UTILIZAÇÃO DA TÉCNICA ESTELLE NA VALIDAÇÃO DE PROTOCOLOS DE ALTO NÍVEL: METODOLOGIA, FERRAMENTAS E EXPERIÊNCIA

V. B. MAZZOLA[1], L. F. R. C. CARMO[2], P. de SAQUI-SANNES[3], J-P. COURTIAT[3]

[1]Laboratório de Controle e Microinformática
Departamento de Engenharia Elétrica
Universidade Federal de Santa Catarina
CEP 88000- Florianópolis - SC - Brasil

[2]Núcleo de Computação Eletrônica
Universidade Federal do Rio de Janeiro
Rio de Janeiro - RJ - Brasil

[3]Laboratoire d'Automatique et d'Analyse des Systèmes (LAAS/CNRS)
7, Avenue du Colonel-Roche - 31077 Toulouse Cedex - France

RESUMO

Este artigo apresenta uma metodologia e as ferramentas associadas para a validação de protocolos de comunicação das camadas altas do modelo OSI (Open Systems Interconnection), que apresentam particularidades em relação aos protocolos das camadas mais baixas. A metodologia é baseada na utilização de uma técnica de descrição formal normalizada na ISO (International Standards Organization) sobre a qual um grande número de experiências tem sido realizadas com sucesso - a técnica Estelle. Um exemplo de aplicação da metodologia e da ferramenta sobre a norma MMS, da camada de Aplicação da arquitetura MAP é apresentado também neste trabalho.
PALAVRAS CHAVE: Engenharia de Protocolos, Estelle, ESTIM, Verificação, OSI, MMS.

ABSTRACT

This paper proposes a validation approach and the associated tools for high level protocols defined within the OSI (Open Systems Interconnection) framework, which present particular features with respect the lowest protocol layers. This approach is based on the use of a formal description technique (FDT), standardized at ISO (International Standards Organization), on which a lot of experiments have been sucessfully done - the Estelle FDT. The use of this methodology is illustrated on a significant example, the MMS services, defined at the Application Layer of the MAP architecture.
KEYWORDS: Protocol Engineering, Estelle, ESTIM, Verification, OSI, MMS.

1. INTRODUÇÃO

Nos últimos anos, um grande número de trabalhos têm sido realizados, visando a concepção de entidades de protocolos de comunicação. Na verdade, a concepção de protocolos de comunicação nada mais é que a aplicação dos conceitos de Engenharia de Programação (Software Engineering) ao caso particular dos protocolos de comunicação, levando em conta as especificidades e requerimentos destes sistemas. O nome *Engenharia de Protocolos* (Protocol Engineering) tem rotulado o conjunto de trabalhos realizados nesta área.

Dentre o extenso conjunto de trabalhos realizados, podemos destacar principalmente aqueles relacionados à definição e utilização de modelos formais para a especificação, validação, implementação e testes de protocolos de comunicação. Nos últimos 10 anos, tem-se notado a larga utilização de modelos baseados em sistemas de transição, tais como o das máquinas de estado finito [1], redes de Petri [2], [3], e, mais recentemente, as chamadas Técnicas de Descrição Formal (FDT, *Formal Description Techniques*) normalizadas na ISO (International Standards Organization) e no CCITT (Comité Consultatif International Télégraphique et Téléphonique). No primeiro caso estão as FDTs Estelle e Lotos [4], [5], e no segundo caso, SDL [6].

Este artigo propõe uma metodologia de validação de protocolos de comunicação, baseada na utilização da técnica Estelle e orientada aos protocolos definidos para as camadas mais altas do modelo OSI.

A seção 2 apresenta, de forma sucinta os principais mecanismos relacionados com a técnica de descrição Estelle, assim como alguns trabalhos realizados em torno desta técnica, particularmente, a definição da técnica Estelle* e do ambiente de validação Estim.

A seção 3 apresenta a metodologia proposta no trabalho, que leva em conta, especificidades dos protocolos das camadas mais altas do modelo OSI.

Na seção 4, é apresentada uma ilustração da metodologia apresentada em 3, o caso estudado sendo os serviços MMS, definidos na camada de Aplicação da arquitetura MAP, assim como de outras arquiteturas de comunicação para os sistemas integrados de manufatura.

Finalmente, na seção 5, são apresentadas as conclusões e algumas propostas de continuidade do trabalho.

2. APRESENTAÇÃO DA TÉCNICA ESTELLE

2.1. Principais conceitos da FDT Estelle

Uma especificação Estelle [4], [7], [8], descreve um sistema estruturado hierarquicamente de componentes sequenciais não deterministas (chamados instâncias de módulos) os quais trocam mensagens (chamadas interações) através de links bidirecionais entre suas portas (chamados pontos de interação). Tanto a hierarquia dos módulos como a estrutura dos links entre eles podem ser modificados em tempo de execução, dando ao sistema uma característica dinâmica. Um módulo é definido por uma definição de tipo (module_header_definition) e uma descrição de comportamento interno associada ao tipo (module_body_definition). Várias instâncias de um módulo podem ser criadas e estar presentes simultaneamente durante a execução de uma especificação Estelle. Neste caso, estas instâncias apresentarão a mesma visibilidade externa e o mesmo comportamento interno caracterizados, respectivamente, pelo tipo e pelo corpo de módulo.

Vista do exterior, uma instância de módulo é uma "caixa preta", acessível através de conjuntos finitos de pontos de interação. Cada ponto de interação de uma instância de módulo x está associado a uma fila FIFO que recebe e armazena as interações enviadas a x naquele ponto. Uma instância de módulo pode também enviar interações a outras instâncias de módulo através de seus próprios pontos de interação. O conjunto de interações (e dos parâmetros associados) podendo ser enviadas e recebidas num dado ponto de interação são determinados pela definição do canal associado àquele ponto de interação.

Os resultados visíveis do comportamento de uma instância de módulo na forma das interações que esta envia, são um efeito de uma atividade interna desta instância, descrita na definição do corpo do módulo associada (module_body_definition). O comportamento interno de uma instância de módulo é caracterizado por uma descrição em máquina de estado extendida [8]. Informalmente, cada estado local de uma instância de módulo tem uma estrutura complexa caracterizada pelos seguintes componentes:

- uma parte *controle*, representada por um valor que caracteriza o estado de controle (ou *major state*) de uma instância de módulo;
- uma parte *ambiente_de_entrada*, representada pelos conteúdos de todas as filas de entrada associadas com os pontos de interação desta instância de módulo;
- uma parte *dados*, representada pelos valores das variáveis declaradas na definição do comportamento da instância de módulo;
- uma parte *temporização* (delay), representada pelos valores dos intervalos de tempo associados às transições da instância de módulo.

A parte *declaração* de um corpo de módulo inclui a declaração das contantes, tipos de dados, a definição dos tipos de canais, das funções e procedimentos Pascal, e, finalmente, a declaração dos estados de controle (major states) e das variáveis locais.

A parte de *inicialização* de um corpo de módulo especifica um estado de controle inicial, os valores iniciais das variáveis e define a hierarquia e estrutura de interconexão iniciais das instâncias de módulos descendentes (se existirem).

A execução de uma *transição* para uma instância de módulo é considerada uma operação atômica. Uma vez iniciada a execução, ela não pode ser interrompida, e, conceitualmente, não se pode conhecer resultados intermediários independente da sua duração.

2.2. Os trabalhos em torno de Estelle

Após sua definição, o desenvolvimento de uma técnica de descrição formal na direção de uma possível utilização industrial pode dar-se considerando três principais aspectos:

- desenvolvimento da semântica formal;
- definição de metodologias e de ferramentas de software para o suporte à concepção;
- desenvolvimento de aplicações realistas utilizando a técnica considerada.

No que diz respeito ao primeiro aspecto, um importante resultado foi a definição de uma semântica formal, baseada em redes de Petri para Estelle [9], trabalho realizado no contexto de um projeto europeu, denominado SEDOS [10].

A continuidade deste trabalho permitiu a definição de um dialeto ou de uma nova versão da técnica, denominada Estelle* [11]. Estelle* foi definida a partir do conhecimento da semântica de Estelle, tendo por principal objetivo aumentar o seu poder de expressão e simplificar o trabalho de especificação do usuário. Dentre as contribuições ao poder de expressão da técnica, pode-se destacar a introdução de um mecanismo síncrono de comunicação, denominado *Rendez-Vous*, implementado através de cláusulas de sincronização de envio e de recepção adicionadas à parte pré-condição das transições de uma instância de módulo.

Com relação ao segundo aspecto, tem-se observado na literatura um grande conjunto de ambientes de desenvolvimento disponíveis atualmente para a técnica Estelle, alguns dos quais, verdadeiros produtos comerciais. Um resultado importante destes trabalhos foi o ambiente ESTIM [13], que teve seu desenvolvimento iniciado no contexto de SEDOS.

ESTIM (*Estelle SimulaTor based on an Interpretative Machine*) foi desenvolvido no LAAS/CNRS (Toulouse - França), apresentando facilidades para a validação de especificações formais em Estelle. A ferramenta foi escrita em ML e suporta principalmente os mecanismos definidos para Estelle*, descrita anteriormente. No que diz respeito à simulação, ESTIM suporta dois modos de pilotagem de uma sessão de simulação:

- uma simulação *passo-a-passo*, na qual o usuário escolhe, no conjunto das transições sensibilizadas, qual transição deverá executar; este modo permite a definição e a condução de cenários de simulação, permitindo validar muitos aspectos do sistema especificado; por outro lado, ele exige a presença do usuário da ferramenta durante toda a sessão de simulação;
- uma simulação *automática*, na qual o usuário define um número *N* de transições a serem executadas, a escolha, neste, caso, sendo aleatória, a ferramenta sendo encarregada desta tarefa; neste caso, a sessão é encerrada quando, ou o número de transições requisitadas foi executado, ou uma situação de bloqueio (*deadlock*) foi encontrada.

Durante uma sessão de simulação, ESTIM permite o acesso a todos os objetos, Pascal e Estelle, compondo a especificação — variáveis, conteúdos de filas FIFO, estados globais, etc — possibilitando a fácil identificação de erros de especificação. Ainda, ele permite o acesso a

informações estatísticas sobre a sessão de simulação, como, por exemplo, a lista das transições já executadas, a lista das transições que não foram executadas, instâncias de módulo criadas, e outras informações mais.

ESTIM implementa igualmente uma técnica de verificação, através da construção de um grafo de alcançabilidade a partir da especificação Estelle* considerada. Este grafo é reduzido através da interface de ESTIM com ferramentas dedicadas (PIPN, ALDEBARAN, etc...) que implementam técnicas de redução por equivalência de processos. Deste modo, o grafo obtido (autômato equivalente) pode ser analizado para a verificação das propriedades do protocolo. O modo de utilização de ESTIM em verificação será apresentado mais adiante neste documento. A figura 1 apresenta um esquema das funcionalidades desta ferramenta.

Com relação, ao terceiro e último aspecto, vários projetos tiveram por objetivo o desenvolvimento de especificações formais em Estelle para fins de implementação. Um resultado importante destes trabalhos foi o desenvolvimento do sistema de controle de uma Célula Flexível de Usinagem [14], que permitiu mostrar a aplicabilidade de Estelle a outros domínios que o dos Protocolos de Comunicação.

3. METODOLOGIA DE VALIDAÇÃO BASEADA EM ESTELLE*

3.1. As etapas da concepção dos protocolos de comunicação

O objetivo da concepção de uma camada de protocolos é gerar um programa concretizando uma entidade de protocolos que ofereça os serviços requeridos para o nível considerado. A concepção deve ser realizada, seguindo um conjunto de procedimentos, ou etapas, que permitam atingir este objetivo num menor tempo e com um maior grau de confiabilidade e de , esta segunda característica sendo definida mais à frente.

A concepção de uma camada de protocolo segue os mesmos procedimentos definidos pelas técnicas de Engenharia de Programação, onde as principais etapas são:

a *etapa de análise de requisitos*, onde são levantados todos os pontos relativos ao problema a ser resolvido, o resultado deste levantamento sendo a realização de uma especificação funcional;

Figura 1 - Funcionalidades da ferramenta ESTIM.

- a *etapa de projeto*, caracterizada pela realização de uma especificação detalhada, descrevendo, se possível, de maneira não ambígua, o comportamento de cada componente do software a ser realizado e as suas interações para a resolução dos problemas levantados na primeira etapa;
- a *etapa de implementação*, que consiste na geração e implantação do código executável, o software devendo ser o mais fiel possível do que foi definido na especificação detalhada;
- a *etapa de manutenção do software*, onde atualizações podem ser introduzidas de modo a se obter novas versões, mais eficientes e mais completas do programa.

A figura 2 ilustra estas etapas e as possíveis interações durante o processo de desenvolvimento do programa.

Como se pode notar nesta figura, porém, a todas as atividades de síntese é associada uma atividade de análise, ou validação, que permite avaliar a evolução do protocolo de um nível a outro. A metodologia proposta neste trabalho situa-se na etapa de projeto e tem por objetivo permitir a validação das especificações formais de protocolo.

3.2. Os diferentes níveis de especificação de um protocolo

Durante a trajetória de concepção de um programa distribuído, vários níveis de especificação podem ser gerados, como foi mostrado na figura 2.

Partindo de uma análise de requisitos, expressa ou não de maneira informal, um primeiro nível de especificação seria o da *especificação funcional*, que permite, principalmente, determinar a organização do software em termos de unidades executáveis e das relações de comunicação e hierárquicas entre estas unidades.

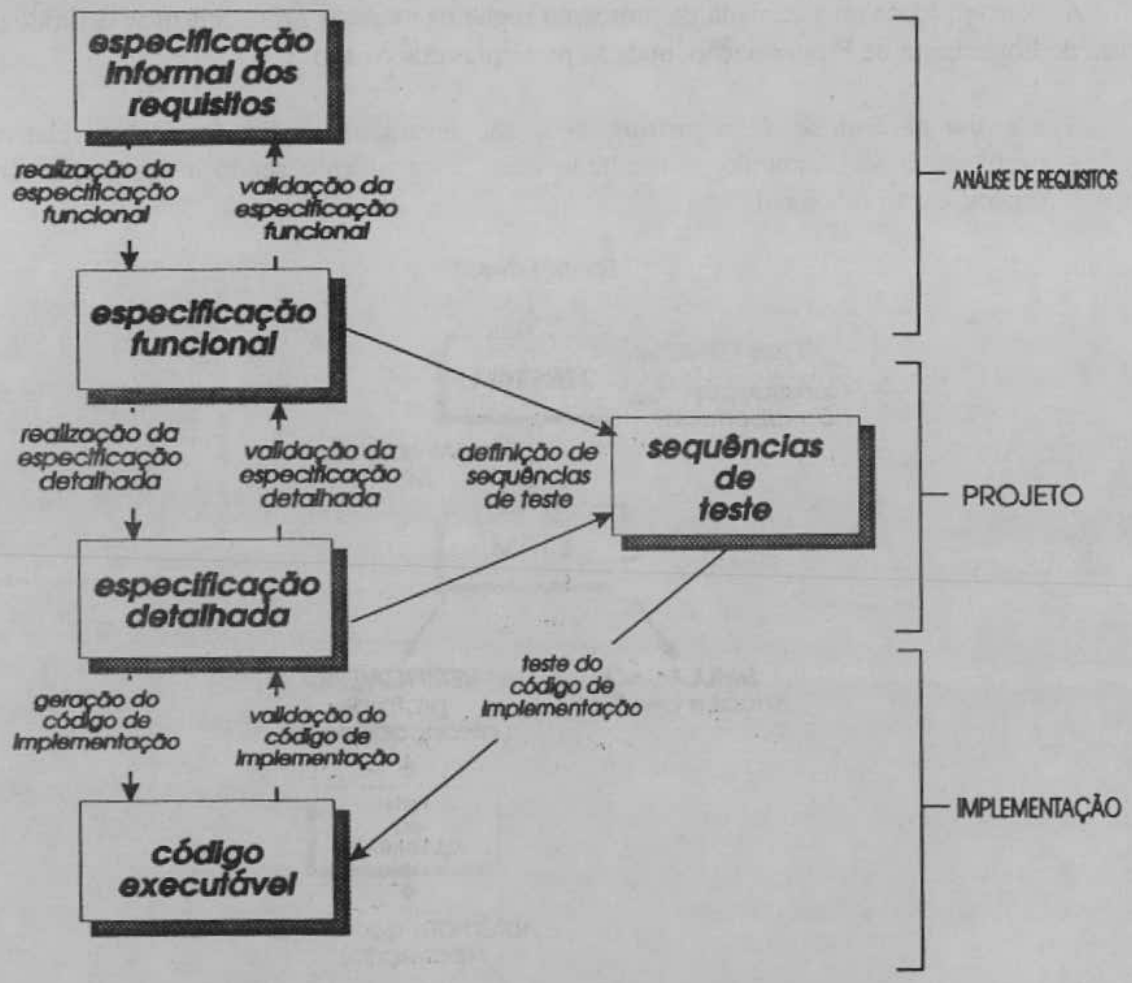

Figura 2 - As atividades de desenvolvimento do software.

No que diz respeito à utilização de Estelle para realizar uma tal especificação, os mecanismos a serem explorados aqui são, principalmente, construções para a declaração de objetos Estelle, tais como os canais e interações (estas, eventualmente, sem parâmetros), os tipos e corpos de módulo e as variáveis módulo.

No caso dos corpos de módulo, nenhuma declaração de estados ou de comportamento é apresentada, limitando-se, neste caso, às declarações de outros objetos Estelle, importantes no contexto do módulo considerado. A parte inicialização da especificação e de alguns módulos pode ainda ser definida, permitindo definir as instâncias de módulo e as possíveis ligações entre estes módulos. Ainda, dependendo das características do sistema em desenvolvimento, pode-se associar os atributos permitindo definir os modos de paralelismo da especificação.

Caminhando na direção de uma implementação, podemos definir um segundo nível de especificação formal, denominada *especificação orientada modelo*, onde os primeiros aspectos comportamentais do sistema são apresentados. As unidades executáveis, definidas na especificação funcional, ganham vida, através de uma representação onde o não determinismo é explorado no seu máximo como forma de obtenção de um alto grau de abstração no que diz respeito a determinados mecanismos a serem implementados. Em particular aqueles que não estejam diretamente associados com o paralelismo e a comunicação entre unidades executáveis da especificação.

Do ponto de vista de Estelle, a especificação é enriquecida dos aspectos de representação de comportamento, inserindo-se a descrição das máquinas de estado regendo o comportamento de cada corpo de módulo declarado na especificação funcional. No que diz respeito à comunicação, cláusula e instrução de comunicação por fila FIFO (respectivamente, *when* e *output*) ou ainda as cláusulas de sincronização (*ip!interação* e *ip?interação*) de Estelle* são inseridas de modo a caracterizar os eventos associados à comunicação entre os módulos. A cláusula *delay*, de temporização, é outro elemento importante neste nível da especificação. Eventualmente, objetos Pascal como tipos e variáveis, funções e procedimentos, devem ser evitados a este nível, a menos que sejam indispensáveis à representação do comportamento do sistema. A especificação deve aproximar-se ao máximo de uma modelização orientada sistema de transição, o que justifica a sua denominação.

Num terceiro nível, encontramos a *especificação detalhada*, na qual o alto grau de não determinismo cede lugar à representação, a mais fiel possível dos mecanismos omitidos na especificação anterior. Neste nível, os aspectos ligados à implementação de algoritmos sequenciais de manipulação de dados são representados, se possível, na sua totalidade.

Neste nível de especificação, tipos e variáveis Pascal, funções e procedimentos e parâmetros de interações devem ser inseridos de modo a dar um caractere o mais realista possível no que diz respeito aos mecanismos a serem implementados no sistema.

Finalmente, num último grau de especificação formal, encontramos a *especificação orientada implementação*, caracterizada pela supressão de alguns aspectos das especificações anteriores (mesmo algumas unidades executáveis) e pela introdução de aspectos relacionados ao ambiente no qual o programa deverá executar. A partir desta especificação, ferramentas automatizadas permitirão gerar o código executável e o ambiente de execução do programa considerado.

Os níveis de especificação até aqui apresentados são marcados por uma característica comum — a descrição, de maneira abstrata, além dos módulos relativos ao sistema a implementar, de módulos representando o ambiente no qual o sistema vai executar. No caso de uma camada N de protocolo, por exemplo, são representados, além das entidades de protocolo N, as entidades usuárias do serviço N (as entidades N+1) e o serviço de comunicação utilizado (o serviço N-1).

Desta forma, as alterações realizadas na especificação detalhada no sentido de obter-se a especificação orientada implementação são, geralmente:

- supressão de todos os módulos representando o comportamento do ambiente (uma vez que estes serão supostos existir no ambiente onde o sistema vai executar);

- substituição, das comunicações dos módulos restantes com os que foram retirados, por chamadas a primitivas de comunicação de biblioteca ou do próprio sistema operacional utilizado para a implementação;
- eventual separação de alguns módulos, antes numa mesma especificação Estelle, em várias especificações (vários sistemas).

A partir daí, ferramentas orientadas implementação permitem gerar o código executável, integrando-o a um ambiente de execução sob o sistema operacional escolhido para a implementação.

3.3. A verificação por abstração

A metodologia de validação proposta é orientada a especificações formais de protocolos de comunicação em Estelle, mais particularmente, Estelle*, utilizando uma técnica de verificação por abstração. A verificação por abstração é baseada na construção, a partir da especificação formal, de um grafo descrevendo todos os eventos (no caso, transições) podendo ocorrer no protocolo representado, este grafo sendo conhecido por *grafo de alcançabilidade*.

A análise do grafo de alcançabilidade permite tirar conclusões importantes a respeito do comportamento das entidades de protocolo em desenvolvimento, uma vez que todas as possibilidades de execução estão ali representadas. Evidentemente, como toda técnica de verificação, para especificações de porte realista, os grafos de alcançabilidade obtidos podem ter uma dimensão bastante grande, atingindo milhares ou milhões de estados possíveis. A análise de um grafo desta dimensão não é impossível, naturalmente, mas é, sem dúvida, uma tarefa de difícil implementação. Além disto, o grafo representa, em boa parte dos casos, uma grande quantidade de eventos cuja importância pode ser posta em questão para a tarefa de análise.

A verificação por abstração é baseada na definição de duas classes de eventos associadas ao comportamento de um processo:

- os *eventos externos* ou *observáveis*, caracterizando normalmente as interações do processo com o seu ambiente (usuários da camada de protocolo, o suporte de transmissão, etc);
- os *eventos internos*, caracterizando ou os eventos expontâneos dos processos, ou as interações entre componentes do processo (com relação ao exemplo apresentado anteriormente, as interações realizadas na porta *c* são consideradas eventos internos — na composição paralela, eles foram etiquetados por τ).

A diferença entre estas duas classes de evento se traduz a nível do autômato quociente. Os eventos observáveis aparecendo no grafo de alcaçabilidade são, normalmente repassados ao autômato quociente, representados na forma de transições etiquetadas. Já os eventos internos podem, ou desaparecer, ou aparecer na forma de transições (etiquetados por τ) no autômato quociente.

Diferentes relações de equivalência podem ser consideradas, onde cada relação vai permitir obter uma visão distinta do comportamento do sistema a verificar, entre elas: a equivalência de rastro (ou de linguagem), a equivalência observacional e a equivalência de teste [15].

Sendo assim, dependendo das características da especificação e da relação de equivalência utilizada para a obtenção do autômato quociente, este pode representar uma redução significativa do grafo de alcançabilidade original, mas guardando, ainda assim, informações suficientes para a análise do protocolo.

3.4. A escolha de uma partição

A verificação por abstração baseada em Estelle* e implementada no ambiente ESTIM é baseada ainda na definição de uma partição, que vai permitir distinguir os eventos observáveis dos eventos internos, como definido acima. Desta forma, a escolha da partição vai permitir determinar,

de um lado o nível de importância dos eventos presentes na especificação, e do outro, o grau de redução do grafo de alcançabilidade, uma vez que todas as relações de equivalência levam esta distinção em conta.

A figura 3 ilustra o conceito de partição e sua importância na distinção do que são eventos observáveis e eventos invisíveis. No caso da figura, a especificação Estelle é composta das instâncias de módulo *UA*, *UB*, *A* e *B*, sendo que a partição foi definida de tal forma que as instâncias de módulo *UA* e *UB* fazem parte do chamado *ambiente* e as instâncias de módulo *A* e *B* são parte do *mundo observado*.

No que diz respeito à distinção entre os dois tipos de eventos, a partição determina o seguinte:

- todas as transições envolvendo operações de comunicação entre módulos do ambiente e do mundo observado serão considerados eventos observáveis e vão aparecer etiquetados no momento da construção do grafo de alcançabilidade;
- as transições expontâneas de cada módulo ou aquelas associadas a operações de comunicação entre módulos do mundo observado serão consideradas eventos internos ou invisíveis e serão etiquetados por τ, no grafo de alcançabilidade.

Para a figura 3, isto significa que todas as transições envolvendo a comunicação entre os módulos *UA* e *A* e os módulos *UB* e *B* serão eventos observáveis, as demais transições sendo, então, eventos internos.

4. EXEMPLO DE UTILIZAÇÃO DA METODOLOGIA PROPOSTA

Para ilustração da metodologia proposta, escolhemos a norma MMS pela sua importância atual na instalação de sistemas integrados de manufatura. As seções que seguem apresentarão, sequencialmente, um breve histórico da norma MMS, os objetos e serviços definidos na norma, e a aplicação da metodologia proposta para a análise das máquinas de protocolo MMS.

4.1. Histórico de MMS

Os sistemas de produção foram durante algum tempo baseados em arquiteturas e protocolos proprietários. As primeiras redes ofereciam facilidades de comunicação fortemente ligadas ao tipo de sistema.

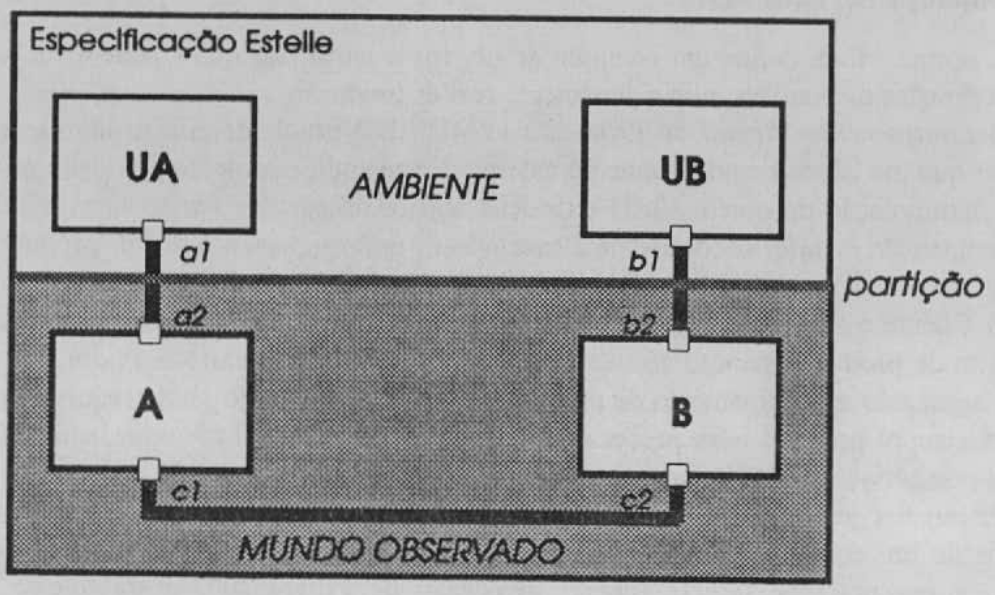

Figura 3 - Técnica de verificação por abstração baseada em Estelle: ilustração do conceito de partição.

As aplicações desenvolvidas sobre estas redes deveriam ser adaptadas às características dos sistemas interconectados, o que dificultava a manutenção se um sistema era substituído. As redes industriais foram concebidas para reduzir este problema. Elas ofereciam um conjunto de protocolos adaptados a todos os sistemas de um determinado fabricante, mas fornecendo facilidades para a interconexão de outros sistemas. As aplicações desenvolvidas sobre estas redes eram já menos dependentes dos sistemas, mas a interconexão de sistemas de diferentes fabricantes (em termos de software e hardware) poderia envolver um custo elevado.

O grupo de trabalho criado pela General Motors em 1981 tinha por objetivo a criação de uma rede local industrial para resolver os problemas de interconexão dos diferentes sistemas da companhia. O resultado deste trabalho, baseado no Modelo de Referência da ISO (OSI) foi a arquitetura MAP (Manufacturing Automation Protocol).

SMF (Standard Message Standard) foi o primeiro resultado obtido dos trabalhos relacionados à arquitetura MAP. SMF oferecia um vocabulário capaz de permitir leitura, escritura e carregamento remoto de programas em controladores programáveis. Em 1984, em decorrência dos trabalhos iniciados sobre SMF, um grupo de trabalho da EIA (Electronic Industries Association), definiu MMFS (Manufacturing Message Format Standard ou "Memphis"), orientado à interconexão de sistemas do tipo máquinas de controle numérico, robôs e controladores programáveis. MMFS foi especificado compondo a versão 2.1 de MAP. A especificação de MMFS definia um grande número de serviços associados à interconexão destes equipamentos, mas apresentava duas deficiências: em primeiro lugar, o método de codificação adotado era incompatível com aquele utilizado por outros protocolos da camada 7 do modelo OSI; em segundo lugar, a especificação deixava grande parte da interpretação a cargo do programador, o que poderia resultar na geração de implementações incompatíveis.

Em 1985, a EIA promove uma re-especificação de MMFS, tendo por principal objetivo a aplicação das regras de codificação de PDUs definidas pela ISO, ou seja a notação de sintaxe abstrata ASN.1 [16], [17]. A primeira proposição de MMS (RS-511) foi apresentada em junho de 1985, num documento composto de duas partes: Manufacturing Message Service: Definição de Serviços e Manufacturing Message Service: Especificação do Protocolo.

Atualmente, MMS [18], [19], é o protocolo de aplicação para a comunicação de sistemas industriais na versão 3.0 da arquitetura MAP, de agôsto de 1988. Subconjuntos dos serviços definidos por MMS foram adotados por outras arquiteturas orientadas à communicação em ambiente industrial.

4.2. Objetos e Serviços MMS

A norma MMS define um conjunto de objetos a partir dos quais podem ser acessados os recursos e funções disponíveis num equipamento real de produção.

O *Equipamento Virtual de Produção* (VMD, de Virtual Manufacturing Device) consiste num objeto que modeliza o comportamento externo de um equipamento de produção no contêxto de MMS. A manipulação do objeto VMD e de seus objetos associados via serviços MMS permite a gestão e a obtenção de informações sobre a execução de um equipamento real de produção.

A comunicação no contexto MMS ocorre numa relação entre usuários (ou processos de aplicação) Cliente e Servidor. O usuário Cliente é aquele que requer o acesso aos recursos de um equipamento de produção remoto através dos serviços MMS. O usuário Servidor é o processo de aplicação associado ao equipamento de produção cujos recursos estão sendo requisitados. O objeto VMD está sempre presente num processo de aplicação Servidor MMS, representando e tornando disponível (via serviços MMS) os recursos do equipamento real de produção associado.

Os objetos gerados pelo Equipamento Virtual de Produção permitem o acesso aos recursos disponíveis de um equipamento de produção para: a transferência de dados e programas (objeto Domínio), a execução de tarefas (objeto Invocação de Programas), o tratamento de eventos (Condição de Evento, Ação de Evento, Envelope de Evento), a definição de variáveis (Variável Anônima, Variável Nomeada, Lista de Variáveis, ...), a exclusão mútua a recursos e a sincronização

de processos (objeto Semáforo), a interface operador (objeto Estação Operador) e a geração de relatórios de operação (objeto Jornal).

A norma MMS [18] define um número de 84 serviços divididos em 10 classes: Gestão de Contexto (estabelecimento e gestão do diálogo, cancelamento de um serviço, sinalização de erros de protocolo), Gestão de VMD (obtenção de estado, identificação do equipamento, obtenção da lista de objetos associados), Gestão de Domínio (transferência de dados e programas - "download" e "upload"), Gestão de Variáveis (criação e destruição dinâmica de tipos e variáveis, leitura e escritura remotas), Gestão de Programas (criação e destruição dinâmicas de invocações de programas, controle de execução de programas - partida, parada, retomada de execução), Gestão de Semáforos (alocação de recursos, sincronização de tarefas), Gestão de Eventos (detecção, sinalização e tratamento de eventos), Gestão de Jornal (leitura, escritura e armazenagem de jornais), Comunicação com Operador (entrada e saída de informação) e Gestão de Arquivos (leitura, escritura, renomagem e diretório).

Um usuário MMS não deve suportar todos os serviços definidos na norma, mas somente aqueles necessários à sua comunicação com outros processos presentes no contêxto de uma aplicação. A norma MMS apresenta de forma bastante genérica a utilização dos serviços MMS para a interconexão dos diferentes sistemas de produção buscando abranger as diversas classes de equipamentos e as suas possíveis utilizações. O programador de uma aplicação pode orientar-se de maneira mais precisa graças à utilização de outros documentos que instanciam a utilização dos serviços objetos a uma classe bem definida de equipamento. Estes documentos, chamados *Companion Standards* (ou *Normas de Acompanhamento*), foram definidos para as classes Robôs [20], Máquinas à Comando Numérico [21] e Controladores Lógicos Programáveis [22].

4.3. Análise do protocolo MMS

Num Elemento de Serviço MMS, uma máquina de protocolo é criada para cada serviço requisitado (ou efetuado) por um fornecedor MMS requisitante (ou respondedor). A máquina de protocolo MMS (ou MMPM, de Manufacturing Message Protocol Machine) permite concretizar os mecanismos previstos no protocolo MMS. A operação das máquinas de protocolo é mostrada na figura 4, levando em conta o requisitante e o fornecedor de um serviço confirmado.

As interações apresentadas na figura correspondem às trocas de mensagem entre a máquina de protocolo e os usuários MMS. O símbolo *?* indica a recepção de uma mensagem (ou primitiva de serviço) e o *!*, uma emissão de mensagem. As interações são compostas de um prefixo indicando o serviço a efetuar (*c* para o serviço *Cancel* e *x* para os demais serviços) e de um sufixo indicando a primitiva de serviço (*req* para *request*, *ind* para *indication*, *rsp* para *response* e *cnf* para *confirmation*). O símbolos + e - complementando os sufixos das primitivas de serviço de *response* e *confirmation* indicam, respectivamente o sucesso e a falha da execução de um serviço.
Para se poder efetuar uma análise exaustiva do protocolo dentro de uma ótica de verificação, é necessária uma redução no modelo. A representação de informações irrelevantes poderia provocar explosões combinatórias, o que tornaria mais dificil (ou impossível) a análise do modelo.

A figura 5 apresenta a arquitetura da especificação realizada para este estudo: os dois módulos superiores representam os usuários MMS Cliente e Servidor. No nosso modelo, foram especificadas todas as interações possíveis entre estes módulos e os fornecedores de serviço MMS. Para isso, os corpos dos módulos foram definidos de forma a permitir, a todo momento, a troca de primitivas de serviço com os módulos inferiores. Estes últimos implantam o protocolo MMS para o requisitante e respondedor como descrito nas máquinas de estado da figura 5.

Utilizando o ambiente ESTIM, a verificação é efetuada graças à capacidade de geração de um grafo de acessibilidade de uma especificação Estelle*. Este grafo, podendo apresentar um número bastante grande de estados e arcos associados, impossibilita uma análise precisa se realizada manualmente. Ele é então reduzido graças aos diferentes algoritmos de projeção implantados na ferramenta PIPN com a qual ESTIM é interfaceado.

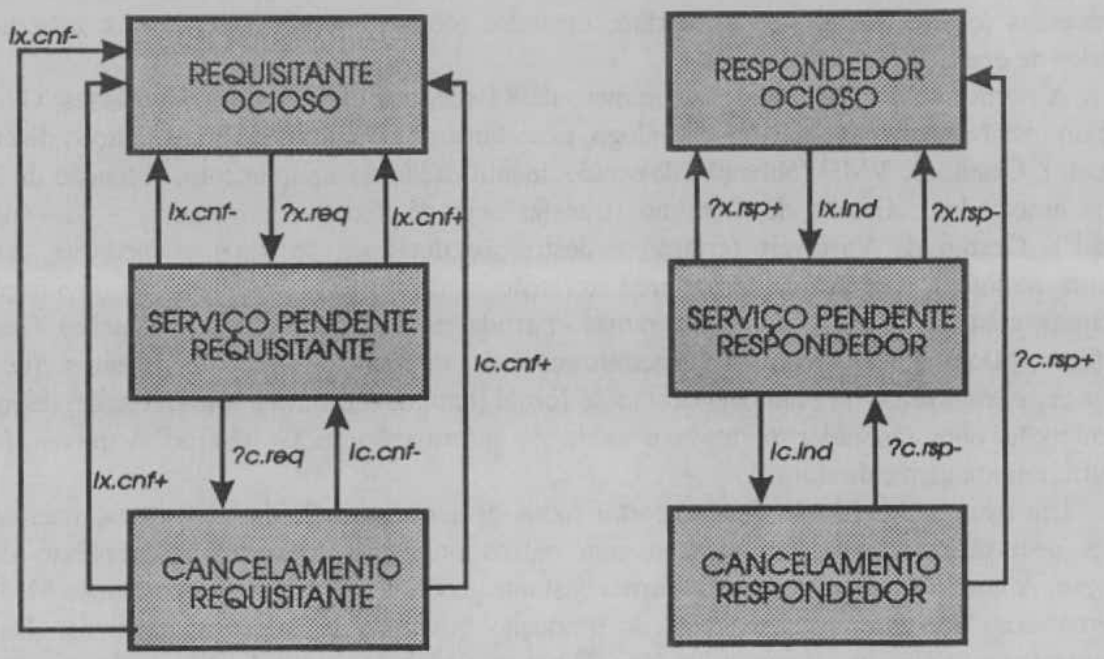

Figura 4 - Máquinas de Protocolo MMS Requisitante e Respondedor.

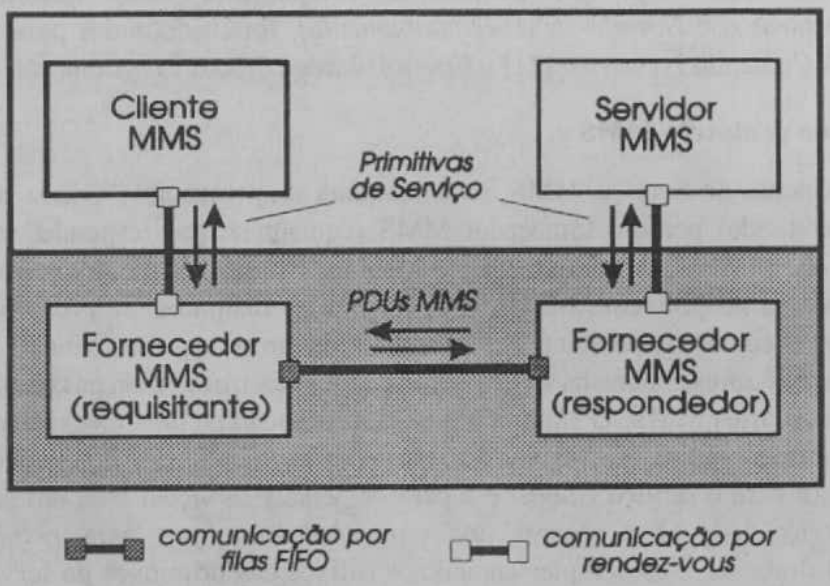

Figura 5 - Esquema de validação do protocolo MMS.

Define-se uma *partição* estabelecendo assim uma fronteira de observação a partir das primitivas de serviço intercambiadas entre os fornecedores do serviço de comunicação (que constituem o *mundo observado*) e os usuários MMS (constituindo o *ambiente*) — figura 5.

O autômato resultante da projeção do grafo de acessibilidade da especificação Estelle* é mostrado na figura 6. Pode-se observar um funcionamento anormal do protocolo, ilustrado pelos diversos caminhos que conduzem ao estado número 14 (por exemplo, o caminho 1→2→3→5→9→13→14), e que indica a existência de um bloqueio mortal (*deadlock*). Estes caminhos correspondem à situação na qual a primitiva de resposta (positiva ou negativa) de um serviço requisitado e a primitiva de pedido do serviço *Cancel* (para anulação do serviço) são enviadas simultaneamente pelo servidor e o cliente MMS, respectivamente.

Em nossa especificação, fazendo uma análise em simulação, foi observado que a diferença entre os estados 1 e 13 é provocada pela permanência da primitiva de pedido do serviço Cancel na fila FIFO do ponto de interação associado ao módulo "Fornecedor MMS (Respondedor)". Nenhuma transição sendo especificada para consumir esta primitiva a partir do estado "Respondedor Ocioso",

novas primitivas enviadas a este módulo (a primitiva *x.req* que aparece entre os estados 13 e 14, por exemplo) serão bloqueadas na fila FIFO.

A norma MMS propõe uma solução a este problema (Cláusula 6.4 de [19]), através do envio de uma resposta negativa, a partir do estado "Respondedor Ocioso", do serviço Cancel, se esta indica um serviço desconhecido. Apesar desta ser uma modificação necessária, ela resolve parcialmente o problema pois, o bloqueio se dará mais tarde do lado do Requisitante, uma vez que nenhuma transição de consumo da resposta ao serviço Cancel é especificada (a partir do estado "Requisitante Ocioso"). Isto significa que, do ponto de vista do autômato mostrado na figura 6, alguns estados seriam adicionados (a partir do estado 13), mas o bloqueio mortal continuaria a existir.

A solução definitiva ao problema está na introdução, em nossa especificação, de uma transição no fornecedor requisitante MMS, que permita a este consumir (a partir do estado "Requisitante Ocioso") a mensagem de confirmação negativa do serviço Cancel.

O autômato obtido por projeção para a nova especificação é mostrado na figura 7. Pode-se notar a eliminação do bloqueio mortal, caracterizado pelo aparecimento dos estados 14 e 20 a 24, este último sendo o estado do qual o protocolo retorna ao estado inicial.

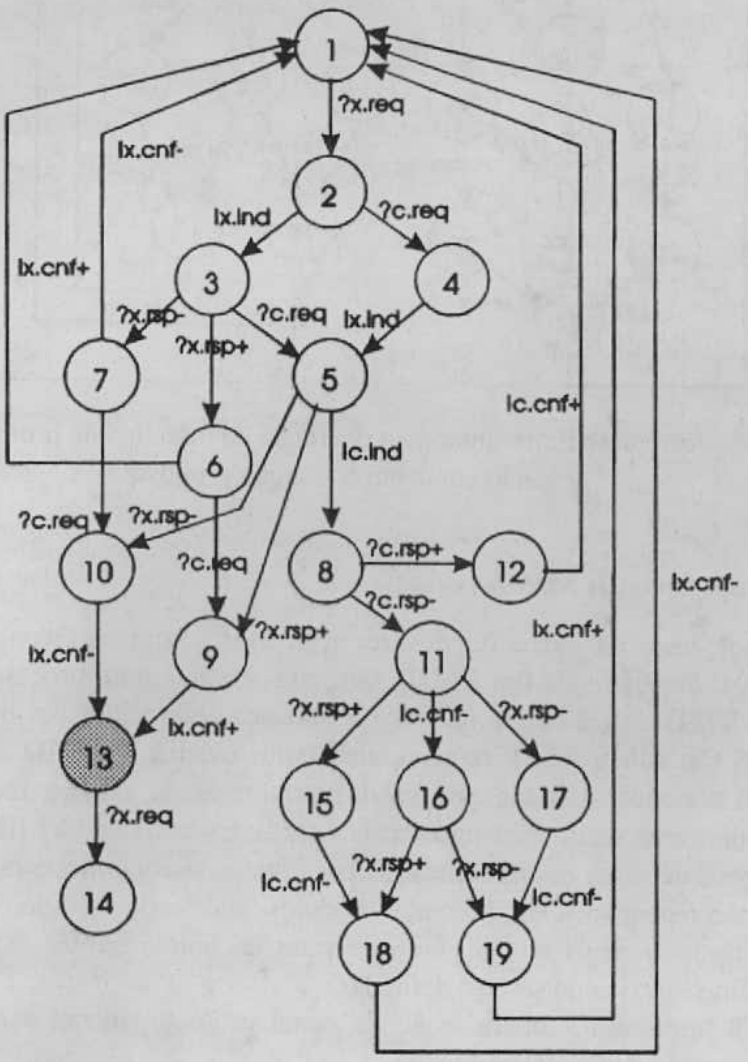

Figura 6 - Autômato representando a operação da máquina de protocolo MMS.

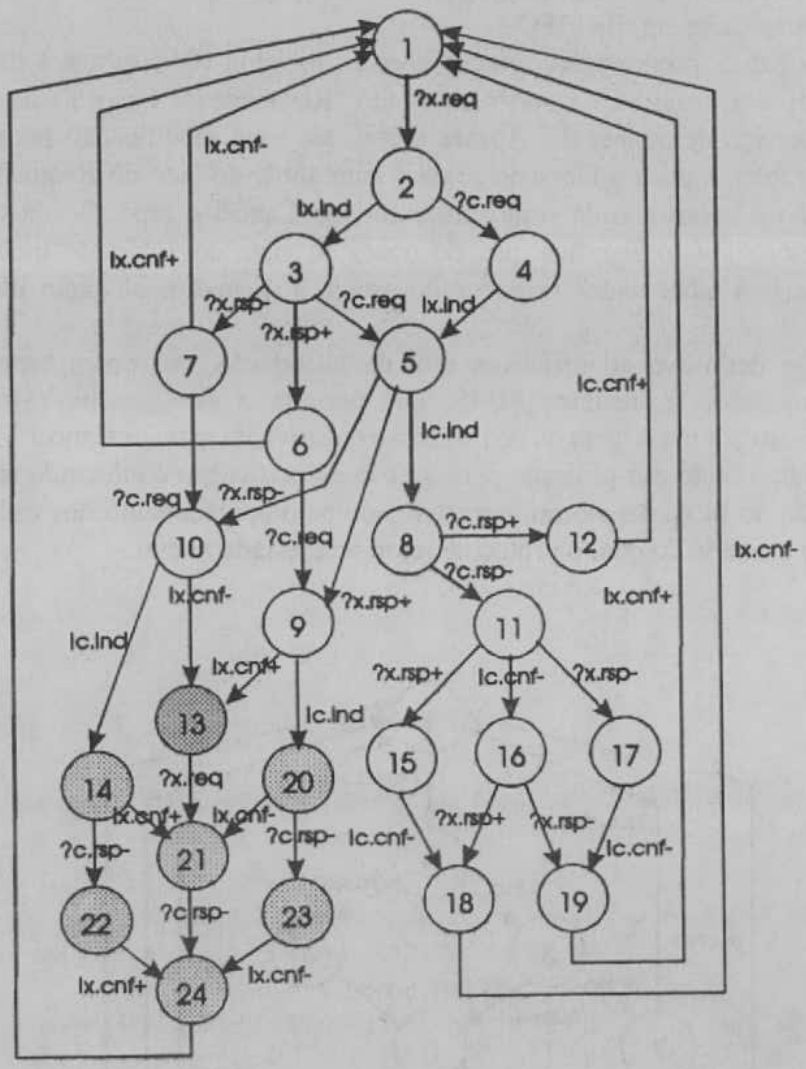

Figura 7 - Autômato representando a operação da máquina de protocolo MMS, levando em conta a solução proposta.

4.4. A análise dos serviços MMS

O objeto de base na execução dos serviços MMS num equipamento de produção é o Equipamento Virtual de Produção (ou VMD), que está presente num processo de aplicação de um servidor MMS. O VMD utiliza os serviços de comunicação oferecidos por um elemento de serviço de aplicação MMS (ou MMS-ASE). A manipulação dos objetos do VMD se dá via execução dos procedimentos dos serviços ativados através das primitivas de serviço recebidas do fornecedor MMS, estes procedimentos sendo executados pela Função Executiva do VMD.

De um ponto de vista de especificação em Estelle, o conjunto de serviços executados na função executiva são regrupados em diferentes módulos, onde cada módulo está associado à gestão de um ou vários tipos de objetos. Do ponto de vista da norma MMS, esta separação pode ser associada às diferentes classes de serviço definidas.

A figura 8 apresenta a nossa visão de como estão organizados os diferentes módulos, responsáveis da execução dos serviços MMS.

O módulo de interface é responsável da recepção e entrega das primitivas de indicação de serviço emitidos pela ASE-MMS, assim como do envio das primitivas de resposta aos serviços executados. Cada módulo é responsável da execução dos serviços associados à classe correspondente, recebendo as indicações (enviando respostas) do (para) o módulo de interface com a ASE-MMS.

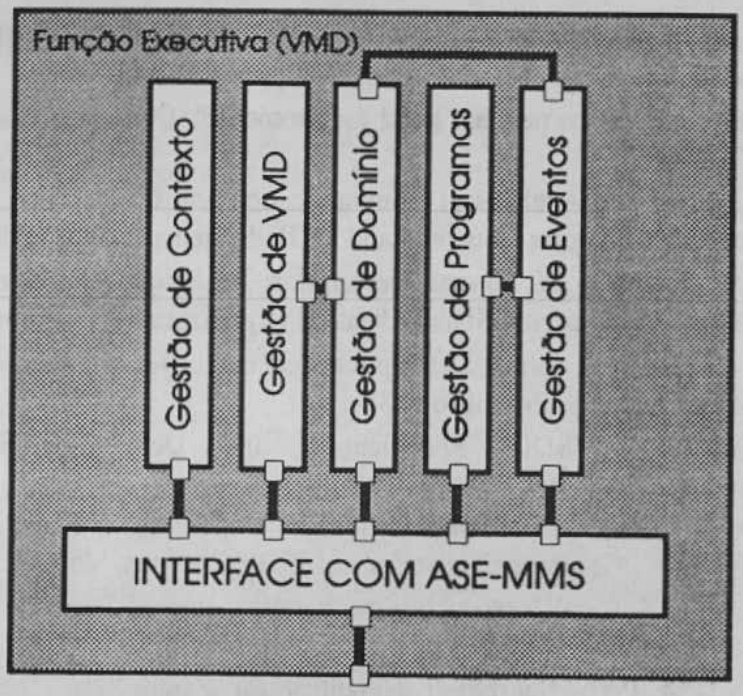

Figura 8 - Função executiva do Equipamento Virtual de Produção.

Nem todas as classes de serviços estão representadas na figura, mas isto não deve ser interpretado como uma restrição ao uso das classes não presentes.

Como pode ser notado, podem existir interações entre os diferentes módulos associados às classes de serviços. Isto é explicado pelo possível entrelaçamento entre serviços associados a diferentes classes. Alguns exemplos destes entrelaçamentos podem ser explicitados:

- notificação de evento associada ao final de execução de um programa;
- utilização de modificadores (*AttachToSemaphore* ou *AttachToEventCondition*), que permite associar a execução de um serviço à possessão de uma ficha de um semáforo ou à ocorrência de um evento;
- utilização do serviço Cancel.

Os módulos compondo a Função Executiva foram especificados em Estelle e deverão ser validados seguindo, inicialmente, uma ótica de simulação. A simulação deverá permitir validar os módulos um a um e, em seguida, em grupos, o que vai permitir obter uma análise mais completa no que se refere aos entrelaçamentos entre serviços.

5. CONCLUSÕES

Este artigo apresentou uma proposta de metodologia de validação de protocolos das camadas altas do modelo OSI, utilizando a técnica de descrição formal Estelle. A ilustração da metodologia através de um exemplo significativo e importante como MMS demonstra a sua aplicabilidade na Engenharia de Protocolos.

Por outro lado, como toda técnica baseada em verificação, a metodologia apresenta suas limitações, principalmente para o caso de especificações formais de grande complexidade.

Isto ressalta a importância da realização de diferentes níveis de especificação, sendo que o nível mais adequado para a aplicação desta metodologia seria a *especificação orientada modelo*, apresentada na seção 3.

No que diz respeito às perspectivas de continuidade deste trabalho, as propostas se direcionam para a consolidação da metodologia através de sua aplicação a outros casos, seja na validação de serviços MMS, seja na análise de outros exemplos de protocolos.

6. REFERÊNCIAS

[1] A. S. Danthine. "Protocol Representation with Finite-State Models", IEEE Transactions on Communications, Vol. COM-28, nº 4, Abril 1980, pp. 632-643.

[2] J. P. Courtiat. et alli. "Petri nets are good for protocols", Computer Communication Review, Vol. 14, nº 2, 1984.

[3] M. Diaz. "Modeling and Analysis of Communication and Cooperation Protocols using Petri Net based Models", Computer Networks and ISDN Systems, Vol. 6, 1982, pp. 419-441.

[4] ISO IS 9074. "Estelle - A Formal Description Technique Based on a Extended State Transition Model". International Standardization Organization, Novembro 1988.

[5] ISO IS 8807. "LOTOS, A formal Description Technique Based on the Temporal Ordering of Observational Behaviour", Novembro 1988.

[6] CCITT/SGXI/WP3-1. "SDL, Specification and Description Language", CCITT Recommendations Z100-Z104, 1988.

[7] S. Budkowski; P. Dembinski. "An Introduction to Estelle: A Specification Language for Distributed Systems", Computer Networks and ISDN Systems, Vol. 14, 1987, pp. 3-23.

[8] R. L. Tenney. "A Tutorial Introduction to Estelle", Invited Paper at the 1st International Conference on Formal Description Techniques, Stirling, Setembro 1988.

[9] J. P. Courtiat. "A Petri Net based semantics for Estelle", in The Formal Description Technique Estelle, North-Holland, 1989.

[10] M. Diaz.; C. Vissers. "SEDOS: Designing Open Distributed Systems", IEEE Software, pp. 24-33, Novembro 1989.

[11] J. P. Courtiat. "Estelle*: a Powerful Dialect of Estelle for OSI Protocol Description", Proceedings of the 8th IFIP Symposium on Protocol Specification, Testing and Verification, Atlantic City, Junho 1988.

[12] J. P. Courtiat. "Contribution à la Description Formelle de Protocoles", Thèse de Docteur d'Etat de l'Université Paul-Sabatier, LAAS - Toulouse , Dezembro 1987.

[13] P. de Saqui-Sannes, J. P. Courtiat. "From the Simulation to the Verification of Estelle Specifications", 2nd International Conference on Formal Description Techniques, Vancouver, Dezembro 1989.

[14] V.B. Mazzola et alli. "Aplicação da Técnica de Descrição Formal Estelle à concepção do Sistema de Controle de uma Célula Flexível de Usinagem", 9º Congresso Brasileiro de Automática (CBA'92), Setembro de 1992, Vitória - ES.

[15] P. Ernberg et alli. "Guidelines for Specification and Verification of Communication Protocols", SICS Perspective, Report nº 1, Swedish Institute of Computer Science, 1991.

[16] ISO 8824 "Specification of Abstract Syntax Notation One [ASN.1]". International Organization for Standardization, Agosto 1987.

[17] ISO 8825 "Specification of Basing Encoding Rules for Abstract Syntax Notation One [ASN.1]". International Organization for Standardization, Agosto 1987.

[18] ISO DIS 9056: Manufacturing Message Services/Part 1 - Service Definition, Agosto 1988.

[19] ISO DIS 9056: Manufacturing Message Services/Part 2 - Protocol Specification, Agosto 1988.

[20] ISO DP 9056: Manufacturing Message Services/Part 3 - Robots for Manufacturing - Companion Standard, Julho 1989.

[21] ISO DP 9056: Manufacturing Message Services/Part 4 - Numerical Control of Machines - Companion Standard, Abril 1989.

[22] IEC/SC 65A/WG 6/TF 7: Programmable Controller Message Specification, Junho 1989.