Conceitos de Engenharia de Protocolos Utilizados na Implementação de um Protocolo de Aplicação em Ambiente de Manufatura

Autores:

Pedro Frosi Rosa

Pós-Graduação Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo (PCS - EPUSP)

E-MAIL : frosi@lsi.usp.br

FAX : (011) 211-4574

Fone : (011) 815-9322, ramal 3589

Stefania Stiubiener

Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo (PCS - EPUSP)

E-MAIL : stefania@brusp.bitnet

FAX : (011) 211-4308

Fone : (011) 815-9322, ramal 3200

Resumo:

As Técnicas de Descrição Formal têm sido desenvolvidas para a especificação de serviços e protocolos de comunicação da OSI, e podem também ser usadas como linguagens de especificação para outras áreas de aplicação. Este artigo apresenta um exemplo completo de especificação formal, simulação e implementação da estrutura da camada de aplicação para o ambiente de manufatura.

A fim de alcançar este objetivo, nós trabalhamos com um elemento de serviço de aplicação, o padrão MMS ("*Manufacturing Message Specification*"), da camada de aplicação da arquitetura MAP ("*Manufacturing Automation Protocol*").

Abstract:

Formal Description Techniques have been developed for the specification of OSI communication protocols and services, and can also be used as specification languages for other application areas. This paper presents a complete example of formal specification, simulation and implementation of the application layer structure for the manufacturing environment.

In order to reach that goal, we had worked with a specific application service element, MMS (Manufacturing Message Specification) standard, from the MAP (Manufacturing Automation Protocol) architecture application layer.

1. Introdução

A proliferação de protocolos utilizados para a implementação de funções de redes de computadores e aplicações de usuários conduziu à necessidade de organizar as atividades envolvendo o uso de tais protocolos.

Assim nasceu a 'Engenharia de Protocolos' que atualmente abrange os aspectos ligados à especificação, ao projeto, à implementação e aos testes de protocolos.

O presente artigo se propõe a descrever, dentro dos conceitos da citada engenharia, a atividade de especificação formal, de verificação, de projeto de implementação e de testes do protocolo MMS ("*Manufacturing Message Specification*") pertencente à arquitetura MAP ("*Manufacturing Automation Protocol*") [Ref 5,8].

Inicialmente será descrito o modo de encaixe do MMS na estrutura do nível de aplicação [Ref 1]; em continuação será mostrada a arquitetura de especificação formal na técnica de descrição formal ESTELLE ("*Extended Transition Language*") [Ref 7], a validação dessa especificação, a arquitetura de "*software*" e os testes de interoperabilidade realizados.

2. Estrutura do Nível de Aplicação

A fim de facilitar o desenvolvimento dos serviços do nível de aplicação do modelo de referência OSI ("*Open System Interconnection*"), a ISO ("*International Standardzation Organization*") desenvolveu um ambiente de trabalho ("*framework*") consistindo de terminologia, conceitos, e uma estrutura para o desenvolvimento destes serviços.

Dentro deste ambiente de trabalho, chamado de estrutura do nível de aplicação (ALS - "*Application Layer Structure*"), o modelo arquitetural, para todos os serviços do referido nível, é especificado [Ref 1].

Na definição deste modelo, os 'arquitetos' da ISO procuraram reduzir o uso dos termos de uma implementação real, tais como processos ("*process*") e bibliotecas ("*libraries*"), a fim de evitar a perda de generalidade do modelo em torno de um sistema oferecido por algum fabricante específico.

Na terminologia ALS, uma aplicação fazendo uso dos serviços OSI, para se comunicar, é chamado um Processo de Aplicação (AP - "*Application Process*"). Note que a palavra processo não necessariamente tem o mesmo significado que em sistemas operacionais.

Em ALS, um Processo de Aplicação é simplesmente uma abstração de uma aplicação real. A especificação da ALS define uma estrutura para um processo de aplicação que consiste de um conjunto de componentes com funções bem definidas.

Estes componentes podem ser empregados em diferentes combinações com a finalidade de oferecer diferentes serviços.

Um Processo de Aplicação é constituído de zero, um ou mais componentes, denominados Entidades de Aplicação (AE - "*Application Entitie*"), conforme está representado na Figura 1.

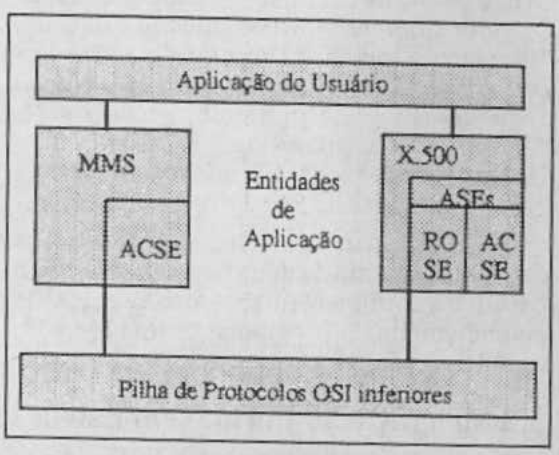

Figura 1 - Um exemplo da relação entre ASEs e ALS

Uma AE pode ser vista como um serviço da rede, tal como uma facilidade de transferência de arquivo, que emprega um ou mais protocolos da rede. Os protocolos da rede são chamados Elementos de Serviço de Aplicação (ASE - "*Application Service Element*").

Uma aplicação de monitoração do estado dos dispositivos do chão de fábrica é um exemplo de um processo de aplicação. Ele pode utilizar os serviços oferecidos pelos ASEs MMS e X.500 (serviço de diretório) [Ref 9].

O MMS é utilizado para ler o estado das variáveis dos dispositivos da fábrica e o X.500 para acessar o diretório (uma base de dados) contendo informações de endereçamento sobre as várias aplicações da fábrica.

Os ASEs são representados pelo MMS e ACSE ("*Application Control Service Element*") [Ref 10], para os serviços de troca

de mensagens entre dispositivos de manufatura, e pelo X.500, ACSE e ROSE ("*Remote Operation Service Element*") [Ref 1], para os serviços de diretório.

O ACSE é um protocolo do nível de aplicação utilizado para iniciar e encerrar (de modo abrupto ou negociado) uma associação entre aplicações, que podem ser do mesmo ou de diferentes sistemas.

O ROSE é um protocolo genérico OSI que permite às aplicações invocar interações de requisições/respostas de aplicações em sistemas remotos.

Neste artigo será apresentada uma aplicação do ASE MMS que implementa os serviços "*Initiate*", "*Cancel*", "*Abort*", "*Reject*", "*Conclude*", "*Status*", "*Identify*", "*Read*", "*Write*", "*Input*", "*Output*", "*Load Domain Content*", "*Store Domain Content*", "*Request Domain Download*", "*Request Domain Upload*", "*Initiate Download Sequence*", "*Download Segment*", "*Terminate Download Sequence*", "*Initiate Upload Sequence*", "*Upload Segment*", "*Terminate Upload Sequence*".

É importante ressaltar que os serviços de gerenciamento de domínios servirão como base para a implementação dos serviços de gerenciamento de programas (estes não foram tratados ainda).

3. Especificação Formal em Estelle

As chamadas Técnicas de Descrição Formal (FDT - "*Formal Description Technique*") são utilizadas nas descrições dos comportamentos de um sistema de forma clara e sem ambiguidades.

Em "*ESTELLE*", do ponto de vista arquitetural, sistemas distribuídos são descritos em termos de módulos. Um sistema, como um todo, é um módulo (denominado "*Specification*") que poderá ser refinado em vários módulos filhos. Estes módulos filhos podem, por sua vez, ser, cada um, refinados em módulos filhos, e assim sucessivamente.

Um módulo constitui-se de um cabeçalho, onde serão definidas as características visíveis externamente (as portas de entrada e saídas dos módulos, por exemplo), e um corpo, que conterá os aspectos do comportamento do módulo. O módulo "*Specification*" constitui a única exceção, ou seja, ele é constituído apenas da parte referente ao corpo de um módulo.

Os módulos, do ponto de vista das suas execuções, podem ser divididos em duas categorias: processo ("*process*"), que são módulos que podem ser executados em paralelos, e atividade ("*activity*"), que são módulos que não podem ser executados em paralelos com outros módulos (isto é, são executados sequencialmente).

Dentro de uma especificação, módulos filhos que tenham algum tipo de afinidade podem ser agrupados dentro de módulos denominados sistemas. Neste caso haverá dois tipos de módulos sistemas: o processo ("*SystemProcess*"), que poderá ser refinado em módulos processos ou atividades, e o atividade ("*SystemActivity*"), que poderá ser refinado, apenas, em módulos atividades.

De fato, as características descritas nos dois parágrafos anteriores são chamadas atributos de um módulo. Um módulo não deverá necessariamente ter algum dos atributos citados, neste caso diz-se que o módulo é não atribuído.

Apenas o módulo "*Specification*" e módulos não atribuídos poderão ser refinados em módulo sistemas.

Quando dois ou mais módulos com atributo atividade ("*SystemActivity*" ou "*Activity*") estão habilitados à execução, a escolha do módulo que será executado é aleatória, ou seja, as execuções de módulos atividade são sequenciais e aleatórias.

O critério citado é válido para módulos em um mesmo nível na hierarquia, isto é, quando o conflito envolve um módulo filho e seu pai, este tem prioridade.

A técnica de descrição formal ESTELLE é baseada numa máquina de estados finita estendida, sendo que esta característica é utilizada na descrição do comportamento dos módulos (os corpos dos módulos).

Sómente módulos atribuídos poderão especificar uma máquina de estados estendida.

Um módulo pode ser visto como uma caixa preta capaz de comunicar com outros módulos, incluindo o módulo pai, se for o caso. A comunicação é feita através da troca de mensagens chamadas "Interações". As interações são recebidas em locais denominados "Pontos de Interações".

A cada Ponto de Interação está associada uma fila com algoritmo de escalonamento FCFS ("*First Come First Server*"). A fila poderá ser comum ("*Common Queue*"), compartilhada entre os pontos de interações de um módulo (aqueles que fazem

uso da fila comum), ou individual ("*Individual Queue*"), utilizada por um único ponto de interação.

Os pontos de interações podem ser declarados tanto no cabeçalho como no corpo do módulo. No primeiro caso diz-se que eles são Pontos de Interações Externos e no segundo caso diz-se que eles são Pontos de Interações Internos.

Os pontos de interações externos são utilizados nas interações com outros módulos em um mesmo nível de hierarquia ou com o módulo pai. Os pontos de interações internos são utilizados nas interações com módulos filhos.

As trocas de interações são feitas através de um enlace bidirecional chamado Canal. Em realidade, o canal define os pontos de interações que podem ser atrelados e os tipos das estruturas de dados que podem ser trocadas.

Para que tais pontos possam trocar interações eles devem ser previamente atrelados. Esse atrelamento pode ser feito de duas maneiras: na primeira, os pontos são conectados (através da assertiva "*Connect*") o que garantirá que uma interação enviada através de um ponto de interação será depositada na fila do ponto de interação destino.

Na segunda, os pontos são ligados (através da assertiva "*Attach*"), onde um dos pontos poderá fazer parte de uma ligação ou de uma conexão, isto é, trata-se de um enlace que tem entre os pontos de interações extremos outros pontos de interações, de qualquer forma, as interações enviadas por um ponto (obrigatóriamente este ponto deverá estar em uma das extremidades) serão colocadas na fila do ponto de interação da outra extermidade do enlace.

Mesmo que o atrelamento seja desfeito, através das assertivas "*Disconnect*" ou "*Detach*" conforme o método usado no atrelamento, as interações existentes nas filas não serão perdidas e serão processadas em algum momento.

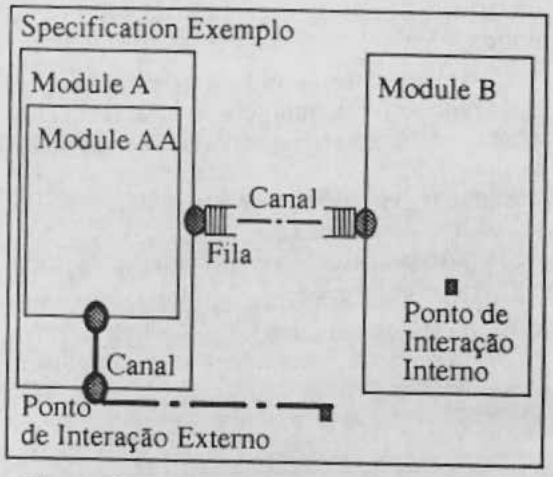

Figura 2 - Representação dos elementos estruturais da FDT ESTELLE

Obs:

1. O Canal feito em linha continua representa um atrelamento feito através da assertiva "*Attach*", já que o ponto de interação externo do Módulo A (para se comunicar externamente) deverá se conectar a um ponto de interação: interno (ao módulo "*Specification*") ou externo (de outro módulo no mesmo nível);
2. O Canal feito em linha tracejada representa um atrelamento feito através da assertiva "*Connect*", já que se tratam de dois pontos de interações externos de módulos no mesmo nível;
3. Todos os pontos de interações têm uma fila, de um dos tipos citados. A omissão nos demais pontos foi para não carregar o desenho.

Outros termos e conceitos relacionados com esta FDT podem ser encontrados em [Ref_7].

3.1. O Processo de Aplicação

A especificação do processo de aplicação será feita em termos dos módulos que o compõem. No âmbito deste artigo deter-nos-emos na descrição funcional de tais módulos.

Inicialmente, um processo de aplicação pode ser refinado em dois módulos: um

representando a parte do usuário-final e outro representando a entidade de aplicação.

O módulo usuário-final, doravante referenciado como UP (Processo do Usuário), conteria as seguintes tarefas: aplicação do usuário (um programa que controla algum dispositivo), interface para as aplicações do usuário, gerente de conexão (do UP com a entidade de aplicação) e gerente de evento.

O módulo entidade de aplicação, doravante referenciado como SPP (Processo Provedor de Serviços), conteria as seguintes tarefas: detectar chamadas para o SPP, coordenar os ASEs dentro de um SPP, gerenciar conexões (do SPP com SPP-parceiro) e os ASEs propriamente ditos.

A Figura 3 mostra os relacionamentos dos módulos citados, e nos parágrafos seguintes dar-se-á uma breve descrição de suas funções.

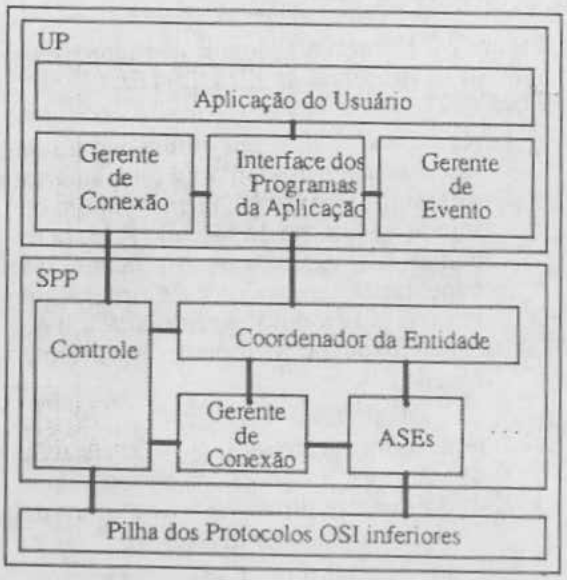

Figura 3 - Relacionamento dos Módulos Estelle de um Processo de Aplicação

Na figura anterior todos os módulos são do tipo processo ("*Systemprocess*" ou "*Process*"). As linhas, que ligam os módulos, representam canais.

3.1.1. Descrição do Módulo UP

É um módulo do tipo processo ("*Systemprocess*") que se relaciona externamente com o módulo SPP (3.1.2). Basicamente, este módulo representa as necessidades do usuário.

Todos os seus módulo filhos são, também, do tipo processo ("*Process*").

Aplicação do Usuário

Esta é um programa ("*Application Program*") que o usuário-final utiliza para acessar os serviços (subjacentes) da rede.

Interface dos Programas da Aplicação

Conhecida como API ("*Application Program Interface*"), é através desta que as aplicações do usuário obtém acesso à serviços específicos da rede.

Gerente de Conexão

Este módulo é responsável pelo gerenciamento do estabelecimento e encerramento de todas as conexões (com SPPs) da aplicação do usuário.

Gerente de Evento

Este módulo mantém as requisições pendentes feitas pela aplicação do usuário à interface de programas de aplicação. A aplicação do usuário sincroniza os encerramentos de serviços através destas pendências.

3.1.2. Descrição do Módulo SPP

É um módulo do tipo processo ("*Systemprocess*") que se relaciona externamente com um módulo SPP-remoto e um módulo UP.

Basicamente, este módulo deixa a rede transparente para a aplicação do usuário.

Todos os seus módulo filhos são, também, do tipo processo ("*Process*").

Coordenador da Entidade

Este módulo tem a responsabilidade de coordenar as interações entre os ASEs presentes em um SPP. Por exemplo FTAM e ACSE.

Gerente de Conexão

Este módulo desempenha funções importantes associadas à iniciação e término do SPP, estabelecimento e fim de conexão com SPP-remoto.

Controle

Este é o principal módulo de controle dentro de um SPP. Sua função principal é detectar chamadas para o SPP (locais ou remotas).

Quando a chamada vem da aplicação, este módulo invoca o Coordenador da Entidade.

Quando a chamada vem de um SPP-remoto este módulo invoca o Gerente de Conexão.

ASEs

O SPP utiliza um ou mais ASEs para constituir um particular serviço de rede.

Os serviços oferecidos por um ASE, e a interface utilizada para o acesso a estes serviços, são especificados em padrões OSI.

4. Validação da Especificação

A especificação formal comentada no ítem anterior aumenta sua importância quando submetida à simulação.

Na simulação podemos levantar vários aspectos sem realmente efetuar a implementação, isto é, qualquer comportamento anômalo pode ser resolvido ainda na fase de especificação.

Na área de projetos de protocolos, como se trata com a ocorrência de eventos distribuídos, é muito difícil prever todas as situações possíveis, por exemplo, uma ocorrência de travamento ("*deadlock*") pode passar despercebida.

No nosso caso utilizamos uma ferramenta denominada *ESTIM* ("*ESTELLE Simulator*") para a submissão da nossa especificação.

Verificamos que havia uma situação anômala quando da requisição do serviço "*Cancel*".

O cenário era o seguinte: o cliente-MMS dispara uma requisição de serviço, e, antes que volte uma confirmação para o serviço requerido, o mesmo cliente tenta, então, cancelar o referido serviço.

O travamento ocorria quando a requisição de cancelamento era disparada depois que o servidor-MMS já tinha respondido (isto é, executado) o serviço em questão.

Em resumo, o cliente tentava cancelar um serviço que não estava concluído do ponto de vista do cliente, mas sim do ponto de vista do servidor. Esta situação pode ser vista na Figura 4 (por exemplo, caminho 1->2->3->6->9->**13**->travado).

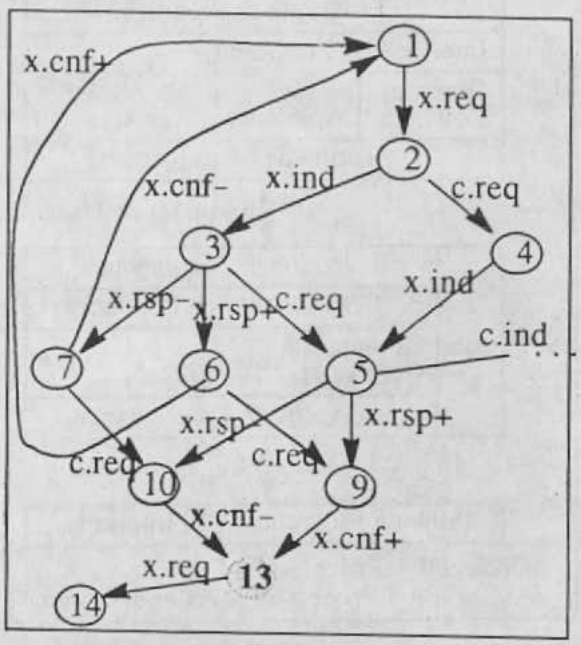

Figura 4 - Situação anômala detectada

5. Arquitetura da Implementação

Esta arquitetura está baseada na especificação da ALS apresentada no ítem 3. Nesta fase aparecem alguns módulos, dependentes da implementação, cujas presenças são pertinentes agora.

Para facilidade podemos imaginar cada módulo da fase anterior como um processo UNIX[1] independente.

Nesta arquitetura um processo de aplicação é implementado como um ou mais processos UNIX trabalhando juntos. Inicialmente podem ser distinguidos dois tipos de processos: o processo provedor de serviços (SPP - "*Service Provider Process*") e o processo do usuário (UP - "*User Process*").

O SPP é um processo que representa uma entidade de aplicação. Ele é composto de um conjunto de módulos representando ASEs e outros protocolos e funções.

O UP é um processo que contém um programa de aplicação que utiliza os serviços oferecidos por um ou mais SPPs.

A Figura 5 é uma representação do que foi dito.

1. UNIX é uma marca registrada da AT&T nos EUA e outros países.

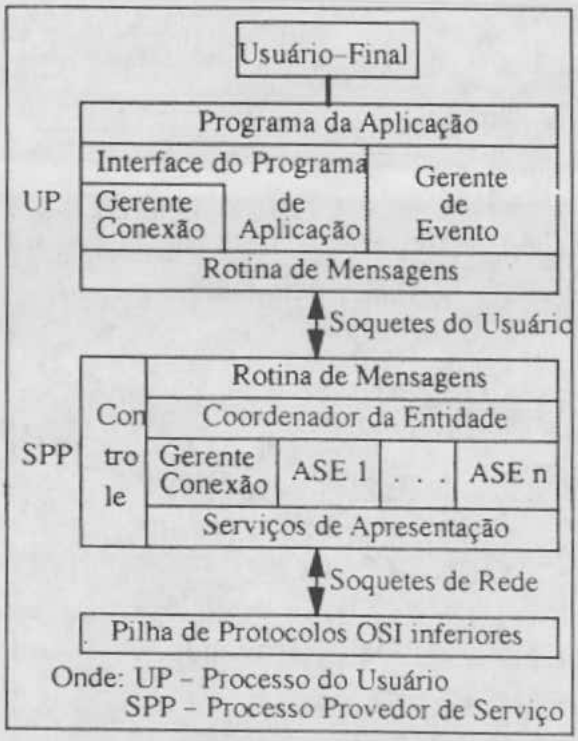

Figura 5 - Arquitetura do Processo de Aplicação

O usuário (um robô, um CLP, etc) inicia um UP que resulta primitivas que ativarão um ou mais SPPs, isto é, cada chamada resultará em operações "*fork()*" e "*exec()*" do UNIX sobre um SPP.

5.1. Processo do Usuário

Os cinco módulos, presentes no UP, oferecem ao usuário-final ("*End-User*") uma interface transparente para a utilização dos serviços subjacentes.

5.1.1. Programa de Aplicação

Este é o programa ("*Application Program*") que o usuário-final utiliza para acessar os serviços (subjacentes) da rede. Para o MAP 3.0 ("*Manufacturing Automation Protocol*"), os programas de aplicação são aplicações do MAP utilizados para a automação da fábrica.

5.1.2. Interface para os Programas

A interface dos programas de aplicação (API - "*Application Program Interface*") é através da qual os programas de aplicação obtém acesso aos serviços específicos da rede, tais como o MMS e o X.500. Para o MAP 3.0, estas interfaces são especificadas pelos subcomitês dele.

5.1.3. Gerente de Eventos

Este módulo mantém as requisições pendentes feitas pelo programa de aplicação à interface. O programa de aplicação utiliza o gerenciamento de eventos para sincronizar a finalização das requisições.

5.1.4. Gerente de Conexão

Este módulo é responsável pelo gerenciamento do estabelecimento e encerramento de todas as conexões do programa de aplicação.

5.1.5. Rotina de Mensagens

As interações entre o programa de aplicação e os SPPs são na forma de mensagens sobre soquetes no domínio UNIX. Elas implementam estas trocas, ocultando os mecanismos específicos utilizados nos programas de aplicação.

Quando um programa de aplicação ativa um ou mais SPPs, ele pode fazer uma requisição, através da API, para estabelecer uma conexão com uma aplicação remota. Cada requisição é passada como uma mensagem para o apropriado SPP através da Rotina de Mensagens.

Dentro do SPP, as conexões com aplicações remotas e a conexão com o programa de aplicação são tratadas por soquetes. O conceito de soquetes é aquele disponível no sistema operacional UNIX 4.3 BSD, de Berkeley.

Os soquetes utilizados para a conexão com o programa de aplicação serão chamados soquetes de usuário (Domínio UNIX), e os soquetes utilizados para as conexões remotas Soquetes de Rede (Domínio INTERNET).

5.2. Processo Provedor de Serviços

Este processo é uma entidade de aplicação na terminologia OSI. Os módulos em um SPP implementam um particular serviço da rede.

5.2.1. Controle

Este é o principal módulo de controle em um SPP. Sua função é detectar chamadas nos soquetes do SPP.

Quando uma chamada é detectada em um soquete do usuário, este módulo invoca o coordenador da entidade de aplicação.

Quando uma chamada é detectada em um soquete de rede, este módulo invoca os serviços de apresentação.

Baseado no lugar onde a chamada ocorre, o coordenador da entidade de

aplicação ou o serviço de apresentação que processará a chamada, se necessário, invocará outros módulos SPP.

5.2.2. Rotina de Mensagens

Como a parceira no processo do usuário, esta rotina implementa a troca de mensagens baseada na comunicação entre o programa de aplicação e o SPP.

5.2.3. Gerenciamento de Conexão

Este módulo desempenha funções importantes associadas com a iniciação, estabelecimento ("*set up*") da conexão e terminação do SPP.

Para o estabelecimento, o gerente da conexão desempenha uma busca no diretório a fim de obter o endereço (bem como outros parâmetros do contexto) da aplicação remota.

Uma vez que estes dados são obtidos, ele junta as informações da unidade de dados do protocolo e invoca os Serviços de Apresentação a fim de enviar uma requisição de conexão.

5.2.4. Elemento de Serviço de Aplicação

O SPP utiliza um ou mais protocolos do nível de aplicação, ou ASEs ("*Application Service Element*"), para constituir um particular serviço da rede.

Os serviços oferecidos por um ASE, e a interface utilizada para o acesso a estes serviços, são especificados pelos padrões OSI.

Por exemplo: MMS, X.500, FTAM, ROSE e ACSE são ASEs com uma especificação OSI correspondente.

5.2.5. Coordenador da Entidade

Este módulo oferece serviços de alto nível em função de serviços de baixo nível presentes nos ASEs.

Por exemplo: a interface de aplicação para o FTAM, no MAP 3.0, especifica um serviço para copiar arquivos. O Coordenador oferece este serviço utilizando várias primitivas de baixo nível especificadas no FTAM da OSI.

O relacionamento dos ASEs envolvidos em uma entidade de aplicação, também, é função deste módulo.

5.2.6. Serviços de Apresentação

Oferece os serviços especificados pela camada de apresentação OSI e algumas características adicionais, tais como, manusear e negociar o contexto de apresentação, codificação e decodificação das unidades de dados do protocolo (PDU - "*Protocol Data Unit*"), fragmentação e envio das PDUs.

5.2.7. Utilitários

Este módulo (não representado na figura 1) oferece serviços para outros SPPs, tais como, gerenciamento do uso de memória, gerenciamento da memória compartilhada e manipulação de sinais ("*signal handling*").

6. Notação de Sintaxe Abstrata

As PDUs dos ASEs e da camada de apresentação são especificadas nos padrões OSI em um formato chamado 'Notação de Sintaxe Abstrata Um' (ASN.1 - "*Abstract Syntax Notation One*").

Ela é uma notação próxima do 'BNF' ("*Bacus Naum Form*") que especifica a estrutura e o conteúdo de uma PDU.

Incluídos na notação estão vários tipos simples, tais como boleano, inteiro, etc, e tipos construídos que são compostos de tipos simples e/ou construídos.

Este mecanismo pode ser aplicado recursivamente e gerar tipos complexos arbitrários.

Por exemplo:

```
AssociatePDU ::= SEQUENCE {
        versionNumber   INTEGER,
        maxPDUsize      INTEGER,
        calledAETitle   ApplicationTitle,
        callingAETitle  ApplicationTitle,
        userData        [UNIVERSAL 6]
                        IMPLICIT
                        OCTET STRING

ApplicationTitle ::= OCTET STRING
```

Os ASEs precisam tratar as PDUs que eles enviam ou recebem. Como as PDUs podem ser complexas, escrever uma função para manipulá-las pode ser assaz trabalhoso.

Esta tarefa pode ser ajudada pelo uso de um compilador. Isto é desejável pois ASN.1 é uma notação formal e pode perfeitamente ser tratada, no sentido da codificação e decodificação, por um processo automático.

A especificação ASN.1, da estrutura de dados de um ASE, serve diretamente como uma entrada para um compilador que gere um conjunto de estruturas de dados da linguagem 'C' e rotinas de codificação e decodificação (funções da linguagem 'C').

As estruturas de dados geradas devem ser incluídas no código do ASE. Uma PDU (como parte da estrutura de dados) é armazenada nestas estruturas antes da codificação ou após a decodificação.

As rotinas de codificação e decodificação são ligadas ("*linked*") nos ASEs pelo módulo Serviços de Apresentação. Algumas funções genéricas, chamadas bibliotecas de tempo de execução ("*run-time libraries*"), também são ligadas nele. Este processo está representado na Figura 6.

Neste trabalho, em razão de não dispormos de um 'compilador' ASN.1, foram implementadas dedicadamente (isto é, para atender ao contexto desta implementação) duas funções: uma para codificação das PDUs a serem enviadas e outra para a decodificação das PDUs sendo recebidas, além da adição das respectivas estruturas de dados.

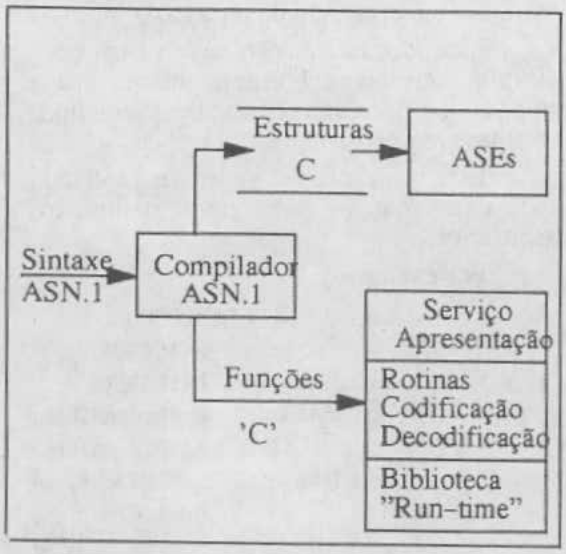

Figura 6 - Processo de Codificação e Decodificação de PDUs.

6.1. Tratamento dos Dados

Abordar-se-á aqui como a troca de dados entre dois programas da aplicação, conectados, é feita.

O usuário-final envia os dados ao SPP, através de uma mensagem que é lida pelo Coordenador da Entidade de Aplicação, que serão tratados pelo ASE.

Neste ponto os dados estão em uma estrutura da linguagem 'C', não codificada ainda.

O ASE invoca, então, o módulo Serviços de Apresentação a fim de codificar os dados. Isto resulta em uma área contígua de memória ("*buffer*") contendo a PDU da aplicação.

O ASE envia esta área para o módulo Serviços de Apresentação que, em seguida, a enviará à aplicação remota.

Imediatamente antes de enviar a PDU, o módulo Serviços de Apresentação acrescentará as informações de controle do protocolo (PCI - "*Protocol Control Information*").

Desde que cada PDU pode conter tipos de dados de diferentes contextos de apresentação, este módulo, também, identifica o contexto da PDU.

Isto significa estabelecer referências (ponteiros) às locações dentro da PDU para cada diferente contexto de apresentação, e manter um contador referenciando o número de contextos.

Neste trabalho foi considerado apenas com um contexto de apresentação (ver figura 3).

6.2. Conexão e Transmissão de PDUs

Quando uma conexão é estabelecida, cada ASE define os possíveis tipos de dados que podem ser enviados. O conjunto destes tipos é chamado Contexto de Apresentação.

Os Serviços de Apresentação negociam estes contextos, com as aplicações remotas, definindo os parâmetros a serem utilizados por cada ASE parceiro.

Uma vez que a negociação termina, os Serviços de Apresentação garantem que quaisquer dados enviados ou recebidos estão de acordo com o contexto negociado.

Quando um ASE envia dados para uma aplicação remota, estes dados, antes, necessitam ser codificados em PDUs.

Do mesmo modo, quando uma aplicação recebe PDUs, elas devem ser decodificadas e então enviadas para os ASEs (que atendem ao contexto da PDU).

Estas funções constituem a codificação, decodificação e envio das PDUs.

Se uma PDU é muito grande, não sendo possível enviá-la de uma única vez, então ela deve ser fragmentada.

Os Serviços de Apresentação fragmentam tais PDUs, quando no envio, e remontam-nas quando são recebidas.

7. Ambiente de Implementação

O ambiente de implementação será composto de estações de trabalho "*Sun*" ("*Sun Microsystems Inc*"), sistema

operacional "*SunOS 4.1.1*" ("*Sun Operating System*" compatível com o sistema "*Unix 4.3 BSD*") e rede local de comunicação baseada no padrão "*Ethernet*".

A Figura 7 a seguir mostra um instante de comunicação envolvendo duas estações, uma requisitante e a outra respondedora.

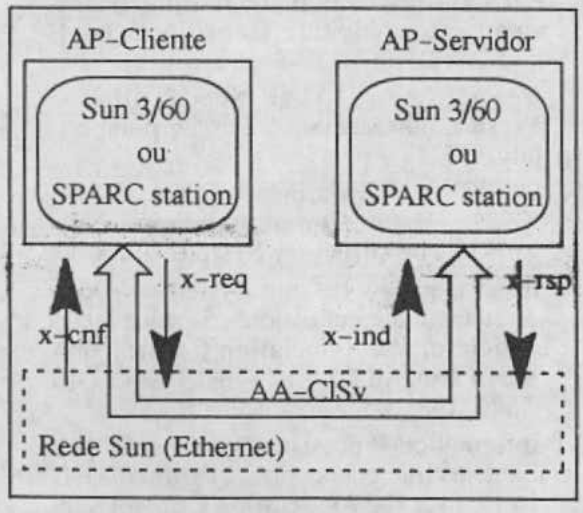

Figura 7 - Ambiente de implementação

8. Testes de Interoperabilidade

Estes testes consistem em definir um conjunto de serviços a ser requisitados pela Aplicação do Usuário (cliente). Estas requisições serão tratadas na Aplicação do Usuário parceiro (servidor).

Basicamente foram utilizados os serviços de gerenciamento de contexto (foi dada ênfase ao estabeleciemento de uma conexão conforme ítem 8.1.), os serviços de leitura e escrita em variáveis remotas e os serviços de gerencimanto de domínios.

8.1. Estabelecimento de uma conexão

Mostra-se aqui um exemplo de como os elementos arquiteturais, de um processo de aplicação, interagem. Note que somente um ASE é utilizado aqui.

Assume-se que antes do início das interações, o usuário-final tenha disparado o processo do usuário, que por sua vez instanciou um SPP.

Isto pösto, o programa da aplicação do usuário-final quer requisitar uma conexão com uma aplicação remota. A Figura 8 mostra a sequência de eventos que será descrita a seguir.

Figura 8 - Exemplo de uma iniciação de conexão

1. A aplicação requisita uma conexão para a API, com parâmetros específicos da aplicação;
2. Uma primitiva gerada pela API invoca o gerente de eventos, a fim de registrar um evento;
3. A API invoca o gerente de conexão, do processo do usuário, a fim de iniciar o estabelecimento de uma conexão;
4. O gerente de conexão envia uma mensagem ao SPP, através da Rotina de Mensagens, para iniciar o estabelecimento de uma conexão;
5. O módulo de controle, que está em um "*loop*" infinito, detecta um evento;
6. Como um evento foi detectado em um soquete do usuário, o módulo de controle invoca o coordenador da entidade de aplicação para tratá-lo;
7. O coordenador utiliza a rotina de mensagens para ler a mensagem enviada pelo processo do usuário;
8. Em se tratando de uma mensagem que é uma requisição de conexão, o coordenador requisita ao gerente de conexão do SPP o estabelecimento de uma conexão;
9. O gerente de conexão invoca o ASE a fim de obter a PDU que necessita para enviar a requisição de conexão;
10. O ASE define seu contexto de apresentação e então codifica sua PDU. Quando tais atividades acabam ele retorna (a PDU) ao gerente de conexão;
11. Tendo obtido a PDU do ASE, o gerente de conexão requisita o serviço de apresentação para enviar a primitiva;
12. O serviço de apresentação envia a requisição através de um soquete de rede, criado para esta conexão;
13. O serviço de apresentação avisa ao módulo controle sobre o novo soquete, que deve ser adicionado à lista de soquetes que ele monitora;

14. De volta a seu "*loop*" infinito, o módulo Controle, novamente, espera pela confirmação do estabelecimento da conexão que deve chegar da aplicação remota.

9. Conclusão

A arquitetura do processo de aplicação foi utilizada para desenvolver uma entidade de aplicação, isto é, um serviço da rede, no ambiente SUNOS 4.1.1 (equivalente ao UNIX BSD 4.3), do MAP 3.0 - o ambiente MMS.

Este trabalho pode ser visto como um ponto de partida para projetistas interessados em desenvolver aplicações OSI, para tanto, basta identificar os componentes estruturais (com as respectivas interfaces); de quaisquer aplicações.

O componente central, o MMS, pode ser trocado por outros ASEs, o que permite aos projetistas a utilização de ASEs específicos gerando, assim, novos ambientes.

Pode-se dizer que o trabalho constitui um exemplo do emprego da metodologia de tratamento de protocolos, em todas as fases do projeto, incluindo todos os mecanismos fornecidos pelos padrões de especificação e de implementação na área.

Bibliografia

1. Information Processing Systems - Open Systems Interconnection - **Application Layer Structure**, ISO DP 9545, july 1987.
2. MAP Technical Subcommitee, **MAP 3.0 Application Interface Specification**, 1988.
3. Information Processing Systems - Open Systems Interconnection - **Specification of Abstract Syntax Notation One (ASN.1)**, ISO 8824, 1987.
4. Information Processing Systems - Open Systems Interconnection - **Specifiction of Basic Encoding Rules for Abstract Syntax Notation One**, ISO 8825, 1987.
5. Information Processing Systems - Open Systems Interconnection - **Service Definition of Manufacturing Message Specification**, ISO 2DP 9506, 1988.
6. Information Processing Systems - Open Systems Interconnection - **Protocol Definition of Manufacturing Message Specification**, ISO 2DP 9506, 1988.
7. Information Processing Systems - Open Systems Interconnection - **ESTELLE - A Formal Description Technique Based on an Extended State Transition Model.** ISO 2nd DP 9074, 1986.
8. General Motors, **MAP Specification - Version 3.0**, General Motors Corporation, july 1987.
9. CCITT Draft Recomendations X.500 - X.521, **Data Communications Networks: The Directory System**, 1988.
10. Information Processing Systems - Open Systems Interconnection - **Service Definition for the Association Control Service Element.** Revised Final Text of DIS 8649, 1988.
11. Information Processing Systems - Open Systems Interconnection - **Protocol Definition for the Association Control Service Element.** Revised Final Text of DIS 8650, 1988.
12. Information Processing Systems - Open Systems Interconnection - **Revised Text of ISO 9074: 1989/PDAM 1, Information Technology - Open Systems Interconnection - ESTELLE - A FDT Based on an Extended State Transition Model - Proposed Draft Amendment 1: Estelle Tutorial.** ISO/IEC JTC 1/SC 21 N 5710, 1991.
13. Ernberg, P; et al **Guidelines for Specification and Verification of Communication Protocolos.** Swedish Institute of Computer Science, SICS Perspective, Report n.1, 1991.
14. Mazzola, V. B. **Contribution à la Conception de Systèmes Flexible de Production - Application de la Technique de Description Fomelle Estelle.** These présentée au LAAS du CNRS en vue de l'obtention du titre de Docteur de l'Université Paul Sabatier de Toulouse, Rapport LAAS n. 91378, Toulouse, France, 1991.
15. Carmo, L. F. R.; Courtiat, J. P.; Hummel, C. G.; Mazzola, V. B. **Modelisation et Verification du Protocole MMS.** Rapport Final - Convention Adermip (LAAS - EDF), Rapport LAAS n. 91391, Toulouse, France, 1991.