3º SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES (3º SBRC)

TÍTULO: UTILIZAÇÃO DOS CONCEITOS DE MÓDULO, PORTA E CANAL EM ESPECIFICAÇÕES FORMAIS DE SERVIÇOS, PROTOCOLOS E INTERFACES DE COMUNICAÇÃO

AUTOR: Wanderley LOPES DE SOUZA

GRC/DSC/Universidade Federal da Paraíba
(trabalho realizado junto ao IRO/Université de Montréal,
com auxílio fornecido pelo CNPq)

SUMÁRIO: O esforço que esta sendo realizado nos últimos anos, visando solucionar os problemas relativos à concepção de Sistemas Distribuidos, tem levado à utilização de uma série de conceitos, alguns novos, outros já conhecidos nas diversas áreas de desenvolvimento de Hardware e Software, os quais tem sido extremamente úteis, sobretudo nas fases de Especificação, Validação e Implementação de Protocolos de Comunicação.

O objetivo deste trabalho, é demonstrar como definições arquiteturais de Sistemas, baseadas em Módulos, Portas e Canais, associadas aos conceitos de Abstração e Refinamentos Sucessivos, podem ser úteis nas Especificações de Serviços, Protocolos e Interfaces de Comunicação.

Para fins didáticos, um exemplo envolvendo as especificações de Serviço e Protocolo, para o nível de Transporte OSI, é esquematizado. Neste exemplo, utiliza-se uma Técnica de Descrição Formal, baseada na "Extended State Transition Language (ESTELLE)", que está sendo desenvolvida no seio da ISO.

1. INTRODUÇÃO

No desenvolvimento de software de comunicação para sistemas distribuidos, certos problemas inerentes ao cárater repartido destes sistemas, devem ser levados em consideração:

(a) os componentes constituintes do sistema, são normalmente desenvolvidos por grupos diferentes de pessoas, trabalhando em organizações diferentes;
(b) a compatibilidade entre os componentes heterogêneos deste sistema, deve ser assegurada e
(c) a difícil compreensão do comportamento global do sistema, devido as operações paralelas, que ocorrem em seus diferentes componentes.

Os aspectos acima delineados, mostram a importância de dispor-se de uma **ESPECIFICAÇÃO** precisa dos componentes de um tal sistema, a fim de que possa servir como base para atividades subsequentes.

Durante o desenvolvimento das diversas fases (ciclo de vida) de um software distribuído, uma especificação precisa, poderá ser útil a vários propósitos. Inicialmente, serve como referência para uma cooperação entre os projetistas dos componentes do sistema. Posteriormente, poderá ser utilizada para:

(a) a **VERIFICAÇÃO** da consistência lógica do projeto;
(b) o desenvolvimento de uma **IMPLEMENTAÇÃO**;
(c) a **VALIDAÇÃO** da implementação em relação a sua especificação e
(d) o **TESTE** da implementação.

O uso exclusivo de linguagens naturais para a descrição de sistemas, embora aparentemente facilite a compreensão, leva a especificações informais, frequentemente ambiguás [BoSu 80].

Recentemente, foi reconhecido no seio da "International Standards Organization (ISO)", a necessidade de dispor-se de **TÉCNICAS DE DESCRIÇÃO FORMAL** (FDT), a fim de facilitar a tarefa de verificação e implementação de protocolos corretos e compatíveis. Como consequência, foi criado um grupo de trabalho (ISO TC97/SC16/WG1), cujo objetivo é o desenvolvimento de FDT's, a serem utilizadas em aplicações de "Open Systems Interconnection (OSI)". Tais técnicas, devem ser úteis para:

(a) o fornecimento, de um modo claro e conciso, de especificações de **SERVIÇOS, PROTOCOLOS e INTERFACES** de comunicação;
(b) a análise das especificações, tendo em vista a sua corretude, consistência mútua, eficiência, etc...;
(c) o fornecimento de suporte para o futuro desenvolvimento de implementações e
(d) certificar implementações, tendo em vista as suas conformidades às especificações.

Durante a reunião deste grupo em fevereiro de 1981, três subgrupos oram constituidos. Em linhas gerais, seus objetivos são os seguintes:

Subgrupo A: desenvolvimento de conceitos arquiteturais, de tal forma a suportar os trabalhos dos demais subgrupos [FDT A 82];
Subgrupo B: desenvolvimento de técnicas de descrição formal, baseadas em **MÁQUINA DE ESTADO FINITO EXTENDIDA** [FDT B 83] e
Subgrupo C: desenvolvimento de técnicas de descrição formal, baseadas em **ORDEM TEMPORAL DE PRIMITIVAS DE INTERAÇÃO** [FDT C 84].

O trabalho realizado pelo Subgrupo A, levou a dois conjuntos de definições. No primeiro, relacionado a arquitetura de um sistema, introduziu-se os conceitos de **MÓDULO, CANAL, PORTA, INTERAÇÃO** e **CONEXÃO** de módulos via canais e portas. No segundo grupo, a idéia, já veiculada, de divisão da especificação em serviço, protocolo e interface, foi aproveitada. A união destes dois conjuntos, deu-se através dos conceitos, ja conhecidos (por exemplo na Programação Estruturada), de **ABSTRAÇÃO e REFINAMENTOS SUCESSIVOS.**

2. MODULOS, INTERAÇÕES, CANAIS, PORTAS E CONEXÕES

A arquitetura de um sistema, é definida por um conjunto de módulos e uma estrutura de conexão (Fig.1).

Um módulo, no nível mais alto de abstração, é uma caixa preta que realiza algum processamento e representa uma unidade do sistema a ser especificada.

Tal especificação, é realizada observando-se o **COMPORTAMENTO** do módulo, sob um ângulo externo. Isto é, um modulo é definido pelas suas possibilidades de interação com o seu AMBIENTE, ou seja, com os demais módulos que compoem o sistema.

Uma interação entre dois módulos, ocorre quando ambos invocam o mesmo **TIPO INTERAÇÃO.** Um mecanismo de interação, chamado **"RENDEZ-VOUS"**, é assumido. Em outras palavras, o primeiro módulo pronto a executar a interação, deve esperar que o segundo esteja pronto.

Os módulos, sao responsáveis pela correta realização das interações. Se numa interação, por exemplo, o valor de um **PARÂMETRO** é passado, então um módulo é responsável pelo fornecimento deste valor, sendo que o outro é responsavel pelo seu recebimento.

A fim de estabelecer uma certa **SINCRONIZAÇÃO** entre os módulos, muitos métodos de especificação, assumem que o primeiro módulo a iniciar uma interação, é responsavel pela determinação dos valores dos parâmetros e este módulo deve completar a interação, antes de se dedicar a outro tipo de atividade.

Uma vez que a especificação de um **TIPO MÓDULO**, deve descrever somente o comportamento deste módulo visto do exterior, tal especificação deve definir:

(a) os possíveis tipos de interação e
(b) a **ORDEM** na qual estas interações devem ser executadas.

Esta ordem, é usualmente definida através de **REGRAS.** Estas regras podem ser expressas de diferentes formas, tais como:

(a) asserções nos valores dos parâmetros;
(b) expressões definindo as ordens, totais ou parciais, das sequências de execuções;
(c) máquinas de estado e
(d) etc...

A fim de fornecer uma certa estrutura para a conexão dos módulos e a fim de que a especificação de um módulo fôsse independente de seu ambiente, introduziu-se o conceito de canal.

Para que este último objetivo seja atingido, é necessario que os módulos incorporem exclusivamente as próprias ações. Portanto, as interações entre os módulos de um sistema e as interações destes módulos com outros, que fazem parte do ambiente deste sistema, passam a ser incorporadas pelos canais.

A definição de um **TIPO CANAL**, permite ao módulo, que utiliza este canal, considerá-lo como uma abstração do seu ambiente. Embora este ambiente possa ser implementado de diferentes maneiras, a especificação do módulo permanece a mesma.

Assim sendo, a responsabilidade das descrições das possíveis interações, antes entregue aos módulos, passa a ser dos canais.

Para facilitar a especificação das regras, que determinam a ordem na qual as interações devem ser executadas, assim como para permitir a distinção, no interior de um módulo, dos diferentes grupos de interações pertencentes aos diferentes canais utilizados por este módulo, introduziu-se o conceito de porta (ou **PONTO DE INTERAÇÃO**).

Desta forma, uma interação será identificada no módulo, pelo nome da porta na qual ela ocorre e pelo seu próprio nome, que deve pertencer a lista de interações do canal relacionado a tal porta.

Por uma porta de um módulo, pode passar um grupo de tipos de interação, cuja execução está sujeita a determinadas **REGRAS** LOCAIS. Tais regras são locais a esta porta, na medida em que independem das interações que ocorrem em outras portas deste mesmo módulo.

Assim sendo, a responsabilidade das descrições das regras locais, antes entregue aos módulos, passa a ser das portas.

Com a introdução dos conceitos de canal e porta, resta para a especificação de um módulo, se ele dispõe de mais de uma porta, a definição de algumas **REGRAS GLOBAIS**. Tais regras devem ser satisfeitas, para a execução das interações nas diferentes portas deste módulo.

Em resumo, os conceitos descritos acima, sao úteis para:

(a) o parcionamento das interações de um dado módulo, em grupos distintos, em função dos diferentes módulos que formam o ambiente do módulo

em questão;
(b) a especificação das conexões entre os diferentes módulos de um sistema. Um canal conectando dois módulos, pode ser identificado através do nome das duas portas, que se encontram nas suas extremidades e que pertencem aos módulos em questão e
(c) o refinamento de um módulo em submódulos.

3. ESPECIFICAÇÃO DE UM SERVIÇO DE COMUNICAÇÃO

Num sistema distribuido, cuja arquitetura de comunicaçao é organizada segundo diferentes níveis hierárquicos de protocolos, como por exemplo no modelo de referência OSI (Fig.2), cada nível fornece um conjunto particular de serviços, aos usuários que se encontram no nível superior [Zimm 80].

Aos usuários de um serviço, interessa apenas a descrição ordenada das entradas/saídas, passíveis de serem trocadas com o **FORNECEDOR** deste serviço, assim como a descrição dos efeitos de tais trocas.

A estas descrições, denomina-se Especificação do Serviço do Protocolo que, na realidade é baseada num conjunto de **PRIMITIVAS DE SERVIÇO.**

A especificação do serviço é abstrata, na medida em que define a forma geral das interações com seus usuários, sem entrar nos detalhes de tais interações e na medida em que define os serviços a serem oferecidos, sem especificar como fornecer tais serviços.

Em função do que foi exposto acima, pode-se delinear três partes distintas na especificação de um serviço de comunicação:
(a) a enumeração das primitivas de serviço, pelas quais os Módulos Usuários do Serviço ("Service User"), interagem com o Módulo Fornecedor do Serviço ("Service Provider");
(b) as regras locais, determinando as possíveis ordens de execução das primitivas de serviço nos Pontos de Acesso ao Serviço ("Service Access Point - SAP") e
(c) as regras globais, determinando as propriedades fim-a-fim do serviço de comunicação.

O item (a) corresponde à especificação de um canal; o item (b) corresponde à especificação das portas, pelas quais os módulos usuários interagem com o módulo servidor e o item (c) corresponde à especificação do módulo servidor.

3.1. ESPECIFICAÇÃO DE UM CANAL

A definição de um tipo canal, a qual sera utilizada posteriormente por determinadas portas de determinados módulos, deve conter:

(a) uma lista das possíveis primitivas de serviço, que podem ser invocadas através de um canal deste tipo;

(b) os nomes dos 'papéis' que os módulos, conectados às extremidades do canal, devem desempenhar (e.g., "user" e "provider") e

(c) as propriedades das primitivas de serviço, as quais devem conter:
- parâmetros, incluindo as definiçoes de seus tipos;
- uma indicaçao da extremidade que estabelece os valores dos parâmetros, incluindo as responsabilidades de fornecimento e recebimento de tais valores;
- dependências no tempo, em relação as diferentes extremidades (e.g., interações atômicas, ou interações extendidas no tempo e interruptíveis) e
- etc...

Como exemplo, pode-se especificar um tipo canal para veicular primitivas de serviço de transporte ("channel TS_primitives"). Este canal pode ser utilizado pelo sistema representado na Fig.3. Tal sistema, é constituido de um módulo servidor, chamado Transporte (nível de Transporte OSI + níveis inferiores) e de um conjunto de módulos Usuário, que interagem com o servidor através de canais do tipo TS_primitives.

Durante o desenvolvimento deste exemplo, sera utilizada uma técnica de descrição, baseada num formalismo de Máquina de Estado Finito Extendida. Portanto, cada interação será considerada atômica e qualquer atividade sujeita a interrupções, deve ser descrita ao menos por duas primitivas de interação: a primeira, correspondendo ao início da atividade e a última, correspondendo ao fim desta atividade.

No caso da definição das primitivas de serviço para o módulo Transporte, as seguintes convenções serão adotadas:

(a) a atividade de interação, pela qual um usuário requisita o estabelecimento de uma conexão, é modelada por duas interações: TCONreq e TCONconf (esta última, iniciada pelo servidor, pode ser substituida por TDISind, no caso em que o destinatário recusa o pedido de conexão);

(b) a atividade de interação, pela qual um usuário é informado da chegada de uma requisição de conexão, é modelada pelo conjunto TCONind e TCONresp (ou TDISreq, caso o usuário não aceite o pedido) e

(c) a transferência de um único bloco de dados de um usuário (tal bloco é normalmente chamado "Service Data Unit _ SDU", dispõe de um comprimento arbitrário e é interruptível), é modelada por um certo número de primitivas TDATAreq (ou TDATAind, quando o dado é entregue ao usuário pelo servidor). Cada primitiva transfere um fragmento deste bloco de dados e um parâmetro indicando se o fim do bloco foi atingido.

Para a descrição do canal TS_primitives, uma notação similar a declaração "function" em Pascal, pode ser usada:

```
type
  TS_addr_type=...;
  quality_of_TS_type=...;
  option_type=...;
  ......
  ......
channel TS_primitives(user,provider);
  by user
    TCONreq(/parâmetros/); (*pedido de conexão*)
    TCONresp(/parâmetros/);(*conexão aceita*)
    TDISreq(/parâmetros/); (*disconexão*)
    TDATAreq(/parâmetros/);(*envio de dado*)
  by provider
    TCONind(/parâmetros/); (*indicação de pedido de conexão*)
    TCONconf(/parâmetros/);(*indicação de conexão aceita*)
    TDISind(/parâmetros/); (*indicação de disconexão*)
    TDATAind(/parâmetros/);(*recebimento de dado*)
```

Nesta definição, temos oito tipos possíveis de primitivas de serviço. As quatro primeiras, são iniciadas pelo módulo que desempenha o papel de usuário e as demais pelo servidor.

Os parâmetros das primitivas, são indicados pelos seus nomes e seus respectivos tipos, os quais foram definidos previamente. No caso de TCONreq, os parâmetros a serem determinados pelo módulo, que desempenha o papel de usuário, poderiam ser:

```
TCONreq(to_TS_addr:TS_addr_type;
        proposed_options:option_type;
        proposed_QTS:quality_of_TS-type;
        ..............................
        ..............................);
```

A definição do tipo canal, da forma como foi realizada, especifica as interações que podem ocorrer através de um canal deste tipo, sem levar em consideração a ordem em que tais interações deve ocorrer.

As regras, que determinam as possíveis sequências válidas de interações, serão definidas durante as especificações das portas e do módulo fornecedor de serviço [Boch 83b].

3.2. ESPECIFICAÇÃO DAS PORTAS DE UM MÓDULO

Do ponto de vista de um observador externo, um módulo pode ser definido, num alto nível de abstração, como um conjunto de portas.

Para um módulo distribuido, a noção de conjunto de portas ,pode ser usada para distinguir diferentes pontos de interação (no espaço) deste módulo. Esta noção, também pode ser usada para distinguir, do ponto de vista do módulo, diferentes partes de seu ambiente.

Os diferentes pontos de acesso ao serviço de transporte (TSAP's), podem ser identificados através de um endereço de serviço de transporte (TS_addr) e do tipo canal associado a estes pontos.

Assim sendo, pode-se definir, numa primeira instância, os módulos representados na Fig.3, da seguinte forma:

```
module Transport_provider
       (TSAP:array[T_addr] of TS_primitives(provider));

module User_module
       (TS:TS_primitives(user));
```

A primeira especificação, define o módulo Transport_provider, como um conjunto de portas TSAP (uma para cada possível valor de TS_addr).

Cada um destes pontos de interação, esta associado ao canal tipo TS_primitives, onde o módulo em questão, desempenha o papel de servidor. Portanto, o módulo pode iniciar as primitivas TCONind, TCONconf, TDISind e TDATAind e receber as demais primitivas definidas para este canal.

É importante notar nas definições acima, que não é especificado quais módulos são interconectados. O fato de um mesmo tipo de canal ser usado por dois tipos de módulos, somente indica que 'ocorrências' destes módulos podem ser interconectadas através do canal.

Uma porta está associada a regras locais, as quais determinam as sequências de interações a serem executadas nesta porta (paragrafo 2.). Tais regras locais, podem ser derivadas de certas propriedades das interações, que são globais ao módulo.

No caso dos serviços de comunicação, as propriedades globais são frequentemente ilustradas através de diagramas de tempo, que representam as sequências das interações.

No exemplo que esta sendo desenvolvido, é assumido, para efeitos de simplificação, que um usuário, num determinado instante, pode dispor de uma única conexão de transporte.

A propriedade citada, que é global ao módulo Transporte, está implícita no diagrama de sequenciamento no tempo das interações, representado na Fig.4.

Este diagrama, descreve as interações que ocorrem em duas portas do módulo Transporte. Ele mostra, por exemplo, que uma conexão de transporte deve ser estabelecida, antes que os dados sejam trocados e esta conexão deve ser encerrada, antes do estabelecimento, através destes mesmos pontos, de uma nova conexão.

As regras globais de interação, expressas informalmente pelos diagramas de tempo, implicam em certas regras nas ordens das interações, que são locais a uma dada porta. Estas regras locais, podem ser capturadas a partir do diagrama de Máquina de Estado Finito, representado na Fig.5.

Tal diagrama, é composto de quatro **ESTADOS** e de um certo número de **TRANSIÇÕES**, as quais correspondem às interações definidas no canal tipo TS_primitives.

Uma determinada sequência de interações, só poderá ser válida, se dispuser de uma sequência equivalente de transições, associadas aos seus respectivos estados, no diagrama de Máquina de Estado.

Pode-se constatar, por exemplo, através do diagrama da Fig.5, que a sequência de interações {TCONreq ,TDISind}, só será possível, se após TCONreq, a máquina se encontrar no estado Wait_for_TCONconf.

Para a definição das regras locais aos pontos de acesso ao módulo de Transporte, duas variáveis são introduzidas:

state : (Closed, Wait_for_TCONconf, Wait_for_TCONresp, Open) e
side : (Calling, Called)

A variável "state", é considerada o **'ESTADO MAIOR'** e indica se uma conexão está estabelecida, ou está sendo estabelecida. As transições podem ser escritas, em linhas gerais, da seguinte forma:

TRANSITIONS
  FROM < estado maior atual >
  PROVIDED < condição >
  WHEN < interação de entrada >

```
TO < próximo estado maior >
BEGIN < "statements" > END;
```

Uma transição será possivel, quando o predicado habilitador, que é a conjunção das cláusulas FROM, PROVIDED e WHEN, for verdadadeiro. Em caso positivo, a transição será executada através das cláusulas TO, que atribue um novo valor ao estado maior e da cláusula BEGIN, que atualiza outras variáveis relativas ao estado do TSAP.

A partir dos estados Closed e Wait_for_TCONconf (Fig.5), as especificações das possíveis transições, para um determinado TSAP, podem ser dadas por:

```
TRANSITIONS

  FROM  Closed
    PROVIDED  no_congestion
      WHEN  TCONreq  TO  Wait_for_TCONconf
        BEGIN  side=calling  END;
      WHEN  TCONind  TO  Wait_for_TCONresp
        BEGIN  side=called  END;

  FROM  Wait_for_TCONconf
    PROVIDED  side=calling
      WHEN  TCONconf  TO  Open
        BEGIN  < statements >  END;
      WHEN  TDISind  TO  Closed
        BEGIN  < statements >  END;
```

As duas primeiras transições, indicam que a partir do estado Closed, duas interações são possíveis:

(a) se a primitiva for TCONreq e não houver congestionamento no TSAP, tal porta representará o lado que solicita a conexão, e passará ao estado Wait_for_TCONconf e

(b) se a primitiva for TCONind, a porta representará o lado solicitado e passará ao estado Wait_for_TCONresp.

No caso da sequência de interações citada {TCONreq, TDISind}, as transições primeira e última, indicam em que condições tal sequência é válida.

Para a especificação completa de um módulo, uma vez que já foram definidos os canais e suas portas, duas possíveis alternativas são:

(a) a definição do comportamento do módulo, usando uma linguagem de especificação e

(b) a definição de uma subestrutura, em termos de submódulos, sendo que cada submódulo pode ser definido ainda de acordo com (a) ou (b).

Caso a alternativa (a) seja escolhida para a definição do módulo de Transporte (Fig.3), chegar-se-á à Especificação do Serviço fornecido pelo módulo.

3.3. ESPECIFICAÇÃO DO COMPORTAMENTO DE UM MÓDULO

A especificação de um módulo, consiste da declaração de suas portas e da especificação de um comportamento global. Este comportamento deve ser respeitado para a execução das interações, uma vez satisfeitas as regras locais associadas às portas deste módulo.

Diferentes métodos podem ser usados para a especificação do comportamento global de um módulo. No caso da utilização de um formalismo baseado em Máquina de Estado Finito Extendida, a especificação constituirá basicamente:

(a) da declaração das variáveis que determinam o estado do módulo;
(b) das possíveis transições deste módulo e
(c) da definição dos procedimentos a serem utilizados pelas transições.

Para a especificação do comportamento do módulo de Transporte (Fig.3), a seguinte metodologia será usada:

(a) cada ponto de acesso ao serviço de transporte TSAP[TS_addr], está associado a um "receive-buffer", que contém o dado do usuário a ser entregue ao TSAP;
(b) uma conexão de transporte, é identificada pelos endereços dos pontos de fim de conexão, no caso "calling_address" e "called_address";
(c) as conexões existentes são gravadas numa matriz 'equivalente', chamada "peer";
(d) as transições são divididas em dois conjuntos:
   - o conjunto das transições encabeçadas pela cláusula WHEN, que são executadas quando uma determinada interação, iniciada por um determinado usuário, ocorre num determinado TSAP[TS_addr] e
   - o conjunto das transições encabeçadas pela cláusula ANY, que são espontâneas e são executadas quando as suas respectivas condições **PROVIDED**, são satisfeitas.

A estrutura de uma possível especificação do comportamento do módulo de Transporte da Fig.3, numa linguagem semelhante a "Extended Transition Language (ESTELLE)", que está sendo definida pelo Subgrupo B (ISO), é apresentada em anexo [Boch 82].

Neste anexo, a título de exemplificação, procurou-se desenvolver a fase de estabelecimento de uma conexão.

A primeira transição, é executada, quando um usuário inicia a

primitiva de entrada TCONreq, num ponto de acesso ao serviço, cujo endereço é TS_addr. Em circunstâncias normais, o pedido de conexão é gravado, através das variáveis de estado do módulo. Em caso contrário, a primitiva de saída TDISind é retornada ao usuário, indicando que o pedido de conexão não pode ser garantido.

É importante lembrar, que a troca de primitivas de interação, num ponto de acesso ao serviço, modifica o estado maior deste ponto, através da variável state, de acordo com as regras locais estabelecidas. Assim sendo, se o pedido de conexão puder ser garantido, TSAP[TS_addr].state assumirá o valor Wait_for_TCONconf.

Este fato, é utilizado para determinar quando a segunda transição da especificação, pode ser executada.

Para qualquer par de endereços (calling,called), que representa uma conexão gravada, uma segunda transição poderá ser exeutada, quando o ponto de acesso solicitante estiver no estado Wait_for_TCONconf e o ponto solicitado estiver ainda no estado Closed. Uma vez tais condições respeitadas, será enviado ao usuário solicitado, a primitiva de serviço correspondente, no caso através de TSAP[called_addr].TCONind, sendo que TSAP[called_addr].state, assumirá o valor Wait_for_TCONresp.

As duas transições seguintes, que descrevem a aceitação do pedido de conexão, são bàstante semelhantes às transições que solicitam tal pedido.

A descrição das fases de transferência de dados e encerramento de conexão, podem ser inspiradas a partir das definições efetuadas para a fase de estabelecimento de conexão. Cabe salientar, que no caso de transferência de dados, um certo número de procedimentos e funções, serão necessários para a manipulação dos receive_buffers.

Para uma especificação realmente completa do serviço oferecido pelo módulo de Transporte, as funções e os procedimetos utilizados devem ser descritos de forma mais precisa. Métodos de especificação desenvolvidos para tipos abstratos de dados, podem ser úteis a tal propósito.

4. ESPECIFICACÃO DE UM PROTOCOLO DE COMUNICAÇÃO

Ao projetista de um protocolo para o nível N (N_protocolo), interessa a estrutura interna deste protocolo. Num ambiente rede, com usuários fisicamente separados, a concepção de tal protocolo, deve ser realizada tendo em mente o aspecto distribuido do sistema, ou seja, através de

ENTIDADES (processos ou módulos) locais a cada usuário.

Uma entidade, relativa a um usuário, comunica-se com uma entidade equivalente ("peer_entity"), relativa a outro usuário que se encontra no mesmo nível do primeiro, através dos serviços fornecidos pelos níveis inferiores.

As descrições das operações das entidades contidas no N_nível, em resposta aos comandos fornecidos pelos seus usuários, em resposta às mensagens provenientes de entidades equivalentes (que chegam via serviços do N-1_nível) e em resposta as ações iniciadas internamente (e.g.,"time-outs"), constituem a Especificação do Protocolo.

A especificação do protocolo, pode ser encarada como um refinamento do serviço. Cada entidade é definida, até o ponto em que a compatibilidade com as outras entidades equivalentes, é assegurada (pode-se utilizar aqui, o termo Protocolo Abstrato).

Futuros refinamentos, podem levar a implementação de cada entidade propriamente dita. Isto é realizado, através da codificação da entidade numa linguagem de programação [Boch 84a].

4.1. DEFINIÇÃO DE UMA SUBESTRUTURA

A especificação de um módulo, pode ser dada definindo-se uma subestrutura, baseada no refinamento deste módulo num conjunto de submódulos (parágrafo 3.2.).

Uma típica subestrutura para o módulo Transporte (Fig.3), é apresentada na Fig.6. Tal subestrutura, consiste de:

(a) um submódulo Rede (nível rede OSI + níveis inferiores), que fornece os serviços de comunicação e
(b) um conjunto de submódulos Entidade, possivelmente localizados em lugares diferentes, cada um executando o protocolo de transporte.

A fim de que uma conexão rede, seja utilizada por mais de uma conexão de transporte, um certo refinamento nos pontos de acesso ao serviço de transporte, é realizado. Assim sendo, um TSAP corresponderá a um conjunto de pontos de fim de conexão de transporte (TCEP) [ISO 83].

Para a identificação dos diferentes TCEP's e consequentemente de seus usuários, a seguinte convenção hierárquica é utilizada:

(a) cada Entidade é associada a um unico ponto de acesso ao submódulo

Rede (NSAP), o qual é identificado por um endereço rede (NS_addr);

(b) um TCEP é identificado por um endereço de transporte (T_addr), que é composto de dois segmentos:
- um prefixo rede (N_prefix), correspondendo ao NS_addr do submódulo Entidade, que fornece serviço ao TSAP e
- um sufixo transporte (T_suffix), que identifica o TCEP no interior do TSAP e consequentemente seu usuário.

Utilizando as convenções descritas acima, os tipos submódulos, que constituem a subestrutura representada na Fig.6, podem ser definidos, através das suas portas, da seguinte forma:

```
module Network_provider
       (NSAP:array[NS_addr] of NS_primitives(provider));

module TP_entity
       (NS:NS_primitives(user);
        Timer:Timer_service(user);
        TCEP:array[T_suffix] of TS_primitives(provider));
```

Nas especificações até aqui realizadas, o canal tipo TS_primitives, foi utilizado pelo módulo tipo TS_provider e pelos submódulos tipo TP_entity. Este mesmo canal, poderá ser utilizado na especificação do protocolo do nível Sessão, pois uma entidade executando tal protocolo, é um usuário do serviço fornecido pelo nível Transporte.

O fato acima citado, demonstra a vantagem de se especificar as propriedades dos canais, separadamente dos módulos. Este procedimento, permite também verificar, se há consitência lógica entre as diferentes fases de especificação de um módulo. Para tal, basta verificar em cada etapa, se o tipo de canal que interconecta dois módulos, é o mesmo

A especificação do protocolo de transporte, para a subestrutura representada na Fig.6, é constituida, na realidade, da descrição das regras que definem o comportamento dos submódulos tipo TP_entity. Novamente aqui, duas alternativas sao possíveis: a descrição do comportamento do submódulo Entidade, numa linguagem de especificação, ou realizar o refinamento deste submódulo.

4.2. REFINAMENTO RECURSIVO DE UM MÓDULO

E possível definir um módulo, através de um processo recursivo. No caso do submódulo Entidade, uma possível subdivisão é representada na Fig.7.

Esta nova subestrutura, é constituida de dois submódulos: [Boch

83a:

(a) um submódulo Protocolo Abstrato (AP), o qual define o comportamento lógico da entidade, determinando os tipos de Unidades de Dados de Protocolo (PDU's), a serem trocados com a entidade remota equivalente e

(b) um submódulo Mapeamento ("Mapping"), responsável pela codificação e decodificação dos PDU's, que são trocados através dos serviços oferecidos pelo submódulo Rede.

Estes dois submódulos, interagem entre si trocando PDU's, os quais podem ser definidos, através do tipo canal PDU_and_control, descrito abaixo:

```
type
  reason_type=...
  .......
  .......
channel PDU_and_control(user,provider);
  by user,provider
    CR(/parâmetros/); (*PDU de requisição de conexão*)
    CC(/parâmetros/); (*PDU de confirmação de conexão*)
    DR(/parâmetros/); (*PDU de requisição de disconexão*)
    DC(/parâmetros/); (*PDU de confirmaçao de disconexão*)
    DT(/parâmetros/); (*PDU de dados*)
```

Assim como foi procedido com os módulos anteriores, estes submódulos podem ser descritos, através de suas portas:

```
module AP
      (Map:PDU_and_control(user);
       Timer: Timer_service(user);
       TCEP:array[T_suffix] of TS_primitives(provider));

module Mapping
      (NS:NS_primitives(user);
       P:PDU_and_control(provider));
```

Para uma especificação completa do módulo AP, caso uma linguagem de descrição seja utilizada, as regras que definem o comportamento deste módulo, podem ser captuaradas a partir do diagrama de Máquina de Estado Finito, representado na Fig.8 [Boch 84b].

Novos refinamentos podem ser empregados, propiciando um nível de detalhamento cada vez maior, da estrutura interna de um módulo. Tal procedimento, poderá levar a realização do módulo em hardaware, ou de sua implementação em software.

Se a alternativa de software, for a escolhida para a realização de um módulo, é interessante, que ao ser atingido um certo nível de detalhamento, a descrição da subestrutura, seja feita utilizando-se uma linguagem de programação. Tal descrição, pode ser usada para implementar diretamente os módulos, que foram obtidos nas fases anteriores.

5. ESPECIFICAÇÃO DE UMA INTERFACE DE COMUNICAÇÃO

Durante as diversas fases de especificação de um mesmo módulo, foi assumido que tal módulo, sempre interage com seu ambiente, através do mesmo conjunto de primitivas.

Estas primitivas, definidas de forma abstrata através de um tipo canal, representam o elo de comunicação entre dois módulos, os quais também são definidos abstratamente, através de suas portas.

Entretanto, as implementações destes módulos, irão comunicar-se entre si, através de 'interações reais' (hardware ou software), realizadas por uma 'interface real'. Normalmente, uma interface é prevista para cada ponto de interação (Fig.9).

A fim de chegar-se a implementação de tais interfaces, faz-se necessário definir o formato e os mecanismos exatos das interações, os quais podem variar de um usuário a outro. Esta definição, é que constitue a Especificação da Interface de Comunicação.

Um processo de refinamentos sucessivos dos canais e das portas de um módulo, pode levar a definição das interfaces. Durante este procedimento, duas questões devem ser levadas sempre em consideração:

(a) se uma interação, que corresponde a uma primitiva num nível abstrato de especificação, é constituida de várias subinterações num nível mais detalhado de especificação e

(b) se uma dada porta, é composta de varias subportas. Neste caso, uma interação pode envolver várias subinterações, em várias subportas.

A principal aplicação do refinamento de portas e canais, no desenvolvimento de software de comunicação, é a especificação e implementação das interfaces, entre os diferentes níveis hierárquicos de protocolos.

Para facilitar a tarefa de especificação de serviços e protocolos, as interações são definidas, nesta primeira fase, em termos das primitivas que são trocadas entre os níveis. Isto corresponde à definição de uma Interface Abstrata.

Durante a fase de implementação de um protocolo, uma incumbência difícil e das mais importantes, é a realização das interfaces, em termos de chamadas a procedimentos, ou primitivas de comunicação no interior dos processos, ou qualquer outra construção em software ou hardware.

Nesta última fase, a especificação das interfaces num nível mais detalhado, pode ser extremamente útil, na medida em que o refinamento dos canais e portas, leva a uma definição mais precisa das interações que ocorrem entre os módulos.

6. CONCLUSÃO

Especificações de protocolos, além de servir como referência para o trabalho dos projetistas, envolvidos na concepção de sistemas hierárquicos, são bastante úteis em fases posteriores de desenvolvimento destes protocolos, tais como, validação, implementação e teste.

A divisão da especificação em serviço, protocolo e interface, diminue o grau de dificuldade desta primeira fase, na medida em que permite uma melhor compreensão do comportamento global de um sistema. Tal divisão, favorece tambem as fases posteriores. A especificação do serviço, por exemplo, pode ser usada para verificar a conformidade da especificação do protocolo. A especificação do protocolo, por sua vez, pode ser usada na fase de validação e num ambiente de teste.

A definição de um sistema baseada em módulos, portas e canais, fornece um suporte, em termos de arquitetura, para um desenvolvimento sistemático das especificações. Tais noções, aliadas aos conceitos de abstração e refinamentos sucessivos, permitem aumentar o grau de complexidade das especificaçoes gradativamente.

Uma condição necessária, para que uma especificação cumpra os objetivos acima delineados, é a sua precisão. O uso de técnicas de descrição formal, para as especificações de serviços, protocolos e interfaces de comunicação, satisfaz tal requisito, na medida em que elimina as ambigüidades criadas pelas especificações informais.

7. REFERÊNCIAS

[Boch 82] : G.V. Bochmann, "A Formal Description Technique for Distributed Systems", documento interno (cap.X), a ser publicado na forma de livro, I.R.O. Université de Montréal, Dez 82.

[Boch 83a] : G.V. Bochmann, "Example of a Transport Protocol Specification", documento de contribuição ao encontro do grupo ISO/TC97/SC16/WG1, I.R.O. Université de Montréal, Fev 83.

[Boch 83b] : G.V. Bochmann, "Formal Description Techniques for OSI: an Example", publicação interna No 493, I.R.O. Université de Montréal, Nov 83, apresentado no INFOCOM'84.

[Boch 84a] : G.V. Bochmann, "A Specification of a Transport Protocol", documento interno (cap.VV), a ser publicado na forma de livro, I.R.O. Université de Montréal, Fev 84.

[Boch 84b] : G.V. Bochmann, "Introduction to the Specification Language", documento interno (cap.XX), a ser publicado na forma de livro, I.R.O. Université de Montréal, Fev 84.

[BoSu 80] : G.V. Bochmann, C.A. Sunshine, "Formal Methods in Communication Protocol Design", IEEE Trans. COM-28, No 4, Abr 80, pp. 624-631.

[FDT A 82] : ISO/TC97/SC16 N, "Concepts for Describing the OSI Architecture", Nov 82.

[FDT B 83] : ISO/TC97/SC16 N1347, "A FDT based on an Extended State Transition Model", Jul 83.

[FDT C 84] : ISO/TC97/SC16/WG1 N, "Definition of the Temporal Ordering Specifiction Language LOTOS", Maio 84.

[ISO 83] : ISO 7498_1983(E), "Information Processing Systems_Open Systems Interconnection_Basic Reference Model", 83.

[Zimm 80] : H. Zimmermann, "OSI Reference Model_The ISO Model of Architecture for Open Systems Interconnection", IEEE Trans. COM-28, No 4, Abr 80, pp. 425-432.

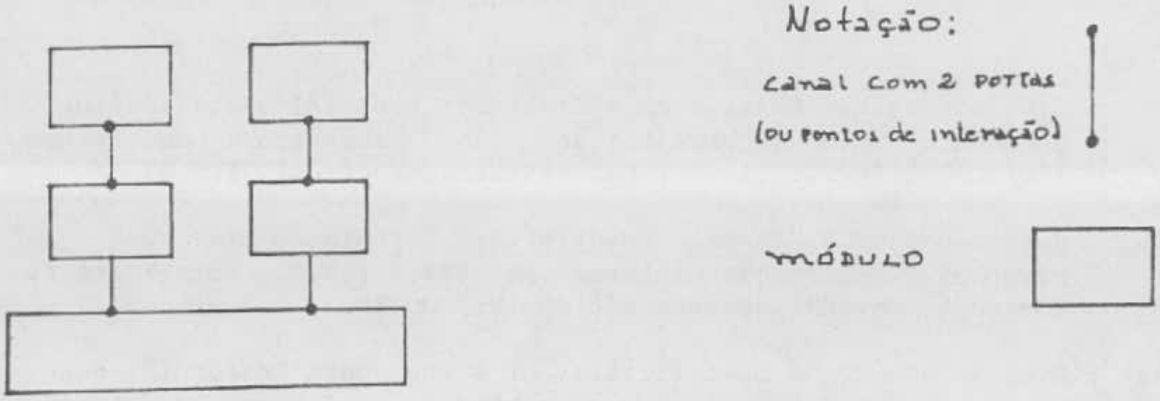

Fig. 1 Módulos, Canais e Pontos de Interação

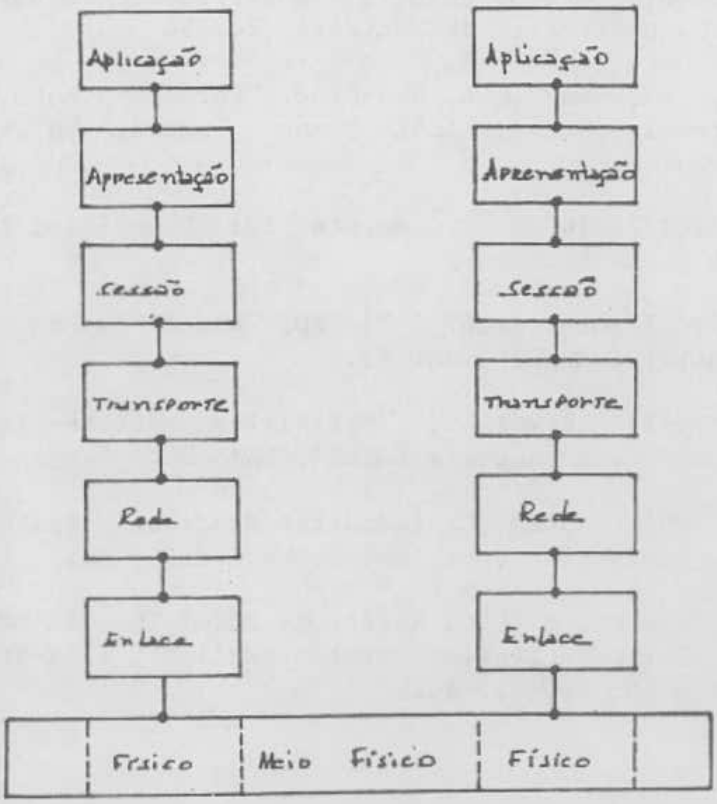

Fig. 2 Modelo de Referência OSI

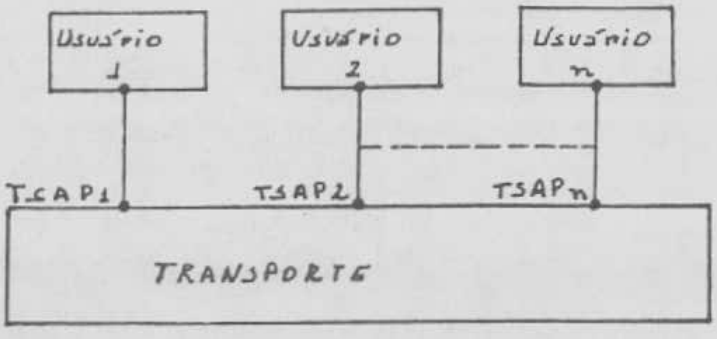

Fig. 3 Sistema constituído de 01 módulo Transporte e n Usuários

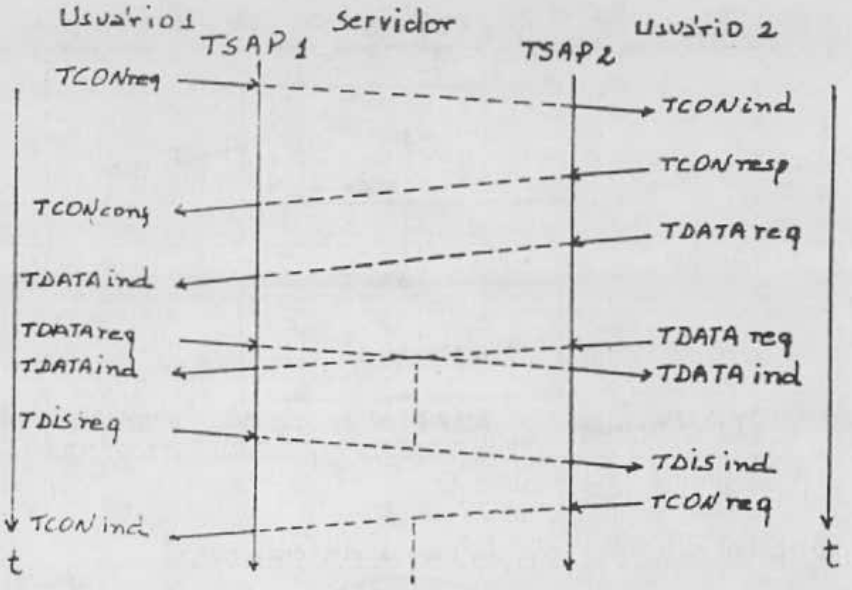

Fig. 4 Diagrama da sequência no tempo das primitivas de serviço de Transporte

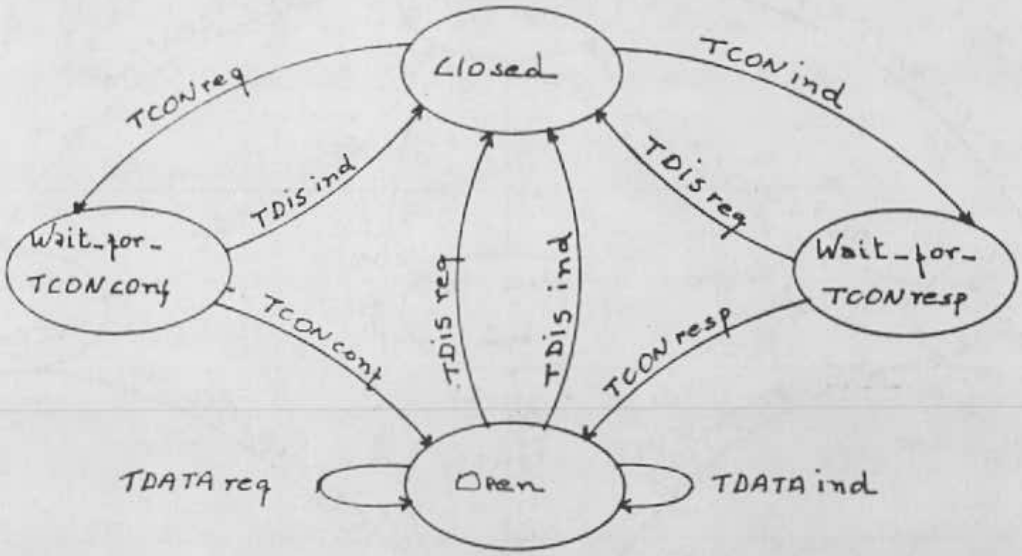

Fig. 5 Máquina de Estado Finito para um TSAP

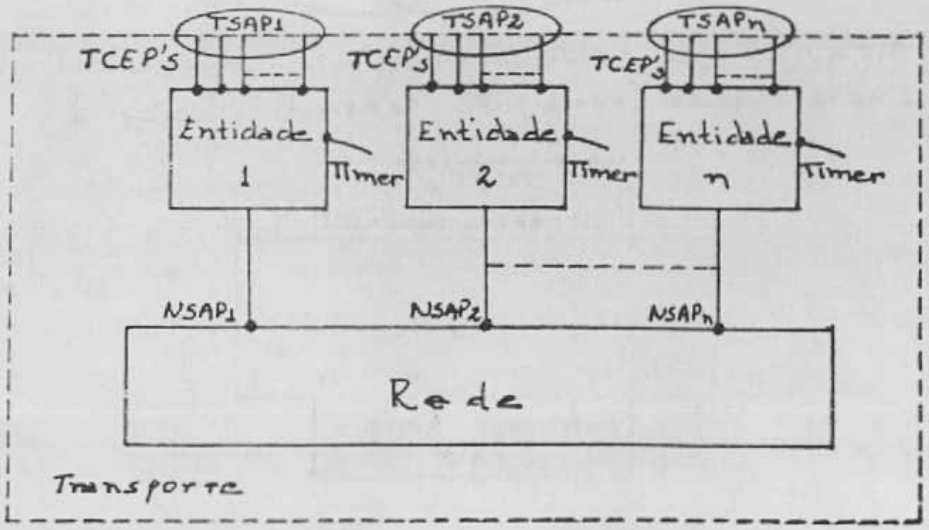

Fig. 6 Subestrutura para o módulo Transporte

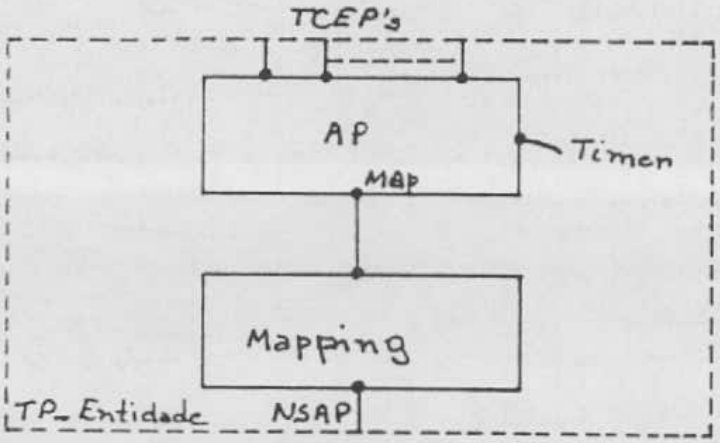

Fig. 7 Subdivisão do módulo Entidade

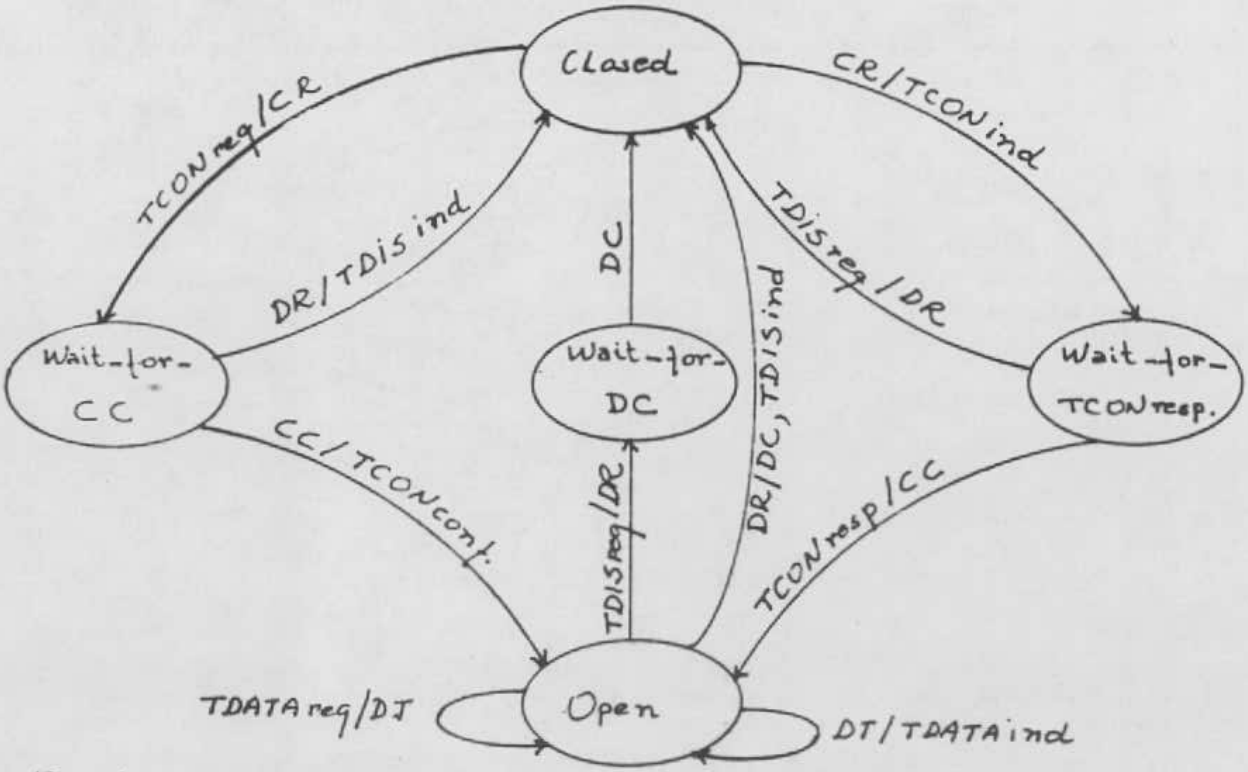

Fig. 8 Máquina de Estado Finito para o módulo AP
(notação das transições: "entrada" / "saída")

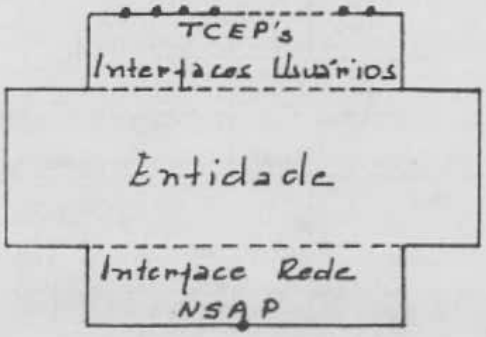

Fig. 9 Interfaces de Comunicação do módulo Entidade

```
module Transport_provider
      (TSAP:array[TS_addr] of TS_primitives(provider));

type
  buffer_type=...;
  TS_disconnect_reason_type=...;
  ........
  ........;

var
  receive_buffer:array[TS_addr] of buffer_type;
  options:array[TS_addr] of option_type;
  TDIS_reason:array[TS_addr] of TS_disconnect_reason_type;
  peer:array[TS_addr] of TS_addr_type;
  ........
  ........;

procedure record_connection(calling_addr,called_addr:TS_addr_type);
  begin
    peer[calling_addr]:=called_addr;
    peer[called_addr]:=calling_addr;
    ........
    ........;
  end

(*FASE DE ESTABELECIMENTO DE CONEXÃO*)
  ........
  ........
end.

when TSAP[TS_addr].TCONreq(/parâmetros/)
  begin
    if ... (*possível fornecer serviço*)
      then begin
           record_connection(TS_addr,to_TS_addr);
           options[TS_addr]:=proposed_options;
           clear (TSAP[TS_addr].receive_buffer);
           clear (TSAP[to_TS_addr].receive_buffer)
           end
      else TSAP[TS_addr].TDISind(/parâmetros/);

(*quando o pedido de conexão atingir o lado solicitado*)

any calling_addr,called_addr:TS_addr_type do
  provided peer[calling_addr]=called_addr
       and TSAP[calling_addr].state=Wait_for_TCONconf
       and TSAP[called_addr].state=closed
    begin
      TSAP[called_addr].TCONind(/parâmetros/)
```

```
(*resposta do lado solicitado*)

when TSAP[TS_addr].TCONresp(/parâmetros/)
  provided proposed_options in options[peer[TS-addr]]
    begin
      options[peer[TS_addr]]:=proposed_options
    end;

(*a resposta ao pedido de conexão atinge o solicitante*)

any calling_addr,called_addr:TS_addr_type do
  provided peer[calling_addr]=called_addr
       and TSAP[calling_addr].state=Wait_for_TCONconf
       and TSAP[called_addr].state=Open
    begin
      TSAP[calling_addr].TCONconf(/parâmetros/)
    end;

(*FASE DE TRANSFERÊNCIA DE DADOS*)

when TSAP[TS_addr].TDATAreq(/parâmetros/)
  provided ........(*controle de fluxo*)
    begin
      append(receive_buffer[peer[TS_addr]],........)
    end;

(*FASE DE ENCERRAMENTO DE CONEXÃO*)
```

Estrutura para Especificação do Módulo de Transporte da Fig.3