens intermediárias que combinam modelos de sistemas contínuos, descritos por equações diferenciais, e de sistemas discretos, descritos por autômatos finitos ou Redes de Petri, onde é introduzida uma interface para a comunicação entre os dois tipos de modelos, como em [Valentin-Roubinet, 1998].

Entre estas abordagens, este trabalho considera particularmente as abordagens do último grupo, pois a extensão de formalismos discretos leva, em geral, a uma ferramenta com capacidade de modelagem da parte contínua relativamente restrita e específica. O equivalente pode ser afirmado para as extensões de formalismos contínuos. Por outro lado, as abordagens que especificam uma solução a partir da mistura de um formalismo discreto com um formalismo contínuo apresentam maior flexibilidade e poder de modelagem. Entre as abordagens deste grupo, considerou-se em especial aquelas onde o formalismo discreto é baseado em redes de Petri. Esta escolha foi realizada considerando-se as já bem conhecidas propriedades deste formalismo, como habilidade para representar a sincronização de processos, eventos concorrentes, causalidade, compartilhamento de recursos, presença de conflitos, entre outros.

Entre as principais abordagens para sistemas híbridos derivadas de redes de Petri que definem uma interface entre a parte discreta e a parte contínua, tem-se as redes de Petri Mistas [Valentin-Roubinet, 2000] e as redes Predicado Transição Diferencial [Champagnat, 1998]. Estes formalismos, no entanto, não apresentam recursos para a decomposição do sistema e para uma modelagem progressiva, por exemplo, através de uma abordagem hierárquica, onde os modelos podem ser expressos com diferentes níveis de detalhamento. A modelagem é realizada de uma forma plana, o que dificulta o estudo de sistemas complexos, onde não é possível compreender o sistema como um todo na profundidade necessária.

Buscando uma solução para estes problemas, este trabalho introduz os conceitos de orientação a objetos para as redes de Petri associadas a equações diferenciais. Entre as vantagens da orientação a objetos ([Booch, Rumbaugh & Jacobson, 1998], [Douglass, 1998], [Paludetto, 1991]) encontram-se a reutilização, a facilidade de modificação, revisão e manutenção dos modelos, e a maior correspondência entre os objetos do sistema de controle e dispositivos reais relacionados ao problema, pois a estrutura e o comportamento dos dados estão diretamente vinculados.

Entre as duas propostas de redes de Petri associadas a equações diferenciais citadas anteriormente, selecionou-se para os propósitos deste trabalho a rede de Petri Predicado Transição Diferencial pois considera-se que esta rede apresenta características que facilitam a sua aplicação para uma abordagem orientada a objetos, como a não utilização de variáveis globais e a não imposição de capacidade unitária para os *lugares*.

De uma forma geral, nas redes Predicado Transição Diferenciais tem-se que:

- à cada *marca* é associado um vetor de variáveis contínuas;
- à cada *lugar* é associado um sistema de equações diferenciais algébricas que definem a evolução do vetor de variáveis contínuas quando uma *marca* está naquele *lugar*;
- à cada *transição* é associada uma *função de junção*, que define o valor do vetor de variáveis contínuas após o disparo, e uma *função de habilitação*, que habilita ou inibe o *disparo* de *transições* de acordo com os valores do vetor de variáveis contínuas.

3. Orientação a Objetos

De uma forma genérica um objeto pode ser definido como uma entidade que possui atributos e comportamento [Douglass, 1998]. Os atributos representam os dados relativos ao objeto. O comportamento define como o estado do objeto evolui de acordo com a interação do objeto com outros objetos do sistema ou com a evolução do tempo.

Um conceito importante relacionado ao comportamento é o de encapsulamento. Através do encapsulamento define-se um objeto como sendo formado por uma estrutura interna e por uma interface oferecida pelo objeto ao ambiente externo, representada por métodos, através dos quais pode-se ter acesso aos atributos do objeto. A estrutura interna é escondida a fim de garantir a independência da visão externa que se tem do objeto em relação à sua estrutura interna.

[Paludetto, 1991] adiciona ainda a esta definição o conceito de coesão interna. O objeto deve apresentar uma forte coesão interna, no sentido que os dados e as operações que ele contém devem ser fortemente ligados e representar um subconjunto lógico significativo do sistema. Da mesma forma os diferentes objetos devem ser fracamente ligados entre si, no sentido que a interface entre os objetos deve ser a mais simples possível. Esta característica é essencial para que o ganho em simplicidade no modelo do sistema devido a decomposição do sistema em objetos seja superior a complexidade introduzida pela comunicação entre os objetos.

3.1. UML – Unified Modeling Language

Dentro do contexto de modelagem orientada a objetos a UML (Unified Modeling Language) está se tornando, de

fato, um padrão para representação deste tipo de sistema. Entre suas vantagens estão a capacidade de expressar modelos em diferentes níveis de precisão e a capacidade de representar visões diferentes de um mesmo sistema.

UML é uma linguagem de modelagem que visa definir a semântica do modelo objeto e fornecer uma notação para capturar e comunicar a estrutura e o comportamento do objeto. É o resultado da união dos esforços de muitos especialistas em orientação a objeto, entre eles se destacam Grady Booch (Método Booch), Jim Rumbaugh (Object Modeling Technique (OMT)), Ivar Jacobson (Object-Oriented Software Engineering (OOSE)) e David Harel (Statecharts). Observa-se que UML não é uma linguagem visual de programação, no entanto os seus modelos podem ser diretamente conectados a uma variedade de linguagens de programação.

De uma forma geral, pode-se descrever a UML como uma linguagem que, a partir de alguns elementos básicos e dos tipos de relação entre estes elementos, define uma série de diagramas que têm como finalidade a representação de diferentes aspectos de um mesmo sistema. Este texto não pretende apresentar uma revisão bibliográfica de todos os aspectos de todos os diagramas, para tanto cita-se como referência [Booch, Rumbaugh & Jacobson, 1998]. Serão apresentadas apenas algumas noções e conceitos da notação UML.

Na UML, os principais tipos de elementos básicos são elementos estruturais (classes, interfaces, componentes, casos de utilização, entre outros) e elementos relacionados ao comportamento do sistema (interações, estados, etc.). As principais relações definidas são de *associação* e *generalização*. A *associação* descreve uma relação estrutural entre dois elementos. Um tipo de *associação* é a *agregação*, onde um objeto é considerado como “uma parte” de outro objeto. Outro tipo é a *composição*, considerada uma relação de agregação mais forte, onde a “vida” do objeto agregado deve estar contida na “vida” do objeto que o agrega. A *generalização*, também chamada de *especialização* ou *herança*, é uma relação onde objetos especializados (filhos) compartilham a mesma estrutura de um objeto generalizado (mãe).

A linguagem UML define, ao todo, nove diagramas, entre eles, destacam-se aqueles utilizados neste trabalho:

- *Diagrama de classes (Class diagram)*: mostra um conjunto de classes que fazem parte do sistema e as relações entre estas classes. Apresenta uma visão estática do projeto de um sistema.
- *Diagrama de interação (Interaction diagram)*: mostra as interações entre objetos, é constituído de conjunto de objetos e suas relações e inclui as mensagens que podem ser trocadas entre eles. Apresenta uma visão dinâmica de um sistema. Pode ser dividido em *diagrama de seqüência (Sequence diagram)* e *diagrama de colaboração (Collaboration diagram)*. O primeiro enfatiza a ordenação no tempo das mensagens e o segundo a organização estrutural dos objetos que enviam e recebem mensagens.

3.2 UML e Redes de Petri

De acordo com diversos autores ([Douglass, 1998], [Baresi & Pezzè, 1998], [Giese, Graf & Wirtz, 1999]), a UML apresenta ainda muitos pontos a serem evoluídos. Entre os problemas apontados tem-se a não representação de forma satisfatória do comportamento do sistema (visão dinâmica com o tempo). Os recursos da UML para descrever aspectos de comportamento são fracos quando comparados aos recursos para descrever estruturas, em particular no que se refere a descrição de paralelismos, sincronismos, concorrência, etc. entre processos.

Visando solucionar, entre outras coisas, este problema foram desenvolvidos alguns trabalhos onde a UML é utilizada em conjunto com redes de Petri. Em [Giese, Graf & Wirtz, 1999] propõe-se a utilização em conjunto da UML com redes OCN (Object Coordination Net). Em [Baresi & Pezzè, 1998] é proposto o mapeamento automático dos modelos em UML em redes de Petri de Alto Nível, usando a abordagem CR (Customization Rules Approach), que permite aos seus usuários ajustar suas interpretações de uma notação informal a uma notação formal através da definição de um conjunto particular de regras. Na mesma linha que o trabalho de [Baresi & Pezzè, 1998], o trabalho de [Gehrke, Goltz & Wehrheim, 1998] propõe um método para traduzir os *diagramas de atividade* e de *colaboração* em redes de Petri (Lugar/Transição), dando uma interpretação precisa da sua semântica. O trabalho discute como usar os dois tipos de diagramas juntos num processo de desenvolvimento de software. Outra contribuição nesta área se refere ao trabalho de [Paludetto & Delatour, 1999], onde os autores propõem uma metodologia UML/OPN (Object Petri Net) com regras para tradução dos *diagramas de casos de utilização* em redes de Petri. Estes diagramas são um meio relativamente simples de representar informalmente os requisitos de comportamento relacionados a um determinado caso de utilização. Eles constituem o ponto de partida para a geração de redes de Petri, levando a uma modelagem formal onde a validação e verificação podem ser realizadas.

4. Metodologia de Projeto de Sistemas Híbridos

Baseando-se nos trabalhos citados no item 3 foi considerada a incorporação dos conceitos de orientação a objeto às redes Predicado Transição Diferencial para projeto de sistemas supervisórios híbridos.

Para complementar o poder de representação gráfica das redes de Petri no que se refere à representação dos aspectos relacionados à orientação a objetos, e considerando a futura implementação prática do sistema, utilizam-se recursos da UML, introduzindo modificações para a melhor representação dos aspectos híbridos do sistema.

Utilizando como exemplo o sistema de controle de um ar condicionado, apresentam-se as principais características da abordagem proposta.

4.1 O Sistema de Ar Condicionado

O exemplo considerado refere-se ao sistema supervisor do sistema de ar condicionado de um edifício de 3 andares, onde cada andar é dividido em 2 zonas (totalizando 6 zonas no edifício). O sistema de ar condicionado é do tipo VAC (Volume de Ar Constante) e permite apenas resfriamento da zona. Um esquema é apresentado na Figura 1.

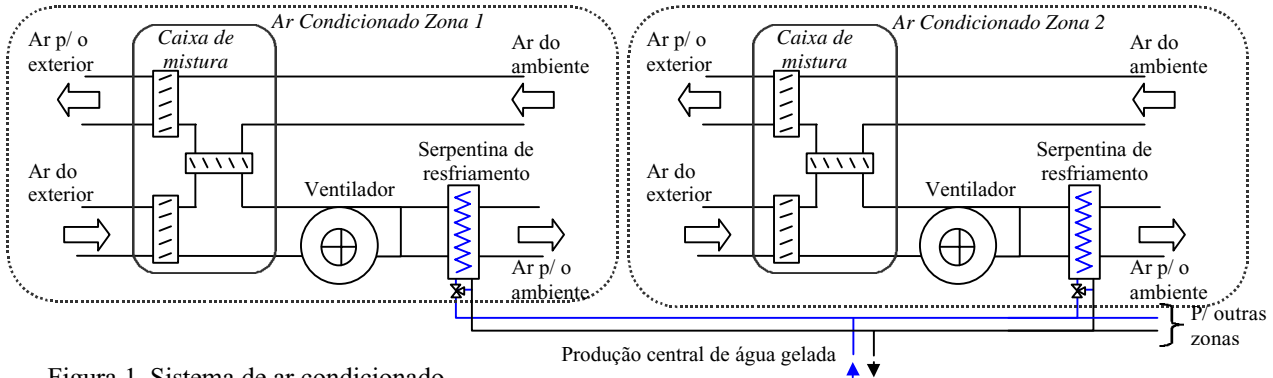


Figura 1. Sistema de ar condicionado.

Cada zona possui:

- Um ventilador, de velocidade constante.
- Uma serpentina de resfriamento cujo fluxo de água é variado de acordo com a posição de uma válvula de 3 vias controlada por um regulador PI em função da temperatura na zona.
- Uma caixa de mistura de 3 posições: sem renovação (0%), renovação parcial (P%) e renovação total (100%).
- Três sensores de temperatura: um para temperatura exterior, um para temperatura na zona e um para temperatura do ar na saída da serpentina, e um sensor de CO₂.

A interface com o usuário é realizada através de um painel de controle presente em cada zona (Figura 2):

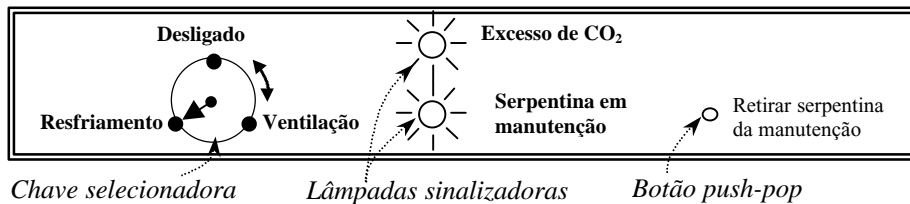


Figura 2. Painel de controle de uma zona.

Este painel possui os seguintes dispositivos:

- Uma chave seccionadora que permite a seleção do modo de operação do ar condicionado entre desligado, apenas ventilação e resfriamento.
- Duas lâmpadas sinalizadoras que indicam excesso de CO₂ na zona e necessidade de manutenção na serpentina.
- Um botão do tipo *push-pop* que recoloca a serpentina em operação após a realização de manutenção.

O sistema de controle a ser projetado deve ter as seguintes características:

- Em cada zona, quando o ar condicionado não estiver desligado e quando a porcentagem de CO₂ no ar ultrapassar um determinado limite, deve-se modificar a posição da caixa de mistura para renovar 100% do ar.
- Quando o condicionamento de uma zona é iniciado, a serpentina deve ser acionada após o ventilador.
- Quando o condicionamento de uma zona é desligado, a serpentina deve ser desligada antes do ventilador.
- A serpentina tem um sistema de detecção de falhas que consiste na simulação de um modelo contínuo onde se avalia a temperatura de saída do ar. Esta temperatura também é monitorada por um sensor. Quando a diferença entre a temperatura prevista e a temperatura real ultrapassa um determinado limite considera-se que ocorreu uma falha, assim o sistema de condicionamento da zona é desligado e a lâmpada sinalizadora correspondente é acesa.

4.2. Definição do Objeto Híbrido

Um primeiro passo antes de abordar o problema de decomposição do sistema é a definição de o que é um objeto do ponto de vista de sistemas híbridos. Para a decomposição de sistemas híbridos, um cuidado particular deve ser tomado no que se refere a interação contínua entre dois objetos. Tem-se então dois tipos de interação:

- **Interações discretas:** a interação pode ser considerada e modelada como eventos discretos (envio de mensagem, recebimento da resposta, etc.). Uma interação discreta corresponde a solicitação realizada por um objeto para execução de um método oferecido pela interface de outro objeto.
- **Interações contínuas:** Uma interação contínua corresponde ao caso em que o estado de um objeto evolui de forma contínua ao longo do tempo e esta evolução contínua depende da evolução ao longo do tempo do estado de um outro

objeto. Um exemplo é apresentado na Figura 3a, onde no modelo de um objeto, as variáveis contínuas que determinam seu estado são calculadas através de equações diferenciais que têm como parâmetros de entrada o valor de variáveis contínuas de outro objeto (x_1 é variável de saída do Objeto 1 e variável de entrada do Objeto 2).

No que se refere a interações contínuas uma atenção especial deve ser dada para o compartilhamento de variáveis contínuas e para os ciclos fechados. Quando a evolução do estado de um objeto depende da evolução do estado de um segundo objeto e a evolução deste depende da evolução do primeiro tem-se um ciclo fechado. Na verdade o que tem-se é que as equações dos dois objetos fazem parte de um único sistema de equações e não podem ser resolvidas de forma independente. Isto interfere diretamente no conceito de independência do objeto. Nestes casos, a evolução do estado dos dois objetos está de tal forma interligada que não existe sentido na definição de dois objetos distintos, sendo mais adequado considerar-se os dois objetos iniciais como um único objeto. Um exemplo é apresentado na Figura 3b. Considera-se, portanto, que a definição dos objetos deve ser realizada de tal forma a evitar as interações contínuas do tipo não permitido.

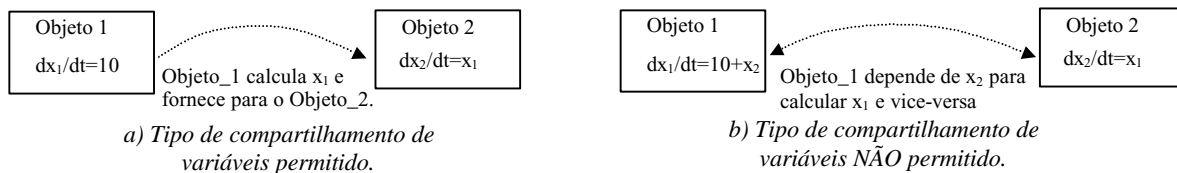


Figura 3. Compartilhamento de variáveis contínuas.

4.3. Decomposição do Sistema

Considerando as diversas etapas do desenvolvimento de um software de controle, a modelagem é uma atividade que pertence principalmente às fases de análise dos requisitos (onde identifica-se objetos e classes do modelo) e de projeto (onde o modelo é construído e, eventualmente, simulado, analisado, validado, etc.). A fase de programação (onde implementam-se os modelos construídos em uma linguagem orientada a objetos) não é considerada neste trabalho. No que se refere a decomposição do sistema, para a modelagem do sistema durante o projeto não aborda-se inicialmente o conceito de herança, que corresponde a inserção do objeto dentro de uma hierarquia de classe, uma vez que esta pode ser considerada uma atividade predominantemente relacionada à fase de programação [Booch, 1994].

Em contrapartida, para a definição de objetos híbridos, prioriza-se a decomposição do sistema segundo relações de agregação e composição entre os objetos. Neste sentido deve-se identificar quais são os candidatos a objetos do sistema e como estes candidatos podem ser decomposto em novos objetos. Desta forma busca-se, através de um refinamento sucessivo, trabalhar a complexidade do sistema.

Tomando como exemplo o sistema de ar condicionado de uma zona, este pode ser inicialmente dividido em três objetos, cuja descrição simplificada é apresentada abaixo:

Objeto 1: Ventilação:

Responsabilidade: circulação e renovação do ar.

Métodos: circular ar com renovação parcial / circular ar com renovação total / não circular o ar.

Objeto 2: Resfriamento

Responsabilidade: controlar a temperatura do ambiente através do resfriamento do ar

Métodos: resfriar o ambiente / suspender o resfriamento.

Objeto 3: Interface

Responsabilidade: indicar a ocorrência de problemas e possibilitar ao usuário modificar a configuração do sistema.

Métodos: acionar resfriamento e ventilação / acionar só resfriamento / desligar o sistema.

A interação entre estes objetos é do tipo discreto, a única exceção se refere a quando o sistema estiver no modo resfriamento, neste caso o Objeto 1 deve fornecer ao Objeto 2 a temperatura do ar na saída da caixa de mistura, para que o Objeto 2 possa estimar a temperatura na saída da serpentina e assim detectar falhas através da comparação desta temperatura estimada com a temperatura real do ar.

Cada um destes sub-sistemas pode ainda ser decomposto em objetos que tenham uma maior relação com os dispositivos físicos do sistema (dispositivos de atuação e de monitoração). Tem-se então uma nova decomposição do sistema onde cada objeto é responsável por fazer a comunicação do sistema de controle com os dispositivos da planta. Ele deve também fornecer para o restante do sistema os métodos correspondentes as possíveis modificações que o sistema de controle possa fazer no modo de operação de cada dispositivo. Além disso, ele deve armazenar as informações relativas ao dispositivo que são de interesse para o sistema de controle. Uma proposta de decomposição é apresentada no diagrama da Figura 4.

Para cada objeto do Nível 2 da Figura 4 apresenta-se a descrição de sua interface, usando a notação UML. Esta descrição é composta por atributos, que podem ser variáveis discretas, variáveis contínuas ou constantes e pelos métodos oferecidos.

Para a classe “Caixa de Mistura” (Figura 5) os atributos disponíveis são a porcentagem de renovação do ar quando a caixa está na posição de renovação parcial (P , constante), a temperatura do ar na saída da caixa de mistura calculada pelo objeto ($T_{\text{caixa_calc}}$, variável contínua), e a quantidade máxima de CO_2 permitida no ambiente ($Q_{\text{CO}_2\text{max}}$, constante). Os métodos disponíveis se referem a modificação da posição da caixa de mistura e a leitura da variável $T_{\text{caixa_calc}}$.

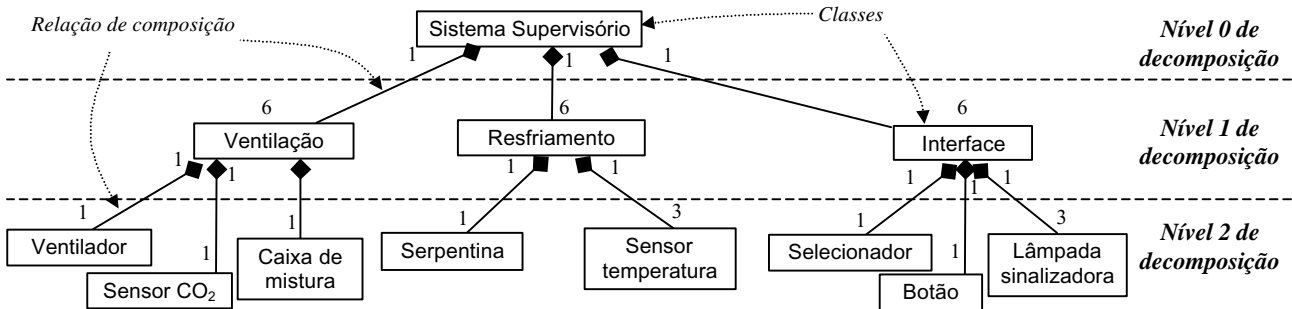


Figura 4. Decomposição estrutural do sistema supervisório representada em um diagrama de classes.

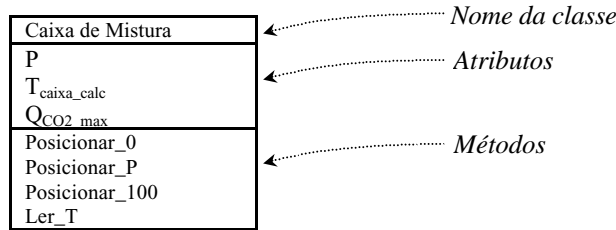


Figura 5. Definição da classe “Caixa de Mistura”.

De modo semelhante define-se os demais objetos (Figura 6):

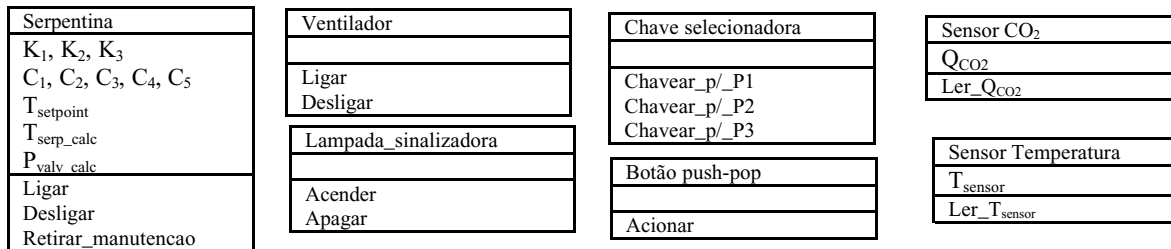


Figura 6. Definição das demais classes.

Nas descrições acima tem-se que

- K_1, K_2, K_3 representam as constantes do controlador da serpentina;
- C_1, C_2, C_3, C_4 e C_5 são constantes utilizadas no cálculo da temperatura do ar na saída da serpentina;
- $T_{setpoint}$ (constante) é o *set-point* da temperatura do ambiente utilizado pelo regulador PI;
- T_{serp_calc} (variável contínua) representa a temperatura calculada do ar na saída da serpentina;
- P_{valv_calc} (variável contínua) é a posição da válvula calculada pelo objeto;
- T_{sensor} (variável contínua) representa a temperatura lida pelo sensor;
- Q_{CO_2} (variável contínua) representa a quantidade de CO_2 detectada pelo sensor;
- Posições da chave selecionadora: P1 – desligado, P2 – resfriamento, P3 – ventilação.

Uma vez definidos os candidatos iniciais a objetos, passa-se então a modelagem dos seus comportamentos em Redes de Petri. Observa-se que, neste exemplo, a decomposição do sistema foi realizada de forma intuitiva. Neste caso através da fusão dos modelos dos objetos de um nível inferior, é possível verificar-se a coerência do nível superior. Uma outra abordagem possível é a construção dos modelos em redes de Petri de um nível superior para utilizá-los como base para a sua decomposição, como proposto em [Paludetto, 1991].

4.4. Modelagem dos comportamentos dos objetos

Para a representação da parte híbrida utiliza-se as redes de Petri e das redes Predicado Transição Diferenciais como suporte para formalização dos modelos representados nos diagramas UML.

Assim, uma vez definidos os candidatos a objetos, procede-se ao detalhamento dos modelos. A modelagem dos comportamentos em redes de Petri é realizada considerando que:

- A estrutura da rede define as possíveis evoluções para o estado de um objeto pertencente a uma determinada classe, ou seja, a rede modela o comportamento de uma classe.
- Cada *marca* presente nesta rede corresponde a instanciação de um objeto daquela classe.
- As *marcas* possuem atributos que correspondem aos atributos da classe que a define. Portanto, os atributos podem ser variáveis discretas (alteradas pelo disparo das transições), variáveis contínuas (alteradas pelas equações associadas aos *lugares*) ou constantes.
- Do ponto de vista da rede de Petri, a identificação do objeto é um atributo adicional constante da marca.
- A chamada de um método corresponde ao *disparo* de uma *transição*.
- A interface oferecida por uma classe corresponde a um conjunto de *transições*, cada uma correspondente a um método oferecido.

Para os objetos do Nível 2 da Figura 2 que apresentam comportamento puramente discreto tem-se os modelos em redes de Petri ilustrados na Figura 7. Conforme já definido, cada uma das *marcas* presentes nas redes da Figura 7 apresentam como atributos os atributos definidos nos modelos do item 4.2, mais um atributo adicional que corresponde a identificação do objeto (nome).

Para os objetos híbridos os modelos são construídos usando redes Predicado-Transição Diferenciais (Figura 8).

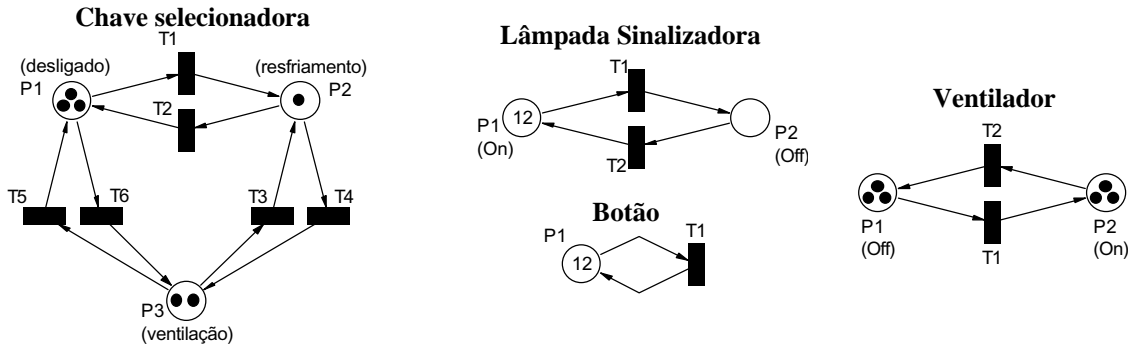


Figura 7. Modelo em redes de Petri dos objetos discretos.

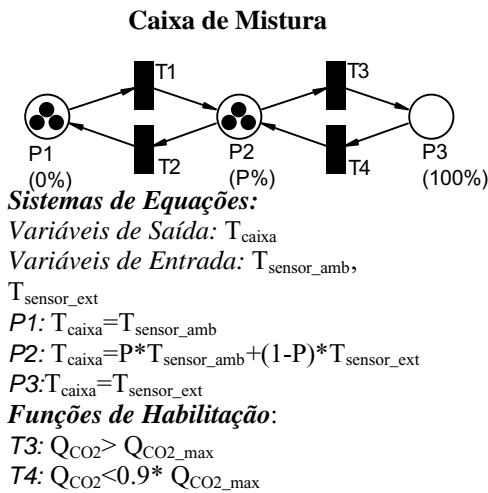


Figura 8. Modelo em redes de Petri dos objetos híbridos.

Observa-se que no modelo acima as variáveis utilizadas nos sistemas de equações correspondem a objetos de uma mesma zona. Por exemplo, no modelo da serpentina, quando uma *marca* estiver em P2, na resolução do sistema de equações será utilizada a variável T_{caixa} da marca correspondente a mesma zona. As variáveis T_{sensor_amb} , T_{sensor_serp} e T_{sensor_ext} correspondem às variáveis das diversas instâncias da classe “Sensor Temperatura” relacionadas aos diversos sensores de temperatura de uma mesma zona

Os sensores são objetos puramente passivos, cuja única variável tem seu valor atualizado constantemente através de comunicação com o dispositivo físico de detecção. Assim, não é necessário a especificação do seu comportamento.

Os modelos de comportamento acima propostos apresentam os possíveis estados dos objeto pertencentes a uma determinada classe e como estes estados podem ser modificados pela execução de métodos (associados a transições). No entanto não apresentam informações sobre como os objetos se relacionam entre si. Para tanto, deve acrescentar informações sobre as seqüências de atividades, ou sejam, sobre como os diversos processos são realizados.

4.5 Detalhamento dos Processos

Uma segunda etapa na modelagem do sistema consiste na modelagem dos processos envolvidos para um posterior detalhamento dos modelos dos objetos. Exemplos de processos (casos de utilização) são “ligar o resfriamento”, “detectar falha na serpentina”, “desligar o ar condicionado”, “detectar excesso de CO₂”, etc.

Para tanto, constrói-se inicialmente um modelo em redes de Petri de cada processo. Neste modelo, cada *transição* representa uma atividade desenvolvida. Os *lugares* representam estados e podem, eventualmente, estar relacionados a interações contínuas realizadas entre dois objetos. Estes modelos não incluem os recursos utilizados para as atividades, incluem apenas o sequenciamento de atividades. Como exemplo apresenta-se o modelo do processo “ligar o resfriamento”.

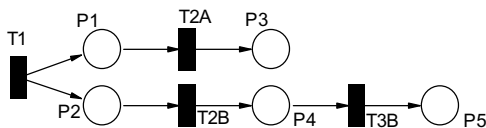


Figura 9. Modelo do processo “ligar o resfriamento”.

As *transições* acima representam os seguintes eventos:

- T1: usuário modifica a posição da chave seletora de “desligado” para “resfriamento”;
- T2A: modificação da posição da caixa de mistura para renovação P%;
- T2B: acionamento do ventilador;
- T3B: acionamento da serpentina.

Dos *lugares* acima, tem-se interações contínuas nos seguintes estados:

- P3: os dois objetos “Sensor Temperatura” correspondentes aos sensores de temperatura exterior e de temperatura ambiente devem fornecer as variáveis “ $T_{\text{sensor_amb}}$ ” e “ $T_{\text{sensor_ext}}$ ” para o objeto “caixa de mistura”, e o objeto “Sensor CO₂” deve fornecer a variável “ Q_{CO_2} ”;
- P5: o objeto “caixa de mistura” deve fornecer a variável “ T_{caixa} ” e o objeto “Sensor Temperatura” correspondente a temperatura ambiente deve fornecer a variável “ $T_{\text{sensor_amb}}$ ” para o objeto “serpentina”.

A partir da descrição do processo procede-se a construção de um diagrama de interação, onde associa-se uma transição ou um conjunto de transições à um método que é chamado por um objeto. Para o processo da Figura 9 tem-se o seguinte diagrama de colaboração:

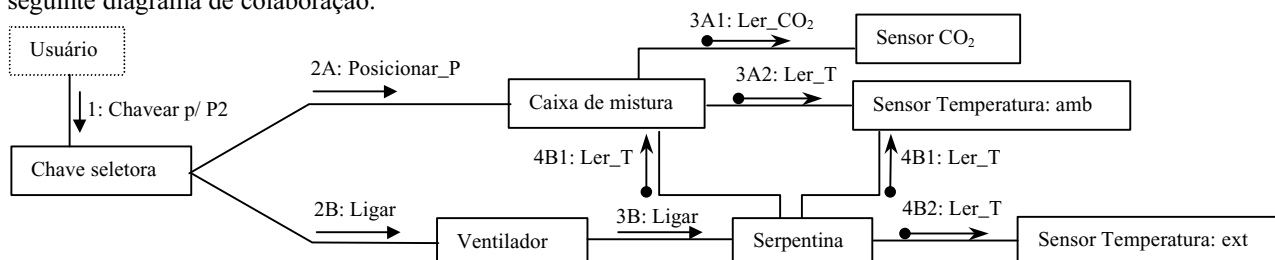


Figura 10. Diagrama de colaboração do processo “ligar o resfriamento”.

De acordo com a notação UML, neste diagrama utilizou-se a o arco \longrightarrow para indicar uma interação discreta síncrona. Para indicar uma interação contínua, uma vez que a UML não dispõe de uma notação específica para este caso, define-se a utilização do arco $\bullet\longrightarrow$. Observa-se que o arco tem origem no objeto que solicita a informação do valor de uma variável e tem destino no objeto que fornece a informação.

Uma vez construído o diagrama de interações, detalha-se os diversos métodos do modelo em rede de Petri de cada objeto, associando à *transição* correspondente a respectiva parte do modelo em rede de Petri do processo. Como exemplo apresenta-se o detalhamento da *transição* T1 do modelo da chave seccionadora” (Figura 11). Neste modelo, a *transição* T2A representa a execução do método “Posicionar_P” e T2B a “Ligar” relativo ao ventilador:

O detalhamento apresentado acima deve ser realizado para todos os métodos correspondentes às interações discretas. Os métodos correspondentes às interações contínuas não necessitam de detalhamento, uma vez que sua execução está implícita nos sistemas de equações.

Para que os modelos sejam consistentes, o *disparo* de *transições* que adicionam *marcas* a um *lugar* do processo onde existe interação contínua (no exemplo da Figura 9, P3 e P5) deve corresponder a execução de um método da interface de um objeto híbrido (T2A – Posicionar_P, da caixa de mistura, e T3B – Ligar, da serpentina). O novo sistema de equações a ser resolvido do objeto híbrido deve ter como variáveis de entrada, variáveis de saída dos objetos com os quais realiza as interações contínuas representadas no diagrama de colaboração. Além disto, os objetos que fornecem as variáveis de entrada devem ter entre os seus métodos aqueles correspondentes a leitura de suas variáveis de saída.

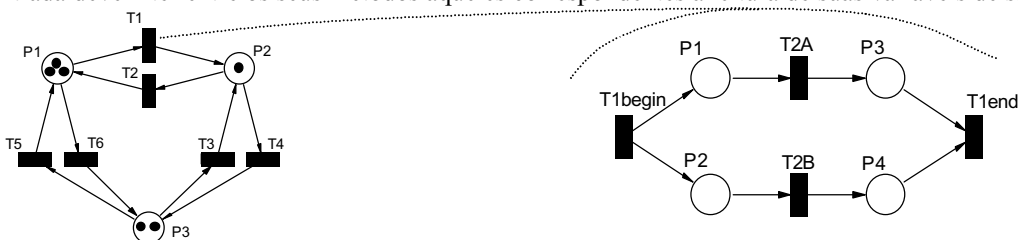


Figura 11. Adição de informações do processo aos modelos dos objetos.

4.6 Modelagem da comunicação entre objetos

Uma vez adicionada as informações sobre o processo, o passo seguinte refere-se a definir como, de fato, as diversas redes de Petri se relacionam entre si. Para tanto utiliza-se os modelos propostos em [Paludetto, 1991], onde a comunicação é realizada através de um fluxo de *marcas* entre objetos. Um exemplo é apresentado na Figura 12.

Uma outra opção para modelagem da comunicação entre os objetos é através de arcos habilitadores. Neste caso, a presença de uma *marca* em P3 habilitaria o *disparo* de T1 e a presença da *marca* correspondente em P2 habilitaria o *disparo* de T1end.

Quanto a comunicação contínua entre objetos, para que os modelos sejam consistentes, o *disparo* de *transições* que adicionam *marcas* a um *lugar* do modelo do processo onde existe interação contínua (no exemplo da Figura 9, P3 e P5) deve corresponder a execução de um método da interface de um objeto híbrido (T2A – Posicionar_P, da caixa de

mistura, e T3B – Ligar, da serpentina). O novo sistema de equações a ser resolvido do objeto híbrido deve ter como variáveis de entrada, variáveis de saída dos objetos com os quais realiza as interações contínuas representadas no diagrama de colaboração. Além disto, os objetos que fornecem as variáveis de entrada devem ter entre os seus métodos aqueles correspondentes a leitura de suas variáveis de saída.



Figura 12. Modelagem da comunicação discreta entre os objetos através fluxo de marcas.

5. Considerações sobre a Análise do Sistema

No que se refere a análise dos modelos desenvolvidos, esta parte ainda está em desenvolvimento. No entanto, algumas conclusões já foram obtidas.

No que se refere a verificação da coerência dos modelos desenvolvidos, métodos podem ser propostos para verificar por exemplo, que não existam ciclos de compartilhamento de variáveis contínuas, ou que um objeto, em um determinado instante, utilize no seu sistema de equações uma variável externa que não está sendo calculada em nenhum outro objeto naquele instante.

Para a análise formal das propriedades da rede global (vivacidade, limitabilidade e reiniciabilidade), não existem métodos que garantam estas propriedades considerando a natureza híbrida do modelo. A determinação da habilitação ou não de uma transição depende da evolução no tempo das variáveis contínuas envolvidas na função de habilitação. Esta evolução é determinada pelos sistemas de equações. Como, para manter a flexibilidade do modelo, não foram impostas restrições quanto ao tipo de sistema de equação diferencial (se de 1ª ordem, 2ª ordem, não-linear, etc.), a resolução destes sistemas não pode ser garantida a não ser por simulação, o que inviabiliza uma solução teórica.

A análise deverá então ser realizada por etapas. Inicialmente será analisado o comportamento discreto de cada objeto isto é, da rede de Petri ordinária (ou predicado-transição) interna de cada objeto. Para tanto, cada objeto deve ser relativamente independente. Através da fusão sucessiva de objetos (observando, por exemplo a hierarquia de decomposição do sistema) deverá ser realizada a análise global do modelo. Observa-se no entanto, que a análise formal global só poderá ser realizada para a rede ordinária, o que dará uma indicação da coerência do modelo desenvolvido.

A parte continua será integrada na análise através a simulação. Poderá então verificar-se que uma determinada seqüência da rede de Petri ordinária global permanece possível tomando em conta a dinâmica continua dos modelos.

6. Conclusões

Este trabalho discute a aplicação dos conceitos de orientação a objetos para sistemas híbridos, propondo uma abordagem para construção de modelos baseados em redes de Petri e equações diferenciais. A notação UML é utilizada para auxiliar a representação de diferentes aspectos do sistema.

A utilização em conjunto de redes de Petri e orientação a objetos visa garantir a consistência dos modelos gerados para sistemas onde os processos e os recursos utilizados apresentam uma elevada complexidade. Neste sentido os modelos em redes de Petri são efetivos como elo de ligação entre os diferentes diagramas UML. Por sua vez a UML, através de suas visões complementares auxilia na representação da estrutura do sistema e na incorporação de informações sobre a seqüência de atividades, garantindo assim que os requisitos iniciais do sistema sejam obedecidos.

Em particular no que se refere a sistemas híbridos, através do exemplo do sistema de ar condicionado, verifica-se que a UML e as redes Predicado Transição Diferenciais se complementam, possibilitando o enfoque dos diferentes aspectos híbridos do sistema. De um lado as redes Predicado Transição Diferenciais explicitam os estados onde acontece uma evolução contínua do estado do objeto, de outro lado a UML, com pequenas modificações possibilita a representação, de uma forma explícita, da interação contínua entre os objetos.

Um resultado importante da aplicação dos conceitos de orientação a objetos é que deriva-se um sistema de inteligência distribuída, uma vez que cada objeto deve ter autonomia para gerenciar seus recursos e executar suas funções, ao contrário da abordagem funcional onde tem-se procedimentos complexos que controlam totalmente a execução de procedimentos de baixo nível.

Finalmente, observa-se que este trabalho é fortemente motivador para o desenvolvimento de uma metodologia para projeto de sistemas de controle híbridos que a partir dos modelos construídos possibilite a obtenção do software de controle através da implementação dos modelos em redes de Petri e em redes Predicado Transição Diferenciais.

7. Agradecimentos

Os autores agradecem o apoio das entidades FAPESP, CNPq e CAPES para o desenvolvimento deste trabalho.

5. Referências

- Alla, H. & David, R., 1998. "Continuous and Hybrid Petri Nets" *Journal of Circuits, Systems and Computers*, vol.8, n.1, pp 159-188.
- Antsaklis, P. J. & Nerode, A., 1998. "Hybrid Control Systems: An Introductory Discussion to the Special Issue" *IEEE Transactions on Automatic Control*, vol 43, n.4, pp 457-459.
- Baresi, L. & Pezzè, M., 1998. "On Formalizing UML with High-Level Petri Nets". *Concurrent Object-Oriented Programming and Petri Nets - Lecture Notes in Computer Science - Springer Verlag* (in press).
- Booch, G., 1994. "Object-Oriented Analysis and Design with Applications", 2nd ed. Addison-Wesley Longman, Inc. Harlow, England.
- Booch, G., Rumbaugh, J, Jacobson, I., 1998. "The Unified Modeling Language User Guide", Addison-Wesley Longman, Inc. Harlow, England.
- Champagnat, R., 1998. "Supervision des Systèmes Discontinus: Definition d'un Modèle Hybride et Pilotage en Temps-réel" Thèse de Doctorat, Université Paul Sabatier, Toulouse, France.
- Douglass, B. P., 1998. "Real-time UML: developing efficient objects for embedded systems", Addison-Wesley Longman, Inc. Harlow, England.
- Fraser, C.J. et al., 1993. "An Educational Perspective on Applied Mechatronics" *Mechatronics*, vol..3, n.1, pp.49-51.
- Garcia, H. E., 1997. "A hierarchical platform for implementing Hybrid Systems in Process Control", *Control Eng. Practice*, vol.5 n.6, pp 779-789.
- Gehrke, T., Goltz, U. & Wehrheim, H., 1998. "The Dynamic Models of UML: Towards a semantic and its application in the Development Process", *Hildesheimer Informatik-Bericht 11/98*, Institut für Informatik, Universität Hildesheim, Germany.
- Giese, H., Graf, J., Wirtz, G., 1999. "Closing the gap between object-oriented modeling of structure and behaviour", *Proceedings of the Second International Conference on the Unified Modeling Language*, Colorado, USA.
- Guéguen, H. & Lefebvre, M., 2000. "A comparison of mixed specification formalisms", *Proc. ADPM 2000 - The 4th International Conference on Automation of Mixed Processes: Hybrid Dynamic Systems*, Dortmund, Germany.
- Lemmon, M., He, K. X. & Markovsky, I., 1999. "Supervisory Hybrid Systems", *IEEE Control System*, vol.8 n.1, pp 42-55.
- Nimmo, I., 1999. "Future of Supervisory Systems in Process Industries: Lessons for Discrete Manufacturing", *Annual Reviews in Control*, vol.23, pp 45-52.
- Paludetto, M., 1991. "Sur la commande des procédés industriels: une méthodologie basée objects et réseaux de Petri" Thèse de Doctorat, Université Paul Sabatier, Toulouse, France.
- Paludetto, M. & Delatour, J., 1999. "UML et les réseaux de Petri – vers une sémantique des modèles dynamiques et une méthodologie de développement des systèmes temps réel", *L'object*, vol.5 n.3, pp 443-468.
- Valentin-Roubinet, C., 2000. "Hybrid Dynamic System verification with Mixed Petri Nets", *Proc. ADPM 2000 - The 4th International Conference on Automation of Mixed Processes: Hybrid Dynamic Systems*, Dortmund, Germany.

PETRI NETS AND OBJECT-ORIENTATION FOR HYBRID CONTROL SYSTEM DEVELOPMENT

Villani, Emilia

Dept. Eng. Mecatrônica e Sistemas Mecânicos, Escola Politécnica, USP
Av. Prof. Mello Moraes, 2001 São Paulo Brasil
emiliav@usp.br

Miyagi, Paulo Eigi

Dept. Eng. Mecatrônica e Sistemas Mecânicos, Escola Politécnica, USP
Av. Prof. Mello Moraes, 2001 São Paulo Brasil
pemiyagi@usp.br

Valette, Robert

LAAS – CNRS (Laboratoire d'Analyse et d'Architecture des Systèmes)
7, Avenue du Colonel Roche, 31077 Toulouse Cedex 4 FRANCE
robert@laas.fr

Abstract. *This paper introduces a new approach for the development of hybrid control systems. The word "hybrid" means that the systems embody, simultaneously, characteristics of discrete event dynamic systems (where events are instantaneous and state variables are discrete numbers) and continuous systems (where events and state variables are continuous). Taking as a starting point a proposal of Petri Net associated with differential equations, we search a solution for the project of complex industrial systems by using a more structured approach. This new approach introduces the object-orientation concepts to the use of Petri Net associated with differential equations. The language UML is used in order to support the description of different aspects of the system. Through an example, we shown that UML and Petri Nets are complementary. By using both languages, it is possible to highlight different hybrid characteristics of the system.*

Keywords. *Hybrid systems, object orientation, Petri nets, UML*