UM MODELO DE INTERFACE DE APLICAÇÃO PARA AMBIENTES DE AUTOMAÇÃO INDUSTRIAL

AN APPLICATION INTERFACE MODEL FOR INDUSTRIAL AUTOMATION ENVIRONMENTS

Prof. Dr. Edmundo Roberto Mauro MADEIRA*
Prof. Dr. Manuel de Jesus MENDES**

ABSTRACT

This paper proposes a General Model of API ("Application Program Interface") for industrial networks with new functionalities added to the API from MAP ("Manufacturing Automation Protocol"). An implementation based on this Model for the MMS ("Manufacturing Message Specification") Protocol developed at UNICAMP is presented and some implementation issues are alanyzed.

KEY-WORDS: Computer Networks, Application Protocols, MAP Project, MMS Protocol, Application Interface.

RESUMO

Este artigo propõe um Modelo Geral de API ("Application Program Interface") para redes industriais com novas funcionalidades adicionadas à API do Projeto MAP "Manufacturing Automation Protocol"). Uma implementação baseada neste Modelo Geral para o Protocolo MMS ("Manufacturing Message Specification") desenvolvida na UNICAMP é apresentada e algumas questões de implementação são analisadas.

PALAVRAS-CHAVE: Redes de computadores, Protocolo de Aplicação, Projeto MAP, Protocolo MMS, Interface de Aplicação.

1 - INTRODUÇÃO

1.1 - Projeto MAP

O Projeto MAP foi criado pela General Motors no início da década de 80, com o objetivo de definir mecanismos e procedimentos de interconexão de equipamentos programáveis para a automação industrial. A meta era facilitar a interconexão das "ilhas de automação" existentes, para construir sistemas integrados da manufatura [MEN89]. Um outro Projeto com objetivos semelhantes aos do MAP, mas orientado para a automação de escritórios, foi criado pela Boeing no mesmo período e denominado TOP ("Technical and Office Protocol"). Uma promoção conjunta da General Motors com a Boeing na AUTOFACT em 1985, consistindo na exposição dos Projetos MAP e TOP, uniu os dois projetos num só, denominado MAP/TOP.

O Projeto MAP/TOP segue o Modelo de Referência OSI/ISO ("Open Systems Interconnection/International Organization for Standardization") e define os protocolos a serem adotados para cada uma das sete camadas do Modelo. Para a camada de aplicação, que provê serviços específicos orientados às aplicações, o Projeto MAP define o uso dos seguintes protocolos:

a - MMS ("Manufacturing Message Specification"), que suporta a transmissão de mensagens entre equipamentos programáveis, num ambiente de manufatura e controle de processos:

b - FTAM ("File Transfer Access and Management"), que provê serviços de gerenciamento e transferência de arquivos e de registros de dados;

c - DS ("Directory Service"), que provê serviços de administração de endereços e atribuição de nomes dos objetos OSI das diversas estações da rede;

d - NM ("Network Management"), que provê serviços de gerenciamento de rede relacionados com configuração, desempenho, falhas, contabilidade e segurança.

(*) Professor do Departamento de Ciências da Computação - IMECC/UNICAMP
(**) Professor da Faculdade de Engenharia Elétrica - UNICAMP/Porfessor do Instituto de Informática - PUCCAMP

O Projeto TOP define para a camada de aplicação os protocolos FTAM, DS, NM, VT ("Virtual Terminal") e MHS ("Message Handling System"). Os dois últimos providenciam os serviços de "login remoto" e troca de mensagens eletrônicas, respectivamente.

1.2 - Protocolo MMS

O Protocolo MMS [ISO9506] é o componente principal do Projeto MAP. Os equipamentos da manufatura são, em geral, simples, mas altamente especializados. Por exemplo, numa célula da manufatura, uma estação supervisora (PC) controla diversos equipamentos tais como CNR, CNC e CLP entre outros. Esses equipamentos podem ser interconectados numa rede local. Usando o MMS, os equipamentos da célula comunicação de uma maneira apropriada e padronizada. Uma estação da célula para comunicar com outros computadores da fábrica usará protocolos mais complexos, como o FTAM.

O Protocolo MMS é visto como um modelo "cliente/servidor", que aceita diversas requisições de serviços complexos do cliente para serem executadas em estações servidoras simples. O MMS modela os equipamentos programáveis (servidores) através de VMDs ("Virtual Manufacturing Devices"). Os VMDs tratam, de uma forma abstrata, dos elementos estruturais e dos objetos que compõem os dispositivos reais. Cada VMD contém pelo menos um domínio. Um domínio representa um subconjunto de recursos alocados de forma estática ou dinâmica (por exemplo, memória, entrada/saída e funções específicas de controle entre outros). Um VMD, além de um ou mais domínios, possui uma função executiva, estações de operador e uma memória de arquivos virtuais. A função executiva administra o acesso aos recursos do VMD. O MMS somente descreve o comportamento da servidora. Nesta, é necessário providenciar o mapeamento do VMD para os aspectos funcionais do Dispositivo Real da Manufatura.

Os serviços do MMS são agrupados nas seguintes unidades funcionais: Gerenciamento de Contexto, Suporte de VMD, Gerenciamento de Domínio, Gerenciamento de Invocação de Programa, Acesso a Variáveis Remotas, Gerenciamento de Semáforo, Comunicação de Operador, Gerenciamento de Evento, Gerenciamento de Jornal e Gerenciamento de Arquivos.

1.3 - Interface de Aplicação

Os protocolos de aplicação normalmente oferecem serviços complexos, apesar do fato de poderem ser modelados por máquinas de estados relativamente simples. Esse fato contrasta com o que geralmente ocorre nos protocolos das camadas mais inferiores do RM-OSI/ISO, onde máquinas de estados mais complexas modelam serviços comparativamente mais simples. A complexidade dos serviços de aplicação ocorre por serem estes em grande número e por seus parâmetros poderem ser de tipos estruturados. Os Protocolos MMS e FTAM são exemplos típicos.

Portanto, a camada de aplicação pode não oferecer seus serviços aos APs ("Application Processes") de uma maneira simples. Por outro lado, o AP normalmente deseja que os serviços dos sistemas de comunicação sejam oferecidos de uma maneira simples ("friendly"). Efetivamente, existe uma lacuna ("gap") a ser diminuída.

Uma solução possível é introduzir no sistema de comunicação uma interface entre a camada de aplicação e o AP para suavizar esta lacuna existente [FON89]. O Projeto MAP/TOP chama esta interface de API ("Application Program Interface") [MAP88a], enquanto que [MAD90b] chama de AI ("Application Interface") (Figura 1.1). A API é definida para os protocolos MMS [MAP88b], FTAM e comunicação privada, enquanto que a AI é uma generalização da API para os protocolos de aplicação existentes atualmente. A lacuna comentada pode ser reduzida a partir de adições de novas funcionalidades à AI.

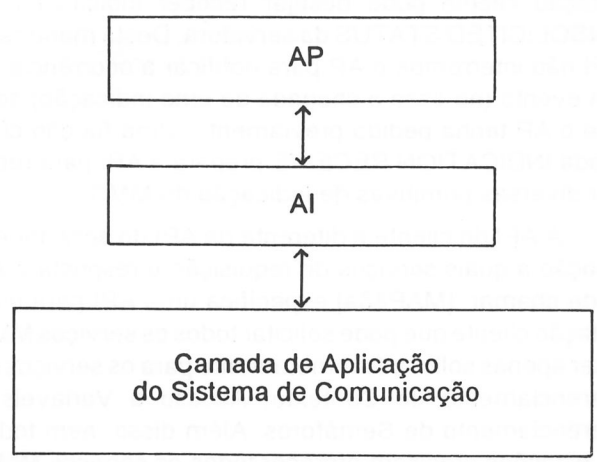

Figura 1.1 - Interface de Aplicação

O IEEE, através do Projeto POSIX ("Portable Operating System Interface") e de outros Projetos, está desenvolvendo APIs para alguns protocolos tais como FTAM, DS e MHS entre outros, para obter portabilidade de aplicações a nível de código fonte. Espera-se que estas padronizações do IEEE para APIs estejam concluídas no prazo de um a três anos [FLE92, HAZ92, NEW92].

A API é um conjunto explícito de funções e, portanto, provê portabilidade aos APs para diferentes ambientes através da padronização destas funções. A API

decrementa também o custo de programação e treinamento, pois os programadores podem usar a interface em ambientes múltiplos. A API executa algumas tarefas que seriam normalmente de responsabilidade do usuário, tais como: verificar se alguns valores de parâmetros das funções (serviços oferecidos) pertencem ao intervalo de valores válidos, preencher parâmetros "defaults" e dividir serviços complexos em serviços mais simples, entre outras.

A API para o Protocolo MMS é uma interface amigável e de alto nível para o usuário, oferecendo ao AP:

- Funções de baixo nível: somente uma primitiva de serviço MMS é invocada para cada chamada de função. READ e WRITE são exemplos de funções deste tipo para leitura e escrita de variáveis remotas, respectivamente:

- Funções de alto nível: diversas primitivas de serviços MMS são invocadas para uma chamada de função. FileGet é um exemplo de função deste tipo pois esta invoca três primitivas; fileOpen request, fileRead request e fileClose request;

- Funções respondedoras: a chamada de função prepara a API para receber primitivas de requisições remotas (denominadas indicações). Por exemplo, uma estação cliente pode desejar receber indicações de UNSOLICITED STATUS da servidora. Desta maneira, a API não interrompe o AP para notificar a ocorrência de um evento (no caso a chegada de uma indicação) sem que o AP tenha pedido previamente. Uma função chamada INDICATION RECEIVE prepara a API para receber diversas primitivas de indicação do MMS.

A API do cliente é diferente da API do servidor em relação a quais serviços de requisição e resposta o AP pode chamar. [MAP88a] específica uma API para uma estação cliente que pode solicitar todos os serviços MMS e ser apenas solicitada remotamente para os serviços de Gerenciamento de Contexto, Acesso a Variáveis e Gerenciamento de Semáforos. Além disso, nem todas as estações precisam oferecer todos os serviços MMS. Portanto, a API deve ser construída de maneira modular para permitir que apenas subconjuntos de funções sejam implementadas nas diversas estações, de acordo com as características próprias de cada estação.

2 - MODELO GERAL PARA INTERFACES DE APLICAÇÃO

Neste capítulo um Modelo Geral de Interface de Aplicação (AI), a partir da definição de API do Projeto MAP, será construído. A API do Projeto MAP é composta de três partes (Figura 2.1):

- Library (LIB): contém as funções que são utilizadas pelo AP para requisitar os serviços de rede desejados;

- Primitive Service Provider (PSP): é responsável pelo envio e recepção de primitivas para/da camada de aplicação;

- Três Procedimentos de Controle denominados:

- High Level Service Provider (HLSP): trata das funções de alto nível. Uma função de alto nível invoca diversas primitivas da camada de aplicação;

- Confirmed Service Provider (CSP): trata dos serviços com confirmação por parte da estação remota;

- Indication Filter and Arbitrator (IFA): trata da recepção de primitivas de indicação.

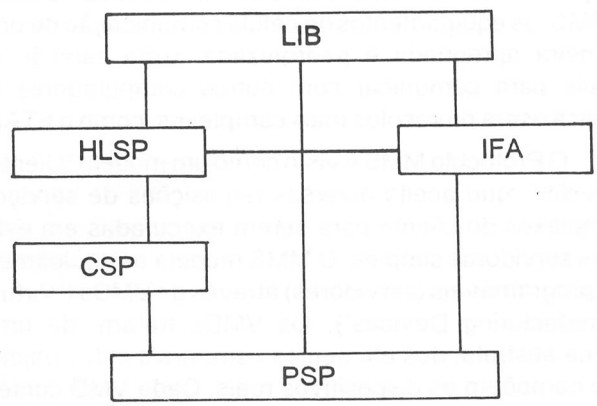

Figura 2.1 - Modelo de API do Projeto MAP/TOP

As particularidades do MMS podem levar a um esquema de resposta aos serviços solicitados remotamente em que a própria AI responda a estes serviços, tais como aqueles relacionados a gerenciamento de contexto, acesso a variáveis e gerenciamento de semáforos. Portanto, uma nova unidade funcional é adicionada à AI para processamento de indicações (solicitações remotas) denominada IP ("Indication Processing"). O IP tem as seguintes funções:

- Processar os serviços solicitados pelas primitivas de indicação aceitas;

- Acessar a memória principal e o sistema de arquivos local (chamando o VDRDB - ver abaixo);

- Solicitar ao CSP ou PSP (dependendo se o serviço é, ou não, confirmado) o envio de primitivas de resposta para as indicações aceitas.

Se a AI tem o bloco IP, o número de chamadas de INDICATION RECEIVE no AP pode ser decrementado, pois algumas primitivas de indicação são respondidas diretamente pela AI. Contudo, o AP precisa ainda chamar a função INDICATION RECEIVE para receber indicações que necessariamente são tratadas no AP, como por exemplo a indicação de ABORT e a indicação de CONCLUDE (para liberação da conexão).

Uma unidade funcional chamada VDRDB ("Virtual Device/Real Device Binding") é introduzida também na AI. Este bloco provê, quando necessário, o mapeamento entre os Dispositivos Virtuais da Manufatura (VMDs) e os aspectos funcionais dos Dispositivos Reais, além de ser usado pelo IP para acessar a memória e/ou o diretório de arquivos.

Por exemplo, a função auxiliar V-GET permite ao AP transformar a informação local (endereço e representação locais) em dado na representação MMS que pode ser enviado através da rede. A função auxiliar V-PUT transforma o dado MMS recebido da rede em representação local e coloca o dado em um endereço local. Estas funções usam o bloco VDRDB para realizar as transformações entre as representações MMS e local e para acessar a memória local.

Por outro lado, protocolos de aplicação interrelacionados são agrupados em Entidades de Aplicação (AEs - "Application Entities") na camada de aplicação. Por exemplo, a AE que contém o MMS deve conter também o Protocolo ACSE ("Association Control Service Element") para gerenciar as conexões. Desta forma, uma AI que requisita serviços de uma AE deve ter um componente lógico associado a cada protocolo de aplicação da AE. Os blocos funcionais TAOCF e RAOCF ("Transmission and Reception Association Object Control Function") supervisionam as interações entre os diversos componentes lógicos da AI para a transmissão e recepção de mensagens, respectivamente. No exemplo acima, os blocos TAOCF e RAOCF controlam as interações entre a AI-MMS e a AI-ACSE.

Por exemplo, na fase de estabelecimento de conexão, o AP providencia informação relacionada com o MMS e o ACSE para a AI-ACSE. A AI-ACSE interage com a AI-MMS para gerar e enviar uma primitiva para a camada de aplicação. Esta primitiva contém informação para o ACSE gerar a primitiva A_associate.req (requisição de estabelecimento de conexão) que tem no seu campo de informação a primitiva A_initiate.req (início de conexão MMS) gerado pelo MMS. Depois do estabelecimento da conexão, o AP chama funções da AI-MMS para realizar serviços específicos do MMS. A AI-MMS e a AI-ACSE têm, cada uma, os blocos LIB, HLSP, CSP, IFA, IP e VDRDB. Os blocos TAOCF e RAOCF supervisionam a interação entre os componentes lógicos AI-MMS e AI-ACSE.

O Modelo Geral proposto de AI é mostrado na Figura 2.2, e é analisado em [MAD90b].

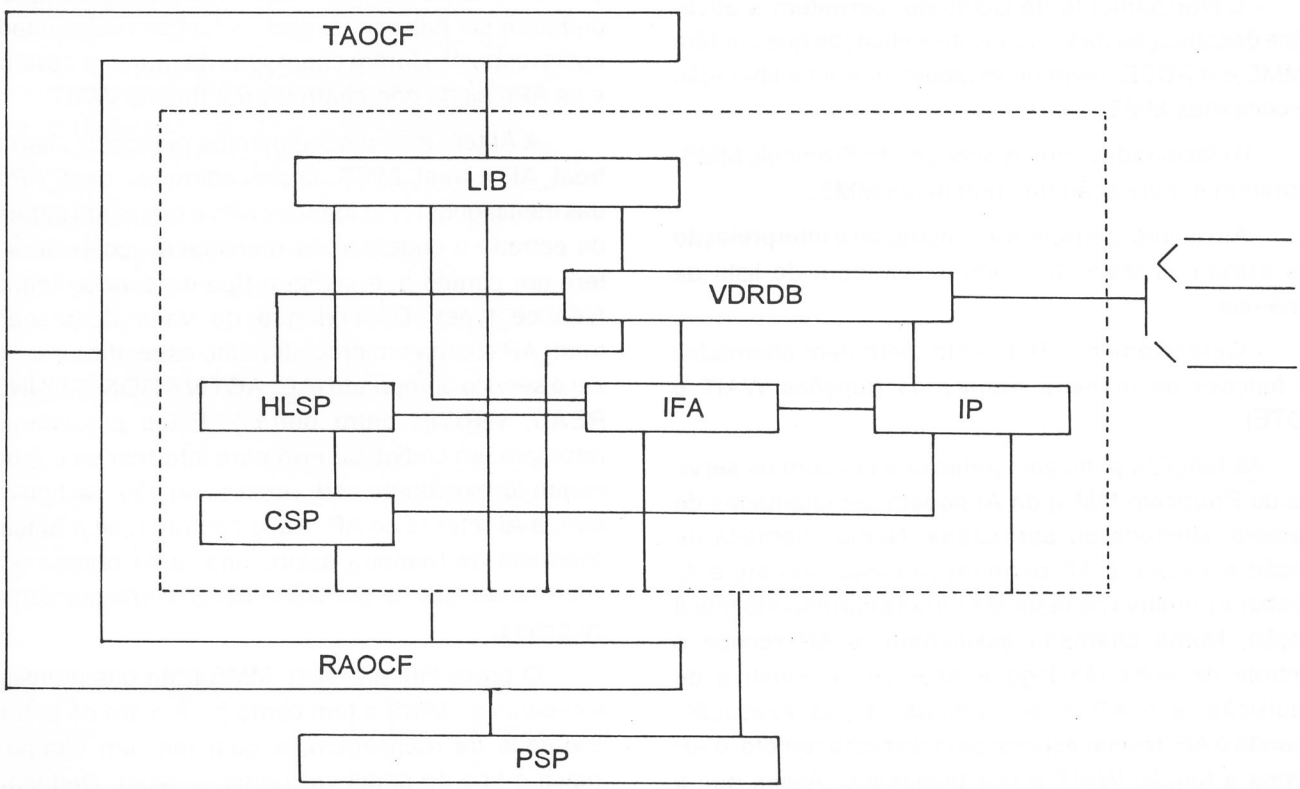

Figura 2.2 - Modelo Geral de Interface de Aplicação

O modelo proposto pode ser utilizado com outros protocolos de aplicação. Por exemplo, o Protocolo FTAM tem os conceitos de cliente e servidor e trabalha com uma estrutura de Sistema Virtual de Arquivos. O mapeamento entre o Sistema Virtual de Arquivos e o Sistema Real de Arquivos pode ser realizado na AI pelo VDRDB. O servidor é geralmente complexo e APs específicos respondem às requisições dos clientes. Contudo, o IP pode responder às requisições simples.

3 - EXEMPLO DE UMA IMPLEMENTAÇÃO DE INTERFACE DE APLICAÇÃO PARA O PROTOCOLO MMS

Usando o Modelo Geral proposto, uma AI para um subconjunto de serviços dos Protocolos MMS e ACSE foi implementada na UNICAMP, para suportar Gerenciamento de Contexto, Acesso a Variáveis e Suporte de VMD [MAD90a, MAD90b]. A AI foi escrita em C para microcomputadores compatíveis com o PC e roda sobre o sistema operacional DOS e um núcleo de tempo real que provê primitivas de comunicação e manipulação de tarefas. Uma nova versão da AI, também escrita em C, está sendo desenvolvida para rodar em estações de trabalho sobre o UNIX [MEN93].

Existem quatro tipos de funções da AI:

- Gerenciamento de Contexto: permitem a ativação e desativação da Entidade de Aplicação que contém o MMS e o ACSE, além do estabelecimento e liberação de conexões MMS;

- Relacionadas com os serviços do Protocolo MMS: permitem a invocação das primitivas MMS;

- Auxiliares: permitem a construção e interpretação das estruturas MMS, tal como a estrutura de lista de variáveis;

- Gerenciamento de Evento: permitem chamadas de funções de maneira assíncrona (funções WAIT e NOTE).

As funções principais (relacionadas com os serviços do Protocolo MMS) da AI podem ser chamadas de maneira, síncrona ou assíncrona. Numa chamada de função síncrona, o AP permanece bloqueado até a AI receber a primitiva de resposta remota correspondente à função. Numa chamada assíncrona, o AP recebe o controle de volta tão logo a AI envie a primitiva de requisição, e o AP pode continuar a sua execução. Quando o AP desejar esperar pela resposta remota, o AP chama a função WAIT e fica bloqueado. Assim que a primitiva de resposta remota chegue à AI, essa notifica o AP chamando a função NOTE.

A implementação utiliza os seguintes vetores [MAD92]: AE_LABEL_MAP, que define as invocações ativas da Entidade de Aplicação que contém os protocolos MMS e ACSE; CONNECTION_ID_MAP, que define as conexões estabelecidas, pendentes, abortadas e livres; INVOKE_ID_MAP, que define os serviços confirmados que ainda não obtiveram resposta remota; RETURN_EVENT_NAME_MAP, que define os nomes de eventos usados pelas chamadas assíncronas; LISTEN_MAP, que define as invocações de AE que estão esperando por pedidos remotos de conexão; WAIT_MAP, que define se um AP está, ou não, esperando por resposta remota de uma ou mais chamadas assíncronas e, finalmente, INDICATION_RECEIVE_MAP, que define se uma conexão está, ou não, esperando por primitivas de resposta remota.

As seguintes filas também são utilizadas: INTERFACE_API_AP, que contém mensagens (ainda com informações parciais) a serem retornadas aos APs como resposta às chamadas de funções da AI; INTERFACE_API_MMS, que contém as primitivas enviadas para o Protocolo MMS que requerem primitivas de resposta remota ou podem ser retransmitidas; e WAIT_QUEUE, que contém mensagens (com informação total) a serem retornadas aos APs, mas que não puderem ser entregues, pois as funções relacionadas a estas respostas foram chamadas de maneira assíncrona e os APs ainda não chamaram a função WAIT.

A AI tem dois procedimentos principais chamados treat_AP e treat_MMS. O procedimento treat_AP trata das mensagens recebidas dos APs e tem como parâmetro de entrada o endereço da mensagem recebida, a qual tem um campo que indica o tipo de serviço requerido (service_type). Dependendo do valor deste campo, treat_AP chama um procedimento específico para realizar o serviço apropriado (AE_ACTIVATION, CONNECT, READ, WRITE, entre outros). Estes procedimentos retornam um código de erro para informar se o procedimento foi executado com sucesso ou não. Se houve um erro, a AI retorna ao AP. Caso contrário, se a função foi chamada de maneira assíncrona, a AI retorna ao AP informando que a primitiva MMS correspondente foi invocada.

O procedimento treat_MMS trata das mensagens recebidas do MMS e tem como parâmetro de entrada o endereço da mensagem, a qual tem um campo que indica o tipo de primitiva (primitive-type). Dependendo do tipo de primitiva (resposta ou requisição remota - positiva ou negativa), um processamento específico é

executado. Depois, o treat_MMS verifica se a primitiva está relacionada a uma chamada de função síncrona ou assíncrona. Se esta chamada foi síncrona, o AP é notificado sobre o término do serviço. Se a chamada foi assíncrona, se o AP chamou previamente a função WAIT, o AP também é notificado sobre o término do serviço. Caso contrário, a mensagem de resposta ao AP é inserida na fila WAIT_QUEUE.

4 - UM EXEMPLO DE PROCESSO DE APLICAÇÃO

A Tabela 4-I mostra um exemplo de AP com chamadas à Interface de Aplicação para invocar serviços MMS. (1), (2), (11) e (12) são funções de gerenciamento de contexto; de (3) a (8) são funções relacionadas com o Protocolo MMS; e (9) e (10) são funções de gerenciamento de eventos.

Tabela 4-I

```
( 1)  mm_aeactivation (my_ae_name1,..., & ae_ label1, ...);
( 2)  mm_ connect (ae_label1,..., & con_id1);
( 3)  mm_status (con_id1,..., & generic_status1,...);
( 4)  mm_gnlist (con_id1,...,& name_type1,...);
( 5)  mm_gaattribute (con_id1,...);
( 6)  mm_gaattribute (con_id1,...);
( 7)  mm_write (con_id1, write_event, list1, & write_dcb1);
( 8)  mm_read (con_id1, read_event, listl, & read_dcb1);
( 9)  em_wait (wait_time1, & write_event);
(10)  em_wait (wait_time2, & read_event);
(11)  mm_conclude (con_id1,...);
(12)  mm_aedeactivation (ae_label1,...);
```

Inicialmente, o AP ativa a Entidade de Aplicação que contém os Protocolos MMS e ACSE chamando a função AE_ACTIVATION que tem como parâmetro de entrada o nome da AE, e como saída o AE_LABEL (identificador da invocação da AE). Depois, o AP requisita uma conexão através da função CONNECT que tem como parâmetro de entrada o AE_LABEL, e como saída o CONNECTION_ID (identificador da conexão). O AP pode requisitar outras conexões (outros CONNECTION-IDs serão providenciados). Depois disto, o AP chama algumas funções relacionadas com o MMS para pedir o estado da estação remota (Status), obter a lista de nomes de seus objetos (Get Name List), obter os atributos destes objetos (Get Access Attribute), e escrever e ler em objetos do tipo variável (Write e Read). Todas estas funções têm como parâmetros de entrada, pelo menos, o CONNECTION_ID e o RETURN_EVENT_NAME. As chamadas da função de gerenciamento de evento EM_WAIT determinam que o AP espere pela notificação de qualquer evento associado com chamadas assíncronas realizadas anteriormente. Finalmente, o AP requisita a liberação da conexão, chamando a função CONCLUDE, e então requisita o fim da invocação da AE, chamando a função AE_ACTIVATION.

5 - CONCLUSÃO

É recomendável apresentar os serviços de comunicação aos APs de uma maneira amigável devido à complexidade da camada de aplicação. Portanto, os sistemas de comunicação devem definir Interfaces de Aplicação. A complexidade destas interfaces depende das aplicações algumas vêzes, os APs podem requisitar diversos serviços da AI, tais como: confirmado, alto nível, mapeamento entre objeto real e virtual, respondedor, resposta automática, entre outros - outras vêzes, os APs podem requisitar apenas o envio e a recepção de mensagens genéricas. No primeiro caso, a AI pode ser construída de acordo com o Modelo Geral proposto. No segundo, a AI pode ter apenas uma biblioteca (LIB) simples além de Provedor de Serviço de Primitivas (PSP). Num caso intermediário, a AI pode, ou não, implementar alguns blocos e ligações do Modelo Geral, dependendo do tipo da estação, cliente ou servidora, e das particilariedades operacionais da estação.

A implementação mostrou que:

- Os blocos do Modelo Geral têm diversas atividades que independem de qual serviço é realizado. Portanto, as estruturas dos blocos são simplificadas;

- O esquema de processar requisições remotas na AI demonstrou ser eficiente, pois a AI não precisa chamar um AP para executar o serviço.

A AI também deve levar em consideração a existência de mais de um protocolo de aplicação numa AE para facilitar as tarefas dos APs. Os blocos TAOCF e RAOCF representam na AI esta característica da camada de aplicação.

REFERÊNCIAS

[FLE92] - FLEISCHMAN, E. - "An User's Guide to Data Communications APIs" - The Open Systems Newsletter; Parte 1: Maio 1992, pp. 1-8; Parte 2: Junho 1992, pp. 1-14

[FON89] - FONG, K. e REINSTEDLER, J. - "The Development of an OSI Application Layer Protocol Interface" - Computer Communication Review - ACM SIGCOMM, Julho 1989, pp. 21-57

[HAZ92] - HAZZARD, M. - "Distributed Services within IEEE TCOS/POSIX" - IEEE Network, Março 1992, pp. 22-23

[ISO9506] - ISO 9506 - "Manufacturing Message Specification" - Parte 1: "Service Specification"; Parte 2: "Protocol Specification", 1987

[MAD90a] - MADEIRA, E. R. M. e MENDES, M. J. - "Application Interface Model for Communication Software and an MMS Protocol Implementation Example" - Colloque International - CIM90 - Productique et Integrations, Bordeaux, France, Junho 1990

[MAD90b] - MADEIRA, E. R. M. e MENDES, M. J. - "An Application Interface Model for Communication Software" - IEEE Global Telecommunications Conference - GLOBECOM'90, San Diego, USA, Dezembro 1990

[MAD92] - MADEIRA, E. R. M. - "An Application Interface Model for Industrial Networks and its External View" - INFONOR'92 - Antofagasta, Chile, Dezembro 1992

[MAP88a] - MAP/TOP - "Application Interface Model and Specification Requirements", Junho 1988

[MAP88b] - MAP/TOP - "MMS Application Interface Specification", Junho 1988

[MEN89] - MENDES, M. J. - "Comunicação Fabril e o Projeto MAP/TOP" - IV EBAI - Santiago Del Estero, Argentina, 1989

[MEN93] - MENDES, M. J., MADEIRA, E. R. M. e outros - "SISDI-OSI: Sistema Didático para o Modelo OSI" - 11º Simpósio Brasileiro de Redes de Computadores - Campinas, SP, Maio 1993, pp. 3-19

[NEW92] - NEWNAN, O. - "SE-OSI: A Prototype Support Environmemt for Open Systems Interconnection" - Computer Communication Review - ACM SIGCOMM, Abril 1992, pp. 43-62