Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Telemática

Especificação formal utilizando projeto orientado a objeto e SDL'92 como alternativa para a evolução de protocolos de interconexão de redes locais e ATM

Jhuli Meire Takahara

Banca Examinadora:
Dr. Walter da Cunha Borelli (orientador)
Dr. Mario A. Nascimento
Dr. Maurício Ferreira Magalhães
Dr. Akebo Yamakami(suplente)

Este exemplar corresponde a redação final da tese defendida por Jhuli Meire Takahara e aprovada pela Comissão Julgada em 13 / 02 / 98.
Orientador

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas - FEEC-UNICAMP, como parte dos requisitos exigidos para a obtenção do título de MESTRE EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO.

13 de Fevereiro de 1998

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

T139e

Takahara, Jhuli Meire
Especificação formal utilizando projeto orientado a objeto e SDL'92 como alternativa para a evolução de protocolos de interconexão de redes locais e ATM. / Jhuli Meire Takahara.-- Campinas, SP: [s.n.], 1997.

Orientador: Walter da Cunha Borelli
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Redes de computação - Protocolos. 2. Redes locais de computação. 3. Simulação (Computadores digitais). 4. Programas de computador (Validação). 5. Engenharia de software. I. Borelli, Walter da Cunha. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Aos meus pais, amigos e em especial ao Cláudio.

Agradecimentos

Ao Prof. Dr. Walter da Cunha Borelli, pela sua orientação, dedicação e paciência durante a elaboração dessa tese.

Aos meus pais, que acima de tudo sempre me incentivaram, ao Cláudio pela paciência, pelo incentivo, e por sempre estar presente até o fim desse trabalho, e a todos os amigos que conheci durante esse período de mestrado.

Aos meus colegas do DT.

Aos funcionários do DT e CPG, em especial à Flávia e ao Jesus.

Ao CNPq pela bolsa de estudo, indispensável para o desenvolvimento desse trabalho.

A Telelogic, que doou as licensas utilizadas, que sem elas, não seria possível desenvolver esse trabalho.

Finalmente, a todos aqueles que contribuiram direta ou indiretamente para a realização desse trabalho.

Resumo

Nesta tese são apresentadas a especificação formal, validação e simulação dos protocolos IP Clássico e NHRP (Next Hop Resolution Protocol), de interconexão de redes locais e ATM segundo recomendações do IETF (Internet Engineering Task Force). É proposta uma metodologia para construção de especificações formais que combina a técnica de orientação a objetos - OMT (Object Modeling Technique) com a potencialidade da linguagem SDL (Specification and Description Language) padronizada pelo ITU-T (antigo CCITT), com o propósito de evoluir os protocolos especificados. Os sistemas referentes aos protocolos e as suas evoluções foram especificados, validados e simulados utilizando a ferramenta CASE SDT* (SDL Design Tool , Telelogic-Suécia).

Palavras Chaves: Redes de computadores, Protocolos de Comunicação, Especificação Formal, Rede ATM, Redes Locais, TCP/IP, OMT - Object Modeling Technique, SDL - Specification and Description Language, MSC - Message Sequence Chart, Validação, Simulação, Projeto Orientado a Objeto.

Publicação:

J.M. Takahara e W.C. Borelli, "Especificação de Protocolos de Interconexão de Redes Locais e ATM utilizando OMT e SDL'92", *Anais do XV Simpósio Brasileiro de Telecomunicações*, Recife - Pernambuco, Brasil, 145-149, 1997.

---

* Pacote SDT, versão 3.02 (*SDT Base, MSC Editor, Validator e Simulator*): adquirido pelo DT/FEEC/UNICAMP através do Projeto Temático [FAPESP] (Proc. 91/3660-0).

Abstract

This thesis presents a formal specification, validation and simulation of IP over ATM and NHRP (Next Hop Resolution Protocol) for the internetworking of LANs and ATM based upon IETF's recomendation (Internet Engineering Task Force). It is proposed a methodology to construct formal specifications that combines the object-oriented technique OMT (Object Modeling Technique) and the potenciality of SDL (Specification and Description Language) with the purpose of developing and evolving the specified protocols. The systems concerning the protocols and their evolutions were specified, validated and simulated using the SDT* (SDL Design Tool, Telelogic - Sweden) tool CASE.

Key Words: Computer Networks, Communication Protocols, Formal Specification, ATM Networks, Local Networks, TCP/IP, OMT - Object Modeling Technique, SDL - Specification and Description Language, MSC - Message Sequence Chart, Validation, Simulation, Object-Oriented Design.

Publication:

J.M. Takahara e W.C. Borelli, "Especificação de Protocolos de Interconexão de Redes Locais e ATM utilizando OMT e SDL'92", *Anais do XV Simpósio Brasileiro de Telecomunicações*, Recife - Pernambuco, Brasil, 145-149, 1997.

* SDT package, version 3.02 (*SDT Base, MSC Editor, Validator and Simulator*): purchased by DT/FEEC/UNICAMP via FAPESP Thematic project (Proc. 91/3660-0).

Indice

Capítulo 1

Introdução 1
1.1- Introdução 1
1.2 - Organização do trabalho 3

Capítulo 2

Arquiteturas TCP/IP e ATM 5
2.1 - Introdução 5
2.2- Arquitetura TCP/IP 6
2.3- Arquitetura ATM 10
2.4 - IETF e ATM Forum 17
2.4.1 - IETF (Internet Engineering Task Force) 18
2.4.1.1 - IP over ATM ou IP Clássico 18
2.4.1.2 - NHRP (Next Hop Resolution Protocol) 22
2.4.1.3 - MARS (Multicast Address Resolution Server) 25
2.4.2 - ATM Forum 26
2.5 - Conclusão 27

Capítulo 3

Sistemas especificados a partir dos Protocolos IP Clássico e NHRP 28
3.1 - Introdução 28
3.2. - Metodologia: OMT + SDL'92 28
3.2.1 - Modelo de objetos em OMT 33
3.3 - Especificação em SDL'92 37
3.3.1 - Sistema Cloud 38
3.3.2. - Sistema IP Clássico 51
3.3.3 - Sistema NHRP 59
3.3.4 - Sistema NHRP que se comunica através do protocolo Proxy-ARP 63
3.3.5 - Sistema IP Clássico Migrado para NHRP 68
3.3.6 - Sistema IP Clássico Migrado e o NHRP 70
3.4 - Comentário sobre as Especificações 72

Capítulo 4

Resultados das Validações e Simulações 73
4.1 - Introdução 73
4.2 - Validação 74
4.3 - Simulação 79
4.2 - Conclusão 85

Capítulo 5

Conclusão 87

Indice Remissivo *90*

Siglas e Acrônimos 92

Referências Bibliográficas _____ *95*

Apêndice A

Linguagem SDL (Specification and Description Language) _____ **102**
A.1 - Introdução _____ 102
A.2 - SDL'88 _____ 103
A.3 - SDL'92 _____ 104
A.4 - Notação SDL _____ 107

Apêndice B

SDT (SDL Desing Tool) _____ **111**

Apêndice C _____ *116*

Especificações em SDL _____ **116**
C.1 - Sistema Cloud _____ 116
C.2 - Sistema IP Clássico _____ 138
C.3 - Sistema NHRP _____ 140
C.4 - Sistema NHRP que se comunica através do protocolo Proxy-ARP _____ 142
C.5 - Sistema IP Clássico migrado para NHRP _____ 143
C.6 - Sistema IP Clássico migrado que se comunica a uma nuvem NHRP _____ 144

Indice de Figuras

Figura 2. 1 - Modelo de Referência TCP/IP [51] 7
Figura 2. 2 - Camadas da Arquitetura Internet [10] 7
Figura 2. 3 - Formato do endereço IP [51] 9
Figura 2. 4 - Endereçamento de subrede na classe B de rede [51] 10
Figura 2. 5 - Estrutura de uma célula ATM [44] 11
Figura 2. 6 - Estrutura em camadas de uma rede ATM [44] 12
Figura 2. 7 - Interface NNI e UNI de Switches ATM [3] 16
Figura 2. 8 - Métodos de interconexão ATM [1] 18
Figura 2. 9 - Roteamento entre LISes e o modelo de IP Clássico [1] 20
Figura 2. 10 - Roteamento inter LIS no protocolo IP Clássico [3] 20
Figura 2. 11 - Comunicação entre Host e Servidor ARP do RFC 1577 [3] 21
Figura 2. 12 - Requisição NHRP pode ser roteado através de mais de um Servidor NHRP [3] 23
Figura 2. 13 - Operação do protocolo NHRP - Next Hop Resolution Protocol [3] 24
Figura 2. 14 - Mecanismo de funcionamento de um Servidor Multicast [3] 26
Figura 3. 1 - Modelo de objeto utilizado para especificação de sistemas 35
Figura 3. 2 - Hierarquia de herança entre os Sistemas Especificados 37
Figura 3. 3 ADT utilizado na representação de endereço IP 38
Figura 3. 4 - Package do Sistema Cloud 39
Figura 3. 5 - Sistema que utiliza o Package para compilar o Sytem Type Cloud 40
Figura 3. 6 - Declaração dos sinais utilizados no Sistema Cloud 41
Figura 3. 7 - Especificação em SDL'92 do Sistema Cloud. 42
Figura 3. 8 - Bloco Tab_Ini do Sistema Cloud 43
Figura 3. 9 - Block Type Cache do Sistema Cloud 44
Figura 3. 10 - Bloco Host do Sistema Cloud 45
Figura 3. 11 - Bloco Subnetwork do Sistema Cloud 46
Figura 3. 12 - Bloco Servidor do Sistema Cloud 47
Figura 3. 13 - Processo Servidor do Sistema Cloud 48
Figura 3. 14 - Processo Router do Sistema Cloud 49
Figura 3. 15 - Procedimento Connect_Request do Sistema Cloud 50
Figura 3. 16 - Hierarquia da Especificação do Sistema IP Clássico 52
Figura 3. 17 - Sistema IP Clássico 53
Figura 3. 18 - Bloco Cache do Sistema IP Clássico 54
Figura 3. 19 - Bloco Host do Sistema IP Clássico 55
Figura 3. 20 - Bloco Server do Sistema IP Clássico 55
Figura 3. 21 - Bloco Subnetwork do Sistema IP clássico 56
Figura 3. 22 - Processo subnetwork do Sistema IP Clássico 57
Figura 3. 23 - Processo Servidor do Sistema IP Clássico 58
Figura 3. 24 - Hierarquia do Sistema NHRP especificado 59
Figura 3. 25 - Descrição do Sistema NHRP 60
Figura 3. 26 - Bloco Servidor NHS do Sistema NHRP 61
Figura 3. 27 - Bloco Subrede redefinido do Sistema NHRP 62
Figura 3. 28 - Hierarquia do Sistema NHRP que se comunica através do protocolo Proxy-ARP 63
Figura 3. 29 - Meio compartilhado Proxy-ARP de uma única nuvem 64
Figura 3. 30 - Sistema NHRP que se comunica através do protocolo Proxy-ARP 65
Figura 3. 31 - Bloco tab_adr que gerencia os endereços de duas Nuvens NHRP 67
Figura 3. 32 - Hierarquia do Sistema IP Clássico Migrado para o NHRP 68
Figura 3. 33 - Sistema IP migrado para NHRP 69
Figura 3. 34 - Sistema IP migrado para NHRP 71
Figura 3. 35 - Hierarquia do sistema que realiza a comunicação entre IP Clássico migrado e o Sistema NHRP 72
Figura 4. 1 - Resultado do Bit State Exploration do Sistema IP Clássico 76
Figura 4. 2 - Resultados da Validação (Random Walk e Bit State Exploration) do Sistema Cloud 76
Figura 4. 3 - Resultado do Bit State Exploration do Sistema NHRP 77
Figura 4. 4 - Resultado apresentado pelo Sistema NHRP que se comunica através do protocoloque se Proxy-ARP 77

Figura 4. 5 - Resultado do Sistema IP Clássico Migrado para o NHRP 77
Figura 4. 6 - Resultados do Bit State Exploration do Sistema IP Clássico Migrado e o NHRP 77
Figura 4. 7 - Parte do Coverage View do Sistema Cloud 79
Figura 4. 8 - Parte do Coverage View do Sistema IP Clássico 80
Figura 4. 9 - Configuração de rede utilizada para simulação e validação dos sistemas 81
Figura 4. 10 - Caso 1 - estabelecimento de conexão 83
Figura 4. 11 - Caso 2 - perda de sinal 83
Figura 4. 12 - Caso 3 - Correção de especificação 84
Figura 4. 13 - Exemplo de funcionamento do Sistema IP Clássico 85
Figura 4. 14 - Loop detectado no funcionamento do Sistema IP Clássico 85
Figura 4. 15 - Exemplo de funcionamento do Sistema NHRP 86
Figura A. 1 - Estrutura de Especificação em SDL 104
Figura A. 2 - Exemplo de conexão N-M 107
Figura A. 3 - Exemplo do uso do any e de Procedimento que retorna valor 107
Figura A. 4 - Símbolos do SDL [46] 110
Figura B. 1 - Visão geral do Funcionamento do SDT [58] 113
Figura B. 2 - Metodologia SOMT [54] 115
Figura C.1 - Process Type Cache_Consulting (1) 117
Figura C.2 - Process Type Cache_Consulting (2) 118
Figura C.3 - Virtual Process Type Host 119
Figura C.4 - Virtual Process Type Sendproc (1) 120
Figura C.5 - Virtual Process Type Sendproc (2) 121
Figura C.6 - Process HostInit 122
Figura C.7 - Process Type Router (1) 123
Figura C.8 - Virtual Procedure Search Hop 124
Figura C.9 - Virtual Process Type Server (1) 125
Figura C.10 - Virtual Procedure Arep 126
Figura C.11 - Virtual Procedure Areq 127
Figura C.12 - Virtual Procedure Crep 128
Figura C.13 - Process Serinit 129
Figura C.14 - Virtual Process Tyep Subnetwork 130
Figura C.15 - Process SubNinit 131
Figura C.16 - Process Inial_ta 132
Figura C.17 - Procedure InitHost 133
Figura C.18 - Procedure InitPrinTab 134
Figura C.19 - Procedure Initroutab 135
Figura C.20 - Procedure InitServ 136
Figura C.21 - Procedure InitServ 137
Figura C.22 - Procedure Init Sub 138
Figura C.23 - Process Type Cache Atualize 139
Figura C.24 - Redefined Procedure Arep 140
Figura C.25 - Redefined Process Type Router 141
Figura C.26 - Redefined Process Type Server 142
Figura C.27 - Package do Bloco Cache 143
Figura C.28 - Block Type Ccache 143
Figura C.29 - Processo servidor que herda o package do Processo Servidor do NHRP 144
Figura C.30 - Process ICA (1) 145
Figura C.31 - Process ICA (2) 146

Indice de Tabelas

Tabela 2. 1- Esquematização das classes de serviços [51] ____ *13*
Tabela 3. 1 - Diagrama de migração de OMT para SDL'92 ____ *33*
Tabela 3. 2- Mapeamento entre o modelo OMT e SDL para Sistema Cloud. ____ *38*
Tabela A. 1 - Tabela de equivalência de conceitos orientados a objeto e SDL'92 [58] ____ *108*

Capítulo 1

Introdução

1.1- Introdução

Com o crescente desenvolvimento da tecnologia digital, criou-se a necessidade de transmitir diferentes tipos de informação, como texto, áudio, vídeo e voz; dando origem aos sistemas multimídia, necessitando-se assim, desenvolver arquiteturas de redes de alta velocidade. Contudo, os sistemas de comunicação foram desenvolvidos para o transporte de tipos de informações específicas. Como solução de integração de transmissão de diferentes informações em uma única rede, foi proposto o conceito de redes de serviços integrados (Integrated Services Digital Network).

A tecnologia de rede ATM(Asynchronous Transfer Mode) foi escolhida como solução de tráfego da B-ISDN (Broadband - Integrated Services Digital Network) [44], por apresentar diversas vantagens como, alta velocidade, escalabilidade, potencial para criar LANs (Local Area Networks) virtuais e principalmente pela habilidade de suportar interconexão de redes com qualidade de serviço (QoS -Quality of Service). Apesar da tecnologia ATM ter sido escolhida e padronizada para redes públicas integradas como a ISDN, essa tecnologia tem despertado interesse da comunidade como uma solução para a interconexão de redes de alta velocidade em ambientes locais.

O estudo para viabilizar a interconexão de redes locais que operam o protocolo TCP/IP (Transmission Control Protocol/Internet Protocol) em um meio ATM ainda tem sido discutido em proposições apresentadas pelo IETF (Internet Engineering Task Force) e o ATM Forum. O IETF é um órgão que padroniza os protocolos que serão executados na rede Internet e o ATM Forum é o órgão de padronização de protocolos na rede ATM. Ambos os órgãos possuem diferentes propostas para a interconexão de suas arquiteturas. O IETF propõe o uso de protocolos do modo nativo que realizam a comunicação diretamente entre as camadas ATM e IP, como o IP Clássico ou IP over ATM [20] [28] [32], NHRP (Next Hop Resolution Protocol) [21] [30] e MARS (Multicast Address Resolution Server) [23] [24]. O ATM Forum por sua vez propõe o uso de protocolos no modo *overlay*, que utiliza a camada de acesso ao meio como uma interface entre as

camadas ATM e IP, como no caso dos protocolos LANE(LAN Emulation) [5] [6] [7] e MPOA (Multi Protocol Over ATM) [4].

O problema de interconexão de duas arquiteturas diferentes como as redes ATM e TCP/IP, possui várias áreas de estudo para promover a melhor interoperabilidade entre elas. A não existência de um padrão estabelecido para implementação dos protocolos IP Clássico[20] [28] [32] e NHRP (Next Hop Resolution Protocol) [21] [30] propostos pelo IETF, motivou à construção de especificações formais desses protocolos com o objetivo de melhor entendê-las e analisar suas funcionalidades.

Neste trabalho, baseando-se nas recomendações propostas pelo IETF dos protocolos IP Clássico e NHRP, e após a apresentação de alguns aspectos relativos à implementação dos elementos envolvidos no estabelecimento de conexão de redes locais para a troca de mensagens, é proposta uma combinação da técnica de orientação a objeto OMT [45] e a versão de 1992 da linguagem formal SDL (Specification and Description Language) [36] [37] para construir especificações formais com recursos orientado a objeto.

A aplicação dessa combinação possibilitou o desenvolvimento de novas propostas que foram surgindo com o decorrer desse trabalho [50] como a migração do protocolo IP Clássico para o protocolo NHRP, em que foi possível aplicar a metodologia proposta neste trabalho para desenvolver novos sistemas para a simulação dessas novas características, ampliando o processo de heranças dos sistemas planejados e especificados.

A linguagem SDL foi escolhida por ser uma linguagem de especificação formal, e por disponibilizar de uma ferramenta CASE (Computer Aided Software Engineering) como o SDT[1] (SDL Design Tool) que utiliza essa linguagem para a construção de especificação formal de *software* de comunicação, permitindo a realização de simulações e validações e geração de código, criando dessa forma sistemas mais confiáveis.

[1] Pacote SDT, versão 3.02 (*SDT Base, MSC Editor, Validator e Simulator*): adquirido pelo DT/FEEC/UNICAMP através do Projeto Temático FAPESP (Proc. 91/3660-0).

1.2 - Organização do trabalho

O trabalho está dividido em 5 capítulos e 3 apêndices (A-C). O capítulo 2 apresenta uma descrição das arquiteturas em que se discute o problema de interconexão, entre redes locais (TCP/IP) e redes ATM, além das propostas apresentadas para a interconexão dessas arquiteturas realizadas pelo IETF (Internet Engineering Task Force) e ATM Forum.

No capítulo 3 é proposto uma metodologia que combina a técnica de orientação a objeto OMT com a linguagem SDL'92 para a especificação formal dos protocolos IP Clássico e NHRP (Next Hop Resolution Protocol) do IETF (Internet Task Force). Os sistemas modelados em SDL'92 desses dois protocolos são apresentados neste capítulo, assim como o acompanhamento de suas evoluções.

As especificações dos protocolos apresentados no capítulo 3 foram construídas a partir das recomendações IP Clássico e NHRP, utilizando a metodologia proposta para o planejamento de um projeto de objetos. Um sistema (Sistema *Cloud*) que engloba as características comuns dos protocolos IP Clássico e NHRP, foi proposto para que se pudesse reutilizá-lo nos sistemas que descrevem suas características (Sistema IP Clássico e Sistema NHRP respectivamente), eliminando a necessidade de especificar características semelhantes por diversas vezes, tornando-se mais eficiente as especificações das evoluções destes sistemas.

Além de apresentar os sistemas Cloud, IP Clássico e NHRP, o capítulo 3 apresenta também a descrição das especificações resultantes da evolução e do acréscimo de novas características a esses protocolos. Os sistemas apresentados neste trabalho são: Sistema IP Clássico Migrado para o NHRP, Sistema NHRP que se comunica através do protocolo *Proxy-ARP* (Proxy Adress Resolution Protocol) e o Sistema IP Clássico Migrado que se comunica a uma nuvem ATM NHRP.

O capítulo 4 apresenta os resultados obtidos das validações e simulações dos sistemas descritos no capítulo 3. O capítulo 5 conclui esse trabalho.

Os apêndices são referentes à metodologia OMT utilizada neste trabalho, uma breve descrição das características da linguagem SDL, os recursos da ferramenta SDT empregada, e também as especificações em SDL dos sistemas especificados. Esses aspectos podem ser vistos nos apêndices A, B, C respectivamente. O conteúdo desses apêndices são:

Apêndice A: Apresenta uma introdução à linguagem SDL, desde as características básicas do SDL'88 [37] à novas características orientados a objeto presentes na linguagem SDL'92 [36].

Apêndice B: Apresenta os recursos da ferramenta SDT (SDL Desing Tool) [56] [58] utilizada para desenvolver as especificação em SDL, a simulação e a validação de cada um dos sistemas propostos na tese.

Apêndice C: Apresenta as principais figuras das especificações gráficas em SDL'92 referentes aos sistemas especificados dos protocolos IP Clássico e NHRP, e que estão presentes no capítulo 3, e que não foram apresentados ou que foram apresentados parcialmente naquele capítulo. As figuras do Sistema Cloud (Apêndice C.1) que não foram apresentadas no capítulo 3 serão todas apresentadas neste apêndice. Dos outros sistemas, [Sistema IP Clássico (Apêndice C.2), Sistema NHRP (Apêndice C.3), Sistema NHRP que se comunica através do protocolo *Proxy-ARP* (Apêndice C.4), Sistema IP Clássico Migrado para o NHRP (Apêndice C.5) e o Sistema IP Clássico Migrado que se comunica a uma nuvem ATM NHRP (Apêndice C.6)] são apresentadas apenas as figuras que contém características novas, como redefinição, acréscimos de sinais, uso de *packages* de blocos, entre outras.

Capítulo 2

Arquiteturas TCP/IP e ATM

2.1 - Introdução

A Internet é uma rede que interliga mundialmente diferentes tipos de tecnologias de rede, e que possui importância por ser uma rede utilizada amplamente. A rede ATM, por sua vez, possui diversas vantagens na sua utilização, tais como qualidade de serviço, possibilidade de atender a tráfegos diversos com altas velocidades, entre outras. A conversação entre essas duas redes é muito importante para obter na Internet as vantagens oferecidas pelo ATM.

A rede ATM (Asynchronous Transfer Mode) surgiu como solução para os problemas encontrados na B-ISDN (Broadband - Integrated Services Digital Network) para transportar diferentes tipos de dados. Contudo, a maior parte das redes locais são implementadas na arquitetura TCP/IP (Transmission Control Protocol/Internet Protocol). Para realizar a comunicação entre as camadas de rede ATM e IP existem diversas características que devem ser implementadas para fazer essa compatibilização [53].

O ATM Forum e o IETF (Internet Engineering Task Force) apresentam propostas que sugerem novos protocolos para realização da interoperabilidade entre essas duas arquiteturas. Além do ITU-T (antigo CCITT), existem órgãos como o IETF, que faz as padronizações dos protocolos que são executados sobre a rede Internet, e o ATM Forum, criado pelas empresas privadas para desenvolver e acelerar o desenvolvimento dos dispositivos em redes ATM. Esses órgãos, estão preocupados hoje com a compatibilização do endereçamento entre essas duas redes.

Nesse capítulo serão apresentadas as arquiteturas TCP/IP e ATM e alguns dos protocolos de interconexão dessas arquiteturas,propostos pelo IETF e ATM Forum para que no capítulo 3, baseados nas descrições informais desses órgãos, sejam apresentadas especificações formais desses protocolos utilizando uma combinação de OMT (Object Modeling Technique) e SDL'92 (Specification and Description Language) na construção das especificações de alguns desses protocolos, usando conceitos orientados a objeto para a evolução de suas especificações.

2.2- Arquitetura TCP/IP

A arquitetura Internet TCP/IP [25], foi desenvolvida com o objetivo de resolver o problema de interligar redes com tecnologias distintas. Para tal foi desenvolvido um conjunto específico de protocolos que resolveu esse problema de forma simples e satisfatória [49]. Contudo, o modelo TCP/IP e seus protocolos da Fig.2.1 também possuem alguns problemas específicos, como os descritos por Tanembaum [51]:

- Este modelo não distingue claramente o conceito de serviço, interface e protocolo, tornando-se um guia inadequado para projetos de novas redes utilizando novas tecnologias.

- A camada de subrede de acesso, não pode ser considerada propriamente como sendo uma camada na arquitetura, apesar de ser considerada como uma camada da arquitetura Internet por alguns autores, como na Fig. 2.2, descrita pela BRISA (Sociedade Brasileira para Interconexão de Sistemas Abertos) [10]. Tanembaum [51] a considera apenas como uma interface entre as camadas Internet e de Inter-rede (Fig. 2.1). A distinção entre camada e interface é crucial, e no modelo TCP/IP, essas terminologias não são bem distinguidas, por terem sido implementadas sem realizar o planejamento de camadas, como proposto pelo modelo OSI (Open System Interconection) da ISO (International Standartization Organization).

- Esse modelo não apresenta uma definição para distinguir as camadas física ou de enlace de dados, ou seja, não possui um padrão de funcionamento, independendo da plataforma utilizada. A camada Inter-rede mencionada na Fig. 2.1 não possui uma padronização de protocolos como no modelo OSI. Essa camada, referente a camada de subrede de acesso da Fig. 2.2, não é padronizada no modelo Internet tal que se pudesse utilizar qualquer equipamento de fabricantes diferentes.

- Embora os protocolos TCP e IP , implementados na camada de transporte e Internet respectivamente, tenham sido cuidadosamente planejados e bem implementados, isso não aconteceu com outros protocolos, como o TELNET, implementado na camada de aplicação, que são amplamente utilizados, e difíceis de serem substituídos, por não terem tido planejamento e documentação adequada, necessária para o seu entendimento.

O modelo Internet não é uma arquitetura bem planejada, que divide as tarefas de cada camada como o modelo OSI (Open System Interconection); sua implementação ocorreu antes da existência desse modelo que apresenta os limites de cada camada, distinguindo as tarefas de cada uma das camadas existentes.

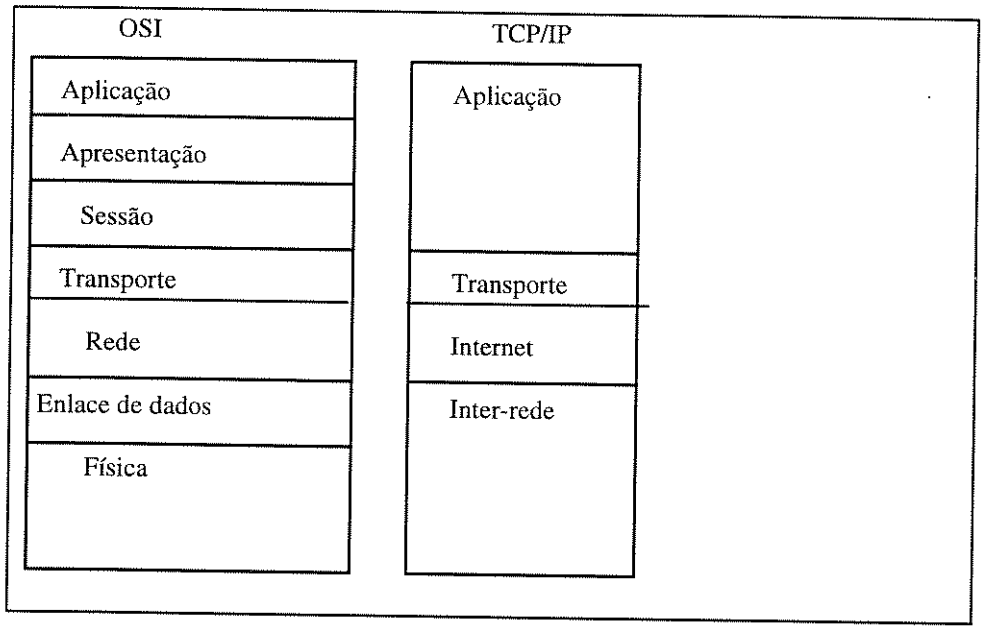

Figura 2. 1 - Modelo de Referência TCP/IP [51]

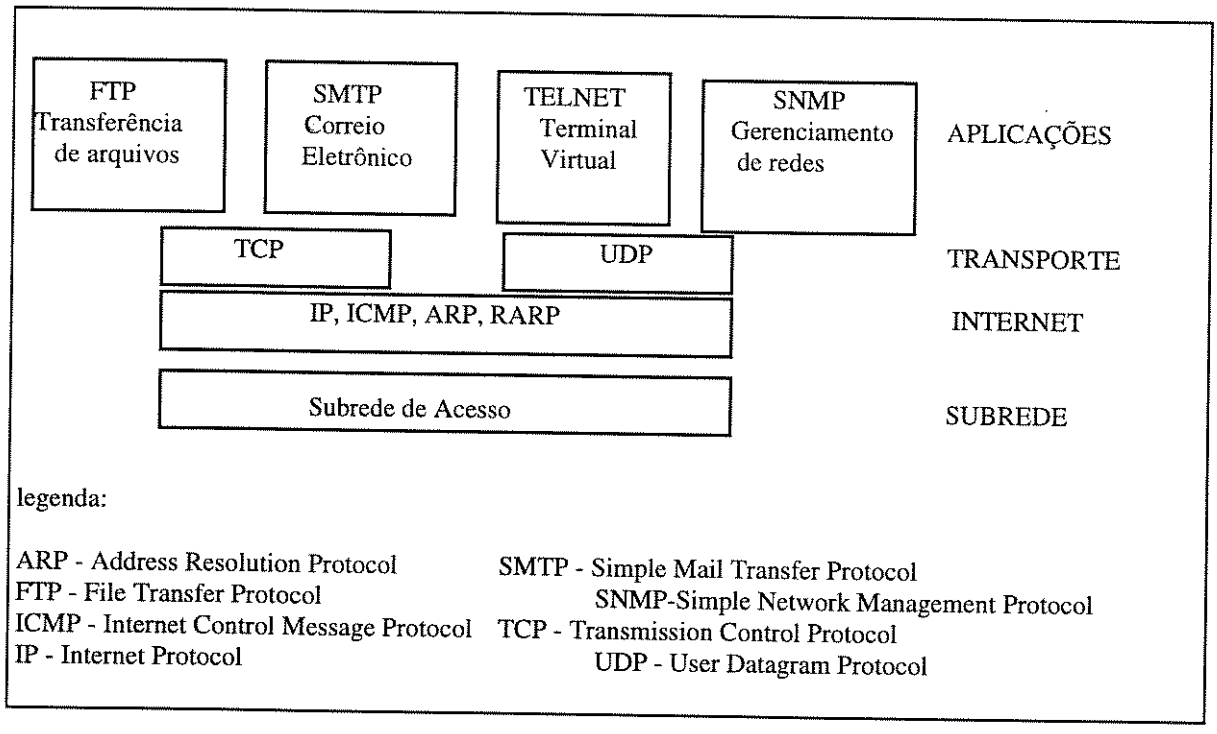

Figura 2. 2 - Camadas da Arquitetura Internet [10]

O conceito adotado pela Internet diz respeito à não necessidade de se utilizar subredes padronizadas. Deve-se apenas implementar interfaces do IP (Internet Protocol) necessárias para sua comunicação com as subredes disponíveis no mercado, sejam elas redes locais ou de longa distância, ficando a compatibilização entre tais subredes a cargo dos *gateways* [10], responsáveis por realizar a compatibilização de protocolos.

Para que seja possível estabelecer comunicação entre estações e *gateways* de uma rede *Internet* são utilizadas tabelas com informações de roteamento. Os principais protocolos utilizados na troca de tais informações podem ser do tipo exterior (EGP - Exterior Gateway Protocol) ou interior (IGP - Interior Gateway Protocol) [10].

1. EGP (Exterior Gateway Protocol)

   Este protocolo é utilizado na troca de informações de roteamento entre redes que pertencem a domínios de autoridade (sistemas autônomos) diferentes e vizinhos entre si. Esse tipo de protocolo não interpreta as distâncias especificadas nas mensagens de roteamento, de forma que os valores somente podem ser comparados dentro de um mesmo sistema autônomo, dificultando a tarefa de calcular a menor rota entre dois sistemas pertencentes a domínios diferentes

2. IGP (Interior Gateway Protocol)

   Este protocolo é usado ente *gateways* pertencentes ao mesmo domínio de autoridade, compreendendo de três protocolos básicos.

   - RIP (Routing Information Protocol)

     É usado na troca de informações de roteamento entre sistemas que empregam o algoritmo *vector-distance*. Tais sistemas são divididos em dois grupos, os passivos e os ativos. O grupo ativo informa suas portas para outros., os passivos (estações ou gateways), que apenas escutam e atualizam suas rotas. Os sistemas ativos fazem a cada 30 segundos um *broadcast* de mensagens contendo informações de roteamento, e constituidos pelo par: endereço IP da rede considerada e distância até essa rede. Tal distância é calculada pelo número de saltos (hops), isto é, do número de subredes que uma mensagem deve atravessar até chegar à rede destino.

- Hello

  Este protocolo também utiliza o algoritmo *vector-distance*. Contudo, difere-se do RIP pela métrica utilizada. O cálculo da sua métrica é feita com base no retardo, ou seja, no tempo estimado para uma mensagem a ser transferida até tal rede. Esse tempo é calculado a partir do campo de data e hora existente no cabeçalho das mensagens.

- OSPF (Open Shortest Path First)

  Este protocolo é utilizado na troca de informações de roteamento entre sistemas que empregam o algoritmo *link-state*. Os sistemas envolvidos, testam periodicamente os estados das suas ligações com todos os seus vizinhos e propagam tais informações de estado a outros sistemas da rede. Dessa forma, todos os sistemas da rede conhecem, a cada instante, todos os enlaces ativos da rede. Assim, os caminhos são calculados pelo algoritmo SPF (Shortest-Path-First de Dijkstra) [52].

O modelo TCP/IP possui um modo de comunicação na camada de rede, sem conexão, mas suporta ambos os modos na camada de transporte, dando aos usuários a escolha. O nível de inter-rede é o responsável pela transferência de dados através da inter-rede desde a máquina origem até a máquina destino. O pacote é encapsulado em um datagrama IP e o algoritmo de roteamento é executado para determinar se o datagrama pode ser entregue diretamente, ou se deve ser repassado para um gateway.

No modelo Internet de endereçamento nenhuma máquina possui o mesmo endereço. O endereço IP é único. Todos os endereços IP possuem um comprimento de 32 bits. A Fig. 2.3 apresentada por Tanembaum [51] mostra o endereçamento de acordo com sua classe de serviço. As características de cada classe são apresentadas na Tab. 2.1 na seção 2.4.

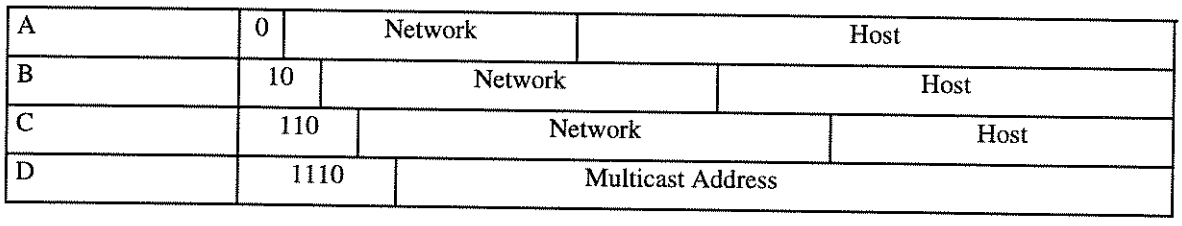

Figura 2. 3 - Formato do endereço IP [51]

Todos os hosts em uma rede devem possuir o mesmo número da rede nos endereçamentos IP, contudo, essa propriedade de endereçamento pode causar problemas com o contínuo crescimento das redes. Para resolver os problemas causados por esse tipo de endereçamento, existe uma solução que permite que essas redes possam ser divididas em várias partes; na literatura, essas partes são chamadas de subredes. O formato desse tipo de endereço é descrito na Fig. 2.3.

Figura 2. 4 - Endereçamento de subrede na classe B de rede [51]

A apresentação do formato do endereço de classe B apresentado na Fig. 2.4 foi devido à adoção dessa classe e desse tipo de endereçamento para construir os ADTs (Abstract Data Type) das especificações descritas no capítulo 3.

2.3- Arquitetura ATM

A rede ATM (Asynchronous Transmission Mode) surgiu como uma solução para a B-ISDN (Broadband - Integrated Service Digital Network), pois é a melhor proposta para a integração de tráfegos multimídia, dados, voz, áudio e vídeo. Possui também a característica de transmitir em altas velocidades e a vantagem de apresentar negociação de qualidade de serviço (QoS) [1] [14] [44] [48].

A rede ATM transmite seus dados com base na transmissão orientado à conexão e utiliza multiplexação por divisão de tempo assíncrona. Sua transmissão consiste de células de tamanhos fixos de 53 bytes. Cada célula possui um campo de informação de 48 bytes e 5 bytes de cabeçalho.

O campo de cabeçalho possui informações de controle para a célula (Fig. 2.5) como identificação, prioridade, além de informação de roteamento e chaveamento. Esse campo pode ser subdividido em outros 5 campos.

1. GFC (Generic Flow Control) - Esse campo é usado pela interface usuário-rede (UNI - User-Network Interface) para controlar a quantidade de tráfego que entra na rede. Isto permite que essa entidade limite a quantidade de dados que entra durante períodos de congestionamento.

2. VPI (Virtual Path Identifier) e VCI (Virtual Channel Identifier) - os identificadores de caminho virtual e de canal virtual, juntos formam o rótulo da conexão utilizado pelos comutadores para encaminhar as células ao destino. São também utilizados para a identificação de canal e simplificação do processo de multiplexação.

3. PTI (Payload Type Identifier) - o identificador do tipo de carga indica o tipo de informação contida na célula, se é do tipo usuário ou manutenção. Isto permite controlar o fluxo das células, sinalizando os dados a serem transmitidos em diferentes subcanais dos dados dos usuários.

4. CLP (Cell Loss Priority) - a prioridade de perda de célula indica se essa célula poderá ser descartada em caso de necessidade, como por exemplo, ao atingir a taxa de pico. Também é utilizado para indicar se uma célula pode ser descartada durante períodos de congestionamento da rede.

5. HEC (Header Error Check) - a checagem de erro de cabeçalho é utilizado para detecção de erros no envio de células.

Célula ATM

Header					Information
GFC Generic Flow Control	VPI/VCI Virtual Path and Channel	PTI Payload Type Identifier	CLP Cell Loss Priority	HEC Header error check	Payload
4 bits	24 bits	2 bits	2 bits	8 bits	48 bytes

Figura 2. 5 - Estrutura de uma célula ATM [44]

O modelo de referência do protocolo ATM é baseado no padrão desenvolvido pelo ITU (International Telecommunications Union) [44]. Esse modelo de arquitetura ATM (Fig. 2.6) é dividido em 3 camadas:

1. camada física

2. camada ATM

3. camada de adaptação ATM (AAL – ATM Adaptation Layer)

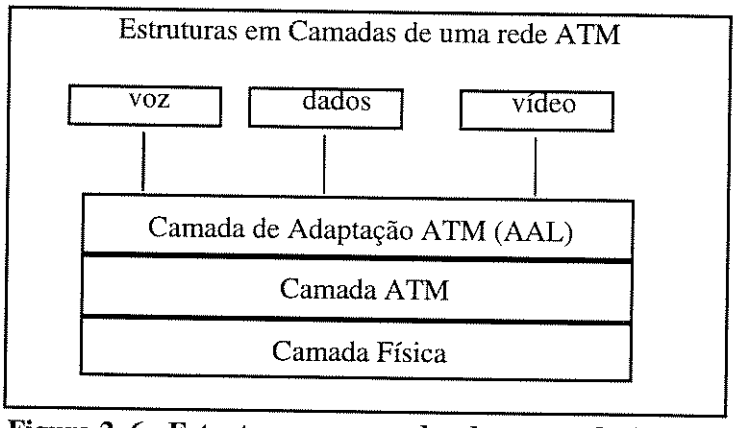

Figura 2. 6 - Estrutura em camadas de uma rede ATM [44]

A camada física codifica e decodifica os dados em formas de ondas adequadas, elétrica/óptica, para transmissão e recepção no meio de comunicação utilizado. Também provê funções de delineamento de células, geração e processamento de checagem de erro de cabeçalho, monitoramento de performance, e a combinação da taxa de carga com os diferentes formatos de transporte usados nessa camada.

A camada ATM pode suportar o transporte de diferentes tipos de serviços suportados pela B-ISDN (Broadband ISDN). Essa camada é responsável pela troca de células entre entidades da camada ATM, e pelo gerenciamento de fluxo de células que passam através dessa camada.

A tarefa da camada de adaptação ATM (AAL - ATM Adaptation Layer) é o de prover a ligação entre os serviços requeridos pelas camadas de rede mais altas e células ATM genéricas usadas pela camada ATM. Nessa camada foram definidas 4 classes (Tab. 2.1); sua classificação é feita de acordo com 3 parâmetros: relação de tempo entre fonte e destino, taxa de bit variável ou constante e modo de conexão.

1. Classe A. existe uma relação de tempo entre a fonte e o destino, a sua taxa de bit é constante e seu serviço orientado a conexão, geralmente utilizado em canal de voz.

2. Classe B. existe uma relação de tempo entre a fonte e o destino, seu serviço também é orientado a conexão, contudo sua taxa de bit é variável. Geralmente usado em canais de áudio ou vídeo comprimidos.

3. Classe C. não existe relação de tempo entre fonte e destino, sua taxa de bit é variável e o serviço orientado à conexão; sua aplicação geralmente é em transferência de arquivo orientado a conexão.

4. Classe D. não existe relação de tempo entre a fonte e o destino, sua taxa de bit é variável e o serviço é sem conexão. Sua aplicação geralmente é dada em interconexões de redes locais (LAN) e correio eletrônico

	Classe A	Classe B	Classe C	Classe D
Tempo entre fonte e destino	Requerido		Não Requerido	
Taxa de Bit	Constante	Variável		
modo de conexão	orientado a conexão			sem conexão

Tabela 2. 1- Esquematização das classes de serviços [51]

O formato uniforme das células, é uma vantagem, podendo facilitar o processamento de hardware, resultando em um ganho de velocidade no chaveamento.

Uma fonte de taxa de bit variável é caracterizada para suportar diferentes tipos de serviço (imagem, voz e dados), pois requerem velocidade e qualidade de serviços diferentes durante a transmissão.

Características de um tráfego de taxa de bit variável podem ser representadas por uma média de tempo longo e picos de taxas de emissão de célula, entre outros. Desde que é esperado que nem todas as fontes de taxa de bit variável gerem células nas suas taxas de pico, um comprimento de banda menor que a taxa de pico pode ser alocado para cada fonte. Isto permite que mais fontes sejam admitidas, do que o número de fontes admitidas pelas taxas de pico. Ao mesmo tempo, variações estatísticas no tráfego de carga de fontes individuais podem ser suavizadas enquanto muitas fontes são multiplexadas, resultando na melhor utilização das fontes compartilhadas.

Embora redes de meio compartilhado sejam baseadas nos mesmos princípios de multiplexação estatística, elas alocam inteiramente o comprimento de banda da ligação para um usuário em uma base temporal. Desde que a contenção entre usuários deva ser coordenado de uma maneira justa, os protocolos tendem a ser complicados. Em contraste, o comprimento de banda alocado é disponível para o usuário todo o tempo no ATM e a mesma variação estatística de tempo em um tráfego de taxa de bit variável é aproveitada para melhorar a utilização de recursos.

Com um formato de célula uniforme, os dados de fontes diferentes podem ser facilmente integrados nas redes ATM. Pela dedicação de recursos para um tempo de célula pequeno por fonte, dados de fontes diferentes parecem ser transmitidos concorrentemente. Isto é similar aos sistemas computacionais de tempo distribuído, onde a cada *job* que está esperando para ser executado, é dado um *quantum* fixo de tempo de processamento.

A unidade de integração de tráfego na rede ATM é uma célula ATM. Uma célula pode ser facilmente inserida num fluxo de célula de uma longa coleção de dados. Em uma rede de dados não ATM, por outro lado, um pequeno pacote pode ser atrasado até que a transmissão de um pacote grande seja completada.

As células são integradas na camada ATM. Contudo, nem sempre é desejável misturar células com qualidade de serviços (QoS) diferentes, principalmente na negociação de QoS, para estabelecer um contrato de serviço que suporte todas as QoS das células. Um algoritmo que faça a distinção das células com exigências de qualidade de serviços diferentes, é necessário, embora isso seja feito baseado no tipo AAL.

A ATM não especifica um algoritmo de controle de células e assume uma política de fila FIFO (First In First Out) com prioridades especificadas no bit de prioridade de perda de célula (CLP - Cell Loss Priority).

Para certas aplicações, a integração de tráfego baseado em perdas, pode não ser prática, pois quando um serviço de transporte confiável é oferecido, a perda de uma única célula, força a retransmissão do dado inteiro. A eficiência de um serviço de transporte confiável pode ser medida pelo produto do atraso e largura de banda. Além desse fator, a eficiência pode ser controlada pelos atributos de qualidade de serviço como: Taxa de pico de célula (PCR - Peak Cell Rate), Taxa de célula sustentada (SCR - Sustained Cell Rate), Taxa de perda de célula (CLR - Cell Loss Ratio), Atraso de transferência de célula (CTD - Cell Transfer Delay), Variação de atraso de célula (CDV- Cell Delay Variation), Tolerância à rajada (BT - Burst Tolerance) e Taxa mínima de célula (MCR - Minimum Cell Rate) [48].

Em uma rede de alta velocidade seus nós precisam ser simples para alcançar uma velocidade de comutação rápida. No ATM, a funcionalidade dos nós da rede são apresentadas de 3 maneiras:

- tomando vantagem da baixa taxa de erro de bit na fibra ótica, ocorrências de erros não são monitoradas pelos nós da rede. A manipulação de erros é executada somente pelos nós de borda ou pelos dispositivos no usuário final.

- visto que ATM possui células de tamanho fixo e também garante a ordem das células, o limite dos quadros dos dados é transparente para o receptor.

- o roteamento de células é realizado de uma maneira simples pela pré-alocação roteando rótulos para todo o caminho fixado através da rede. Isso elimina a necessidade para remontagem e reencapsulamento na rede.

O ATM é visto como uma rede cuja inteligência está localizada nos nós de borda enquanto os nós de trânsito provêm a transferência de dados através de caminhos pré-estabelecidos. Esta estratégia de abertura de loop é um exemplo de estratégia de controle fora da banda (out-of-band; OOB), onde o controle dos dados de informação podem ser levados por caminhos diferentes na rede. Devido à falta de informação de retorno, um problema no caminho dos dados pode não ser reconhecido imediatamente. Isso ocorre quando um congestionamento acontece em um caminho que pode passar despercebido por algum tempo, resultando em perda de uma grande quantidade de dados. Com o decorrer do tempo, a promessa de uma rede ATM simples pode não ser cumprida, devido ao fato de mais funções complicadas se tornarem necessárias na ATM. Foi apontado então, que certas formas de planejamento inteligente pudessem ser necessárias para satisfazer os requisitos de qualidade de serviços diferentes durante a conexão. Para aplicações que requerem serviços de transporte confiável, a falta de checagem de erro na rede pode forçar retransmissões dispendiosas.

Uma rede ATM consiste de um conjunto de switches ATM interconectados por interfaces e ligações ATM ponto-a-ponto [1]. Os switches ATM suportam dois tipos de interfaces, NNI - Network-Node Interface e UNI - User-Network Interface. A UNI conecta sistemas finais ATM (hosts, roteadores, etc.) a switches ATM, enquanto a NNI pode ser descrita como qualquer ligação lógica ou física através do qual, dois switches ATM trocam o protocolo NNI. Por essa razão, que a conexão entre switch ATM público e switch ATM privado é um UNI, conhecido como uma UNI

pública, desde que estes switches não trocam tipicamente informações NNI. A esquematização dos conceitos de UNI e NNI pode ser visualizada na Fig. 2.7:

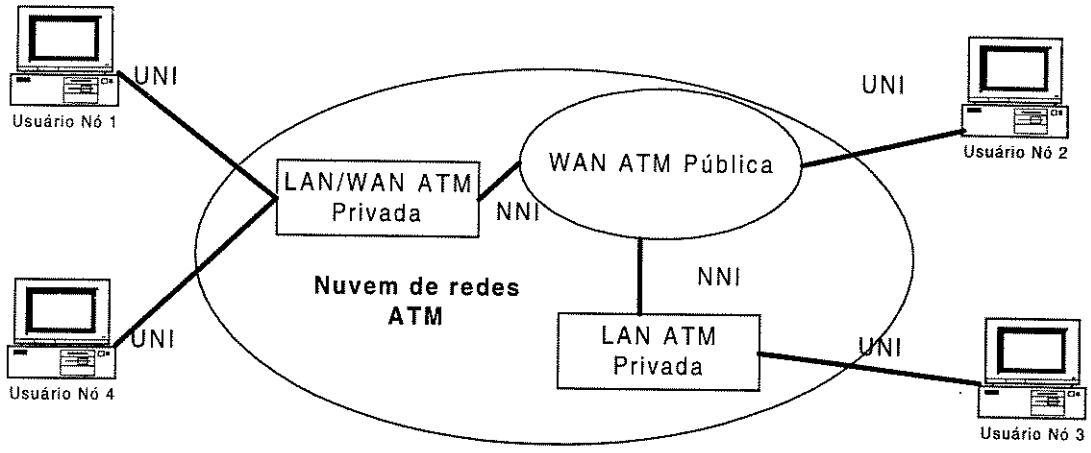

Figura 2. 7 - Interface NNI e UNI de Switches ATM [3]

As redes ATM são orientadas a conexão, isto significa que se necessita estabelecer a conexão de um circuito virtual para fazer a transferência de dados. Os circuitos ATM são de dois tipos: VCI (Virtual Channel Identifier) e VPI (Virtual Path Identifier). A combinação do identificador de canal virtual (VCI) e do identificador de caminho virtual (VPI), identifica os canais virtuais. Um caminho virtual é um pacote de canais virtuais, que são comutados transparentemente através da rede ATM em uma base VPI comum.

A operação de comutação é simples por que mecanismos externos estabelecem a priori as tabelas de tradução local para a transmissão dos dados. A maneira na qual essas tabelas são estabelecidas, determinam dois tipos fundamentais de conexões ATM:

- PVC (Permanent Virtual Connection): Uma PVC é uma conexão estabelecida por algum mecanismo externo, tipicamente de gerenciamento de rede, na qual um conjunto de switches entre uma fonte ATM e um sistema destino ATM são programados com valores VCI/VPI apropriados. A sinalização ATM pode facilitar o estabelecimentos dos PVCs, mas por definição, PVCs sempre requerem alguma configuração manual.

- SVC (Switched Virtual Connection): Uma SVC é uma conexão que é estabelecida automaticamente através de um protocolo de sinalização. SVCs não requerem uma interação manual, necessária para estabelecer uma conexão PVC. Todos os protocolos das camadas mais altas operando sobre ATM primariamente usam SVCs.

Os protocolos de sinalização ATM variam de acordo com o tipo de ligação ATM - sinalização UNI ATM é utilizada entre um sistema final ATM e um switch ATM através de uma UNI ATM; sinalização NNI é utilizada através de ligações NNI.

O ATM Forum analisou modelos fundamentalmente diferentes para endereçamento. No desenvolvimento de esquema de endereçamento para redes privadas, é utilizado o padrão de endereçamento UNI. O ITU-T emprega o uso de endereçamento E.164 como uma estrutura de endereçamento para redes ATM públicas (B-ISDN). Os endereços E.164 são recursos públicos e caros, inviabilizando sua utilização em redes privadas. O endereço final ATM padronizado pelo ATM Forum, é codificado tanto no endereço NSAP (Network Service Access Point) ou como endereço UNI-Público E.164.

Os tipos de endereços E.164 e NSAP diferem na forma de como são vistos na camada ATM em relação as camadas de rede existentes (IP, IPX, entre outros). Para transportar os endereços ATM existem 2 modelos, o modelo *Peer* e o modelo *Overlay*. O modelo *Peer* identifica a camada ATM, como uma camada de rede, que poderia ser identificada como um *end-point* de outras camadas de rede. Esse modelo trata a camada ATM como um par de camada de redes existentes.O modelo *Overlay* separa a camada ATM de qualquer protocolo existente, definindo uma nova estrutura de endereçamento. Nesse modelo todos os protocolos devem operar sobre a camada ATM, ficando esse modelo conhecido omo subrede ou *overlay*.

2.4 - IETF e ATM Forum

Duas entidades se preocupam em elaborar padronizações dos protocolos de interconexão de redes locais com a rede ATM. O IETF (Internet Engineering Task Force) é uma entidade que faz a padronização dos protocolos utilizados pela rede Internet. As propostas de interconexão dessa entidade se baseiam no modo nativo (Native Mode) [47], fazendo a comunicação direta entre a camada IP (Internet Protocol) e a ATM (Asynchronous Task Force) (Fig. 2.8) . A outra entidade, o ATM Forum realiza fóruns para padronizações para rede ATM. Esse órgão possui a proposta do LANE (LAN Emulation) que trata a comunicação dessas duas camadas através da emulação da camada MAC (Medium Access Control), esse modo é conhecido como modo *overlay* (Fig. 2.8).

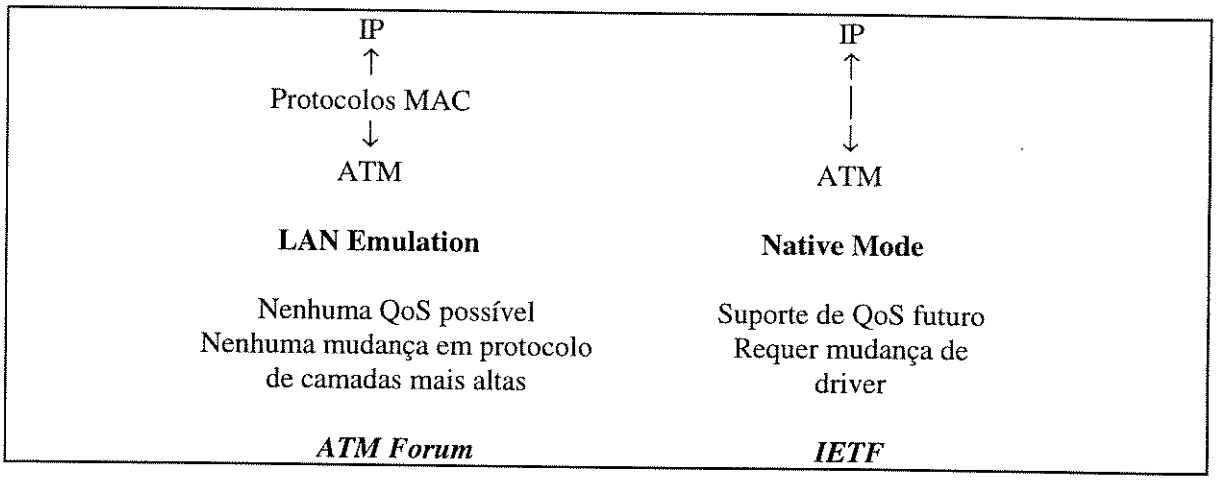

Figura 2. 8 - Métodos de interconexão ATM [1]

O IETF padronizou alguns protocolos de modo nativo [1] [3], como o Classical IP Over ATM [20] [28] [32], conhecido como IP Over ATM ou IP Clássico; NHRP - Next Hop Resolution Protocol [21] [30], MARS- Multicast Address Resolution Server [23] [24]. Outros protocolos de interconexão são os estabelecidos pelo ATM Forum (modo *overlay*), como o LANE - LAN Emulation [5] [6] [7] e o trabalho MPOA - Multiprotocol Over ATM [4]. Esse último se baseia nos protocolos especificados pelo IETF e LANE.

2.4.1 - IETF (Internet Engineering Task Force)

O IETF é uma entidade que faz a padronização dos protocolos utilizados pela rede Internet, como o TCP/IP. Além deste, esse órgão propõem outros protocolos para a interconexão entre a rede ATM: IP Clássico, NHRP e o MARS .

2.4.1.1 - IP over ATM ou IP Clássico

O protocolo IP over ATM ou IP Clássico [20] [28] [32] proposto pelo IETF, sugere um modelo clássico que trata os adaptadores como uma interface de rede para a operação da pilha de protocolos IP em um paradigma baseado em LAN (Local Area Network).

As características do modelo clássico são:

- O mesmo tamanho de unidade de transmissão máxima (MTU - Maximum Transmission Unit) é utilizado para todas as conexões virtuais nas subredes IP lógicas (LIS - Logical IP Subnetwork) [40].
- Encapsulamento LLC/SNAP (Logical Link Control/SubNetwork Access Protocol) padrão dos pacotes IP.

- As arquiteturas de roteamento IP fim-a-fim permanece a mesma.

- Endereços IP são traduzidos para o endereço ATM pelo uso de um serviço ATMARP (ATM Adress Resolution Protocol) dentro da LIS.

- Uma subrede IP é usada por muitos hosts e roteadores. Cada conexão virtual (VC - Virtual Connection) conecta diretamente dois membros IP dentro da mesma LIS.

O IP Clássico sobre ATM pode ser executado pelos tipos de conexões: PVC (Permanent Virtual Connection) ou SVC (Switched Virtual Connection). Também suporta subredes IP lógicas (LIS) e permite aos administradores de rede definir as características de QoS na base subrede para subrede [3].

No cenário LIS (Logical IP Subnetwork), cada entidade administrativa em separado, configura seus hosts e roteadores dentro de uma subrede IP lógica (LIS) fechada (Fig. 2.9) [1]. Em uma rede ATM, cada LIS opera e comunica independentemente de outras LISes. Hosts conectados em uma rede ATM comunicam diretamente a outros hosts dentro de uma mesma LIS (LIS 2 da Fig. 2.9). Comunicações com LISes locais são providas via um roteador virtual. Este roteador é um end-point ATM conectado a uma rede ATM, que é configurada como parte de uma ou mais LISes. Esta configuração pode resultar em operação de LISes disjuntas sobre uma mesma rede ATM (Fig. 2.9 e 2.10).

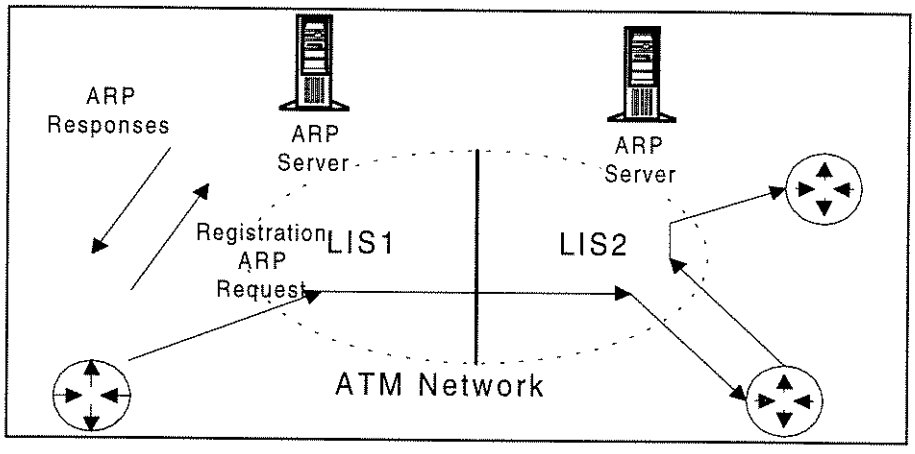

Figura 2. 9 - Roteamento entre LISes e o modelo de IP Clássico [1]

Hosts de diferentes subredes IP devem se comunicar via um roteador IP intermediário, embora possa ser possível abrir um VC (Virtual Channel) direto entre 2 membros IP sobre uma rede ATM. A Fig. 2.10 apresenta a troca de sinais através de uma LIS1 e uma LIS2. Nesta figura a LIS1 possui um LLC (Logical Link Control) para a realização de uma conexão, compartilhando essa LLC para interface IP e ATM e cliente ARP. No caso da LIS2, existe um LLC para interface IP/ATM e outro LLC para interface com o cliente ARP.

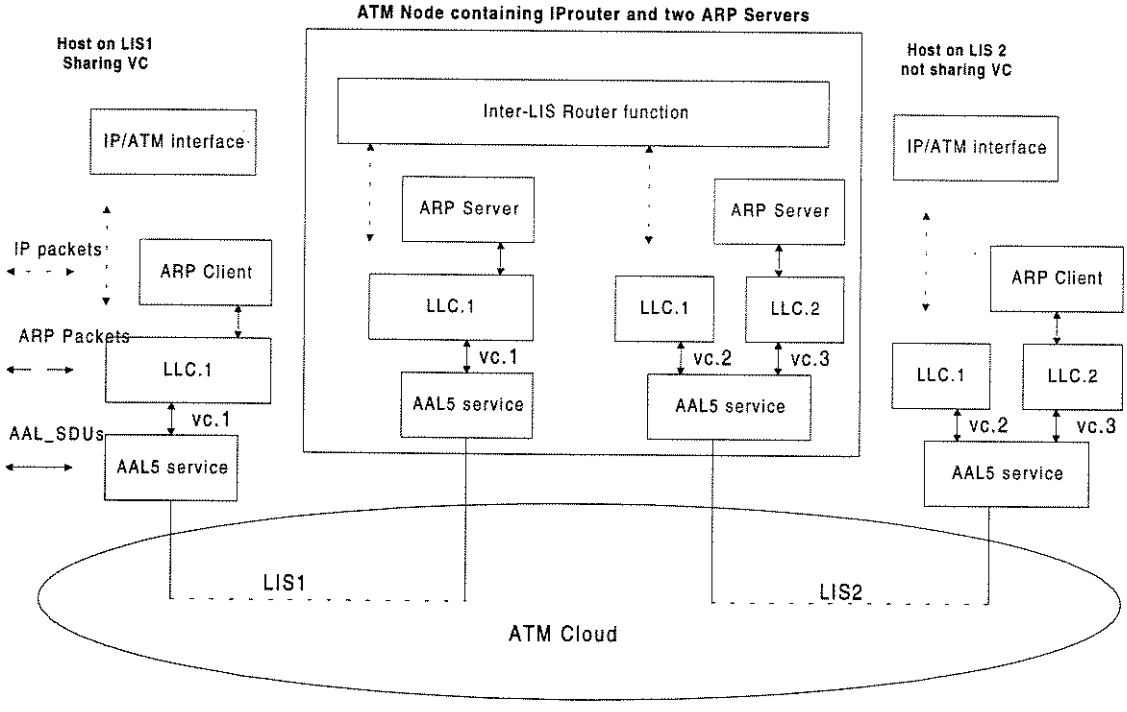

Figura 2. 10 - Roteamento inter LIS no protocolo IP Clássico [3]

No protocolo IP Clássico é necessário um esquema de mapeamento de endereços IP em endereços ATM, como também regras para determinação de quando construir uma nova chamada se for utilizado uma ligação SVC (Switched Virtual Connection) ao invés de PVC (Permanent Virtual Connection), para executar protocolo de datagrama IP sem conexão sobre uma rede ATM baseado em conexão.

O comprimento do pacote padrão do IP Clássico sobre ATM é 9180 bytes (MTU-Maximum Transmission Unit), que é grande o suficiente para suportar pacotes Ethernet, Token Ring, FDDI (Fiber Distributed Data Interface) e SMDS (Switched Multimegabit Data Service) sem fragmentação.

Em ligações SVCs, estações finais devem ter um modo de mapear endereços IP em endereços ATM e conexões virtuais automaticamente sob demanda. É necessário um elemento de protocolo adicional, servidor de ATMARP (ATM Adress Resolution Protocol), para realizar essa operação.

O servidor ATMARP é um recurso em cada LIS para que as estações finais possam pedir informações com o objetivo de encontrar o endereço ATM de um dado endereço IP destino (ATMARP_Request) (Fig. 2.11). O servidor ATMARP mantém automaticamente em cada LIS uma base de dados para mapear endereços IP para endereços ATM. Esses endereços de mapeamento geralmente são armazenados na memória cache de cada Servidor ARP de cada LIS (Fig. 2.11).

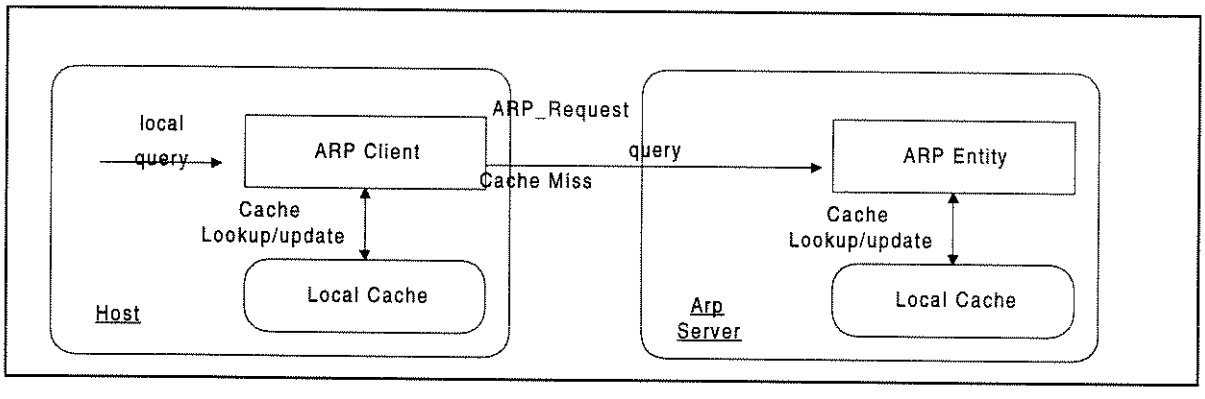

Figura 2. 11 - Comunicação entre Host e Servidor ARP do RFC 1577 [3]

Além do RFC1577 [28] que propõe o protocolo IP Clássico, existe o RFC1932 [32] que apresenta novas propostas a esse protocolo para solucionar as limitações existentes na proposta atual. O RFC1932 propõe a integração de técnicas adotadas no protocolo NHRP (Next Hop

Resolution Protocol) para realizar o estabelecimento de conexão direta. Características como a utilização de endereçamento Proxy-ARP do protocolo NHRP, são propostas apresentadas pelo grupo de trabalho ROLC (Routing Over Large Clouds) para obter endereços de um roteador que serve um destino fora de uma NBMA (Non Broadcast Multi-Access) que é servido por um único roteador na NBMA.

O RFC1932 [32] apresenta diversas propostas para se construir o protocolo NHRP, baseado no modelo apresentado pelo RFC1577, um deles é o modelo *Peer*, que substitui roteadores/gateways em uma base de endereçamento *peer* correspondente a cada entidade em uma nuvem ATM. Contudo, discussões chegaram a conclusão que esse modelo é o mais difícil de se desenvolver, desencorajando o desenvolvimento desta proposta. Outro modelo é o chamado modelo convencional que assume que um roteador possa transmitir pacotes IP, célula por célula com o VPI/VCI (Virtual Path Identifier/Virtual Channel Identifier), identificando um fluxo entre roteadores adjacentes, ao invés de identificar num fluxo entre pares de nós. Um outro modelo proposto é o modelo integrado que considera um único protocolo de roteamento para ser usado tanto para IP como para ATM, este protocolo está sendo estudado no grupo de Multiprotocolo do ATM Forum. O modelo de roteamento integrado é uma proposta que usa o roteamento PNNI (Private NNI) como um protocolo de roteamento IP.

Apesar desses modelos propostos no RFC 1932 [32], muitos desses trabalhos sugeridos não apresentaram bons resultados. Mas através das discussões desse RFC surgiram novos drafts, como *IP Broadcast over ATM Networks* [19], *Classical IP and ARP over ATM* [20], *Classical IP to NHRP transition* [22] que discutem o novo funcionamento do protocolo IP Clássico e ARP.

Neste trabalho de tese são apresentadas especificações formais utilizando uma técnica de construção de projeto orientado a objeto OMT (Object Modeling Technique) combinada à linguagem SDL'92 para que as especificações possam ser reutilizadas, com o objetivo de evoluí-las, como foi o caso das especificações para o protocolo IP Clássico proposto pelo RFC 1577 [28], nos quais foram empregadas para a sua evolução os seguintes *drafts: Classical IP over ATM* [20], *Classic IP to NHRP transition* [22] , assim como as propostas sugeridas pelo RFC 1735 [30] e o *draft NBMA Next Hop Resolution Protocol NHRP* [21].

2.4.1.2 - NHRP (Next Hop Resolution Protocol)

O protocolo NHRP (Next Hop Resolution Protocol) [21] [30] é um protocolo baseado no modelo IP Clássico, apresentado na seção anterior, substituindo o conceito de LIS (logical IP Subnetwork) pela noção de rede NBMA (Non-Broadcasting Multi Access), exemplos desse tipo de tecnologia de rede podem ser citadas: ATM, Frame Relay ou X.25; que permitem que múltiplos dispositivos sejam ligados a mesma rede, mas que não possibilita facilmente o uso de mecanismos *broadcast* (difusão).

Uma única rede NBMA pode suportar múltiplos domínios administrativos; isso pode ser alcançado dentro de cada uma das conexões diretas permitidas. Apesar disso, esse protocolo impõe restrições de topologias para evitar problemas de *loops* no roteamento.

Nesse protocolo, no lugar dos servidores ARP, é utilizado a noção de servidor NHRP (NHS - NHRP Server). Cada NHS mantém tabelas caches "next-hop resolution" com mapeamento de endereço IP para ATM de todos aqueles nós associados com um NHS particular, ou para prefixos de endereços IP alcançados através de nós (roteadores) servido pelo NHS.

O funcionamento das requisições realizadas entre o Servidor NHRP e o *Host* no protocolo NHRP, mostrado na Fig. 2.12, é similar ao funcionamento do protocolo IP Clássico (Fig.2.11), não necessitando atualizar a Cache.

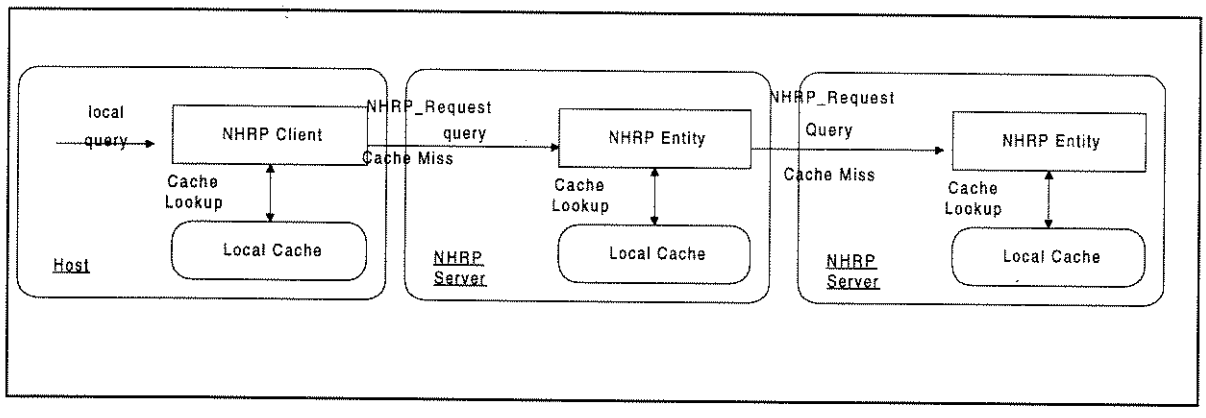

Figura 2. 12 - Requisição NHRP pode ser roteado através de mais de um Servidor NHRP [3]

Pode-se descrever o modo de funcionamento desse protocolo da seguinte maneira (Fig. 2.12): quando um nó determina que necessita transmitir um pacote através da rede NBMA, e conseqüentemente necessita resolver um endereço ATM particular, formula e transmite uma requisição NHRP (NHRP_Request) que empacota e envia para seu NHS. Tais requisições, como todas as mensagens NHRP, são enviadas em pacotes IP.

O NHRP permite um número de características adicionais, incluindo armazenamento de rota, detecção de loops dentro de uma rede NBMA e recuo aos NHSs capazes de transmitir pacotes ao longo de uma rota para um endereço particular. O servidor pode também ser um ponto de passagem intermediária para aqueles endereços que o sistema final não seja capaz de interpretar ou deseje suportar conexões de dados diretos.

O protocolo NHRP pode ser utilizado para comunicações tanto entre roteadores ou entre hosts. Alguns detalhes importantes são descritos pelo IETF nos RFC 1735 [30] e RFC 1932 [32] e no draft *NBMA Next Hop Resolution Protocol NHRP* [21].

O modo do funcionamento do protocolo NHRP, pode ser visto na Fig. 2.13. Esta figura apresenta o pedido de conexão entre o *host* localizado na LIS1 que deseja comunicar ao *host* localizado na LIS4, para estabelecer a conexão entre esses *hosts,* o pedido deve passar pelos servidores NHS intermediários até alcançar o destino desejado.

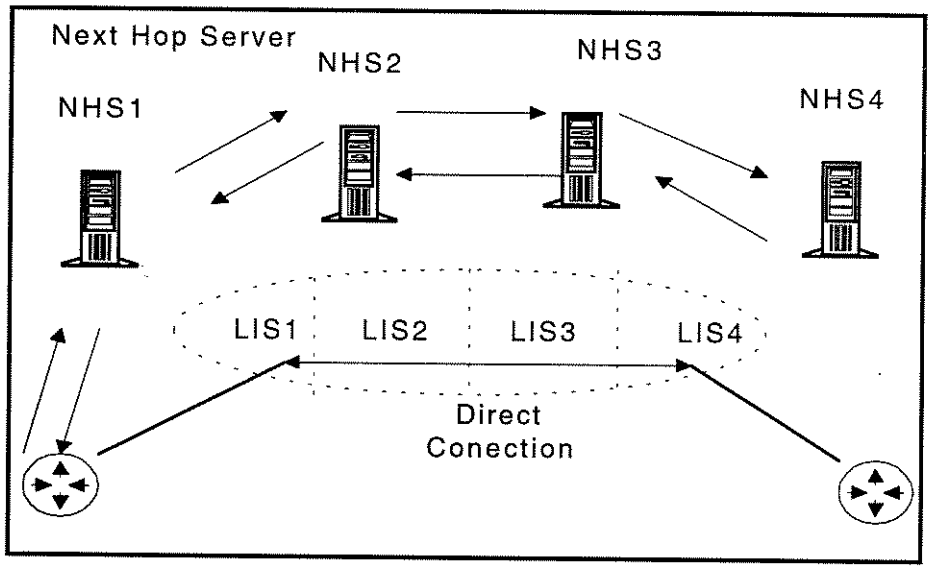

Figura 2. 13 - Operação do protocolo NHRP - Next Hop Resolution Protocol [3]

Como discutido no artigo de Armitage [3], um objetivo do trabalho do NHRP é o de eliminar a implementação do estilo de LIS (Logical IP Subnetwork) do RFC 1577; como mostrado na Fig. 2.13. Na prática isso irá requerer, na implementação de hosts, que seus clientes ARP do RFC 1577 sejam descartados em favor dos clientes NHRP.

2.4.1.3 - MARS (Multicast Address Resolution Server)

O protocolo IP Clássico (seção 2.4.1.1) não possui suporte específico para operação multicast. Para resolver esse problema, foi proposto a noção de um servidor de resolução de endereço multicast (MARS - Multicast Address Resolution Server) [23] [24] que pode ser considerado análogo de um servidor ARP descrito no RFC 1577 [28].

Um MARS serve um grupo de nós conhecidos como "*clusters*". Todos os sistemas finais dentro de um cluster são configurados com um endereço ATM de um MARS. O MARS suporta multicast através "*multicast meshes*" de conexões ponto-para-multiponto ligadas nessa *mesh*, ou através de servidores *multicast* [48].

O modelo de servidores *multicast* (Fig. 2.14) é geralmente proposto em ambientes onde os circuitos virtuais através da UNI (User-Network Interface) são importantes, ou no contexto do AAL (ATM Adaptation Layer) sejam limitados aos pontos finais (LIS Host). Os circuitos virtuais são estabelecidos no servidor *multicast* à todos os membros do grupo (*clusters*) que desejam enviar um tráfego *multicast*. Os pacotes enviados ao servidor *multicast* são retransmitidos à todos os pontos finais registrados no servidor (MARS) como sendo membro de um grupo *multicast* particular. Esse mecanismo de funcionamento pode ser visto na Fig. 2.14 [3].

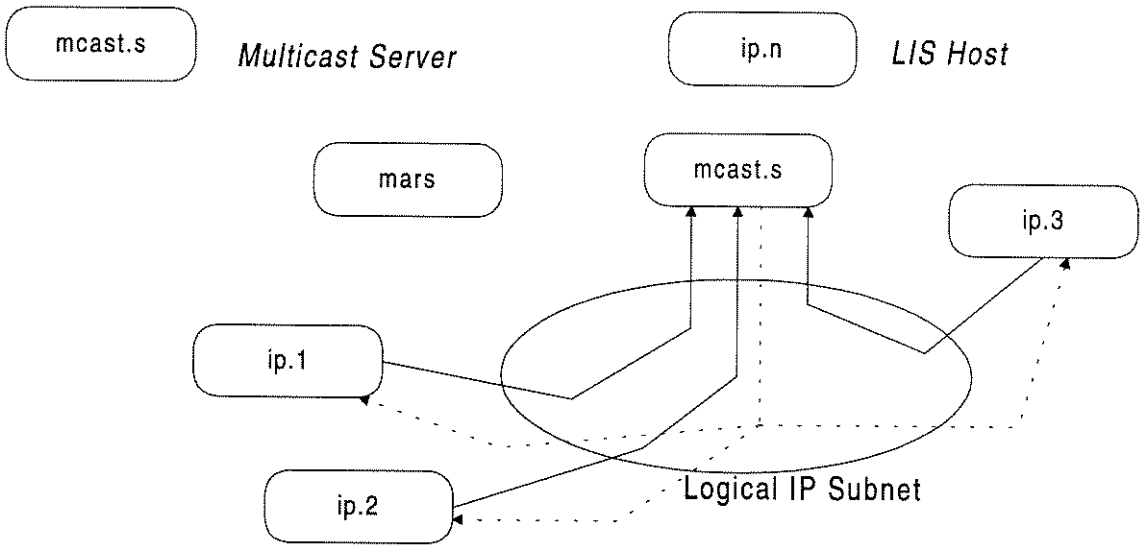

Figura 2. 14 - Mecanismo de funcionamento de um Servidor Multicast [3]

Infelizmente, usuários que requerem uma alta performance de *multicasting*, pode achar um inconveniente a utilização de servidores, pois esses servidores podem ser tornar um gargalo,

causando uma demora na distribuição do tráfego. Trabalhos recentes tem sido propostos para redefinir esse mecanismos de multicast no RFC 1577; estes trabalhos tentam suportar o comportamento *multicast* IP descrito no RFC 1112 [24], pela combinação de servidores e abordando conexões ponto-multiponto. O MARS pode também ser utilizado como modelos para suporte multicast em outros protocolos incluindo NHRP (Next Hop Resolution Protocol) e MPOA (Multiprotocol over ATM) [4].

2.4.2 - ATM Forum

O ATM Forum surgiu pela união dos operadores de telecomunicações com o objetivo de acelerar o desenvolvimento e emprego dos produtos ATM e serviços em ambientes privados. Essa entidade utiliza como o tipo de endereçamento UNI3.0/3.1 (User-Network Interface) para redes privadas, e em redes públicas utiliza a padronização do B-ISDN E.164. As propostas de protocolos de interconexões são o LANE (LAN Emulation) [5] [6] [7] e mais recentemente o MPOA (Multi Protocol over ATM) [4].

No ATM Forum, a tarefa de especificação dos serviços de LANE foi originalmente tratado pelo grupo de trabalho SAA (Service Aspects and Applications) no começo de 1993. Um novo grupo de trabalho "LAN Emulation" foi criado em novembro de 1993 para acelerar essa especificação a qual foi adotada pelo ATM Forum em março de 1995 [60].

O LANE [5] [6] [7] é uma proposta que permite que um protocolo ATM emule serviços de redes locais existentes, como Ethernet e Token-Ring, fazendo com que os protocolos da camada de rede operem como se eles fossem conectados a uma LAN (Local Area Network) convencional. A especificação do LANE define como uma rede ATM possa se passar por uma rede local, executando um conjunto suficiente de serviços de controle de acesso ao meio de tecnologias de LAN existentes (Ethernet e Token Ring), tal que os protocolos de camadas mais altas possam ser usadas sem modificações.

Um outro protocolo também estudado pelo ATM Forum é o MPOA [1] [2] [4] [61] que acopla muitas das características de endereçamento envolvidas pelos esquemas que esse protocolo aborda (IP over ATM, NHRP, MARS e LANE) [2]. Esse protocolo suporta múltiplos protocolos de rede sobre ATM de uma maneira eficiente e escalonável. Também define subredes virtuais na camada de rede que pode estender os limites geográficos.

O protocolo MPOA permite que o tráfego seja encaminhado diretamente ao seu destino sobre um circuito virtual de um *hop*. Com o mapeamento dos protocolos da camada de rede diretamente para ATM, MPOA reduz o overhead e o tráfego *broadcast* sobre a rede, suporta tamanhos variados de MTUs (Maximum Transmission Unit) e abre caminho para que as aplicações tirem vantagens das capacidades de qualidade de serviços oferecidas pela rede ATM.

2.5 - Conclusão

Os órgãos IETF e ATM Forum apresentam propostas para interconexão de redes locais e ATM, que operam de diferentes formas como no modo *overlay* e no modo *nativo*. O protocolo IP Clássico [20] [28] [32] e o NHRP (Next Hop Resolution Protocol) [21] [30] foram escolhidos para que fossem especificados neste trabalho, por apresentarem propostas abertas para discussão, por ainda não existir implementações desses protocolos e por terem sido propostos pela entidade que padroniza os protocolos que são executados na rede Internet, a maior rede utilizada atualmente.

Esses protocolos escolhidos poderão ser reutilizados para eventuais implementações dos protocolos MARS, MPOA e LANE, que foram apresentados superficialmente neste capítulo, apenas para realizar uma breve comparação com os protocolos escolhidos.

Capítulo 3

Sistemas especificados a partir dos Protocolos IP Clássico e NHRP

3.1 - Introdução

Com base nas descrições apresentadas nas seções 2.4.1.1 e 2.4.1.2 do capítulo 2, referentes aos protocolos IP Clássico e NHRP (Next Hop Resolution Protocol) apresentados pelo IETF (Internet Engineering Task Force), neste capítulo são propostas as especificações formais desses protocolos, utilizando uma metodologia orientada a objeto OMT (Object Modeling Technique) combinada a linguagem formal SDL (versão 92) para auxiliar no desenvolvimento e evolução dessas especificações.

O reuso das especificações apresentadas neste capítulo é baseado no Sistema Cloud, apresentado neste capítulo, que representa a especificação do funcionamento comum entre os protocolos IP Clássico e NHRP para se que pudesse reutilizar essas características empregando o recurso de herança.

A partir do Sistema Cloud, foi gerado diretamente o Sistema IP Clássico e o Sistema NHRP. Dessas especificações foram surgindo novos sistemas, referentes a evolução e acréscimo de novas características como os sistemas IP Clássico migrado para NHRP, duas nuvens NHRP se comunicando através do protocolo *Proxy-ARP* e uma nuvem NHRP se comunicando a uma nuvem IP Clássico através do uso do *Proxy-ARP*.

A utilização de uma metodologia orientada a objeto combinada a uma linguagem formal, foi aplicada para que se pudesse obter resultados eficientes durante a realização das especificações e suas evoluções. A descrição da metodologia utilizada está proposta na próxima seção.

3.2. - Metodologia: OMT + SDL'92

A adoção de uma especificação formal de protocolos de comunicação auxilia o desenvolvedor a compreender o funcionamento dos protocolos e a analisar características de seu comportamento. Visando a construção de uma especificação formal de sistemas de

telecomunicações, é necessária a escolha de uma linguagem que seja capaz de realizar o modelamento de protocolos, como é o caso da linguagem SDL (Specification and Description Language)[36] [37].

A utilização de linguagens com recursos orientados a objeto tem crescido com o passar do tempo devido às vantagens trazidas por esse tipo de modelagem. O emprego de recursos orientado a objeto permite a reutilização de classes já modeladas para o desenvolvimento de um novo sistema, e também, para acrescentar novas características sem necessitar que se construa todo o sistema novamente, resultando em um menor esforço de custo e tempo.

A construção de uma especificação envolve a escolha de uma linguagem formal para construir as especificações. Existem várias linguagens formais de especificação de sistemas (FDT - Formal Description Technique) como LOTOS (Language of Temporal Ordering Specification) e Estelle padronizadas pela ISO (International Standatization Organization) e SDL padronizada no final da década de 70 pelo antigo CCITT (Comité Consultatif International Téléphonique), atualmente conhecido como ITU-T (International Telecommunication Union - Telecommunication Standartization Sector).

LOTOS é uma linguagem formal desenvolvida pela ISO que foi especificamente desenvolvida para descrições formais da arquitetura OSI (Open Systems Interconection)[12]. Essa linguagem chamada de álgebra de processos, tem como objetivo o de especificar o comportamento de processos em um alto nível de abstração [17].

A linguagem Estelle é a segunda geração de FDT (Formal Description Technique) padronizada pela ISO. Estelle pode ser visto como uma extensão para o Pascal ISO[2] . É uma técnica baseada em um modelo de transição de estados estendidos, ou seja, um modelo de um autômato estendido não determinístico[11]. Essa linguagem colaborou com a construção da padronização da linguagem SDL [34].

A linguagem SDL, foi padronizada pelo ITU-T para especificação e projeto de sistemas de telecomunicações. É uma linguagem de fluxo de dados que apresenta as formas de programação textual e gráfica. A seguir são citadas algumas características da linguagem SDL que tem tornado sua escolha preferencial em relação as outras linguagens formais para especificação formal de protocolos [33]:

[2] Linguagem Estruturada Pascal.

1. A estrutura da linguagem SDL imita o modo como as pessoas pensam sobre o funcionamento dos serviços a serem implementados, tornando-o mais natural do que linguagens procedurais como a linguagem C/C++.

2. O SDL suporta refinamento passo-a-passo, pois sua hierarquia permite partir de uma visão geral (sistemas e blocos) até níveis mais altos de detalhamento (processos, procedimentos e serviços) além de mostrar a troca de sinais necessárias no sistema.

3. O formalismo do SDL suporta uma tradução mecânica da especificação em um modelo de máquina de estado finito (FSM - Finite State Machine) e o uso de gráficos MSC (Message Sequence Chart) que descrevem o funcionamento temporal dos sistemas através do uso de máquinas de estado. Os códigos para simulação e validação podem ser mostrados através do MSC, e são gerados a partir de códigos descritos na linguagem C.

4. SDL é uma linguagem que possui suporte internacional em ferramentas profissionais disponibilizadas no final da década de 80 como por exemplo, Telelogic da Suécia (Package SDT) e mais recentemente pela Verilog (Package GEODE) da França.

A linguagem SDL foi aqui escolhida por apresentar essas vantagens e principalmente pelo fato de ter disponibilizado a partir de sua versão de 1992 (SDL'92), recursos de orientação a objeto, que possibilitou um casamento mais adequado com um projeto de objetos do sistema a ser especificado para eventuais modificações, resusos e evoluções. Neste trabalho de tese foi utilizado a ferramenta *SDT* (SDL Design Tool) que emprega recursos gráficos da linguagem SDL para construir as especificações, tornando suas realizações mais simples de serem modeladas e entendidas e num prazo de tempo menor. Além dessas vantagens, a ferramenta SDT (descrita em resumo no apêndice B), apresenta recursos como validação, simulação das especificações e prototipação pela geração de código C++, o que garante características eficientes de verificação e de correções das especificações.

A versão mais recente da linguagem SDL apresenta recursos orientados a objeto. Como na maior parte das linguagens de programação, há uma necessidade de empregar uma técnica para planejar as classes que serão implementados.

Para auxiliar no planejamento de classes de objetos, existem várias metodologias, entre as mais conhecidas estão a de Booch [9], Jacobson [39] e Rumbaugh [45]. Neste trabalho foi empregado a metodologia OMT (Object Modeling Technique) proposta por Rumbaugh, para auxiliar no planejamento das classes a serem especificadas e também para auxiliar a documentação do trabalho especificado, e também por ser um dos modelos mais conhecidos, sendo assim possível de ser entendido por uma grande parte dos desenvolvedores.

A adoção do modelo proposto na metodologia deste trabalho não limita a restrição ao uso do modelo OMT, outros modelos propostos como o de Booch e Jacobson entre outros, poderiam ser utilizados em seu lugar. A adoção de um modelo está relacionado à afinidade do desenvolvedor a entender o modelo proposto. O modelo OMT também é proposto em alguns trabalhos, como do projeto europeu Insyde [42] [43], desenvolvido pela Alcatel, Verilog S.A. e algumas Universidades, que empregam essa metodologia para que seja aplicada a linguagens formais como VHDL e SDL na modelagem de sistemas.

O emprego da linguagem SDL com recursos orientados a objeto, combinada a uma metodologia de desenvolvimento de projetos orientado a objeto, resulta num projeto completo [50], com documentação das especificações, possibilitando um melhor entendimento, e o acompanhamento da evolução também por outros eventuais desenvolvedores.

A construção de um sistema envolve várias fases para se obter um resultado consistente e conciso. A modelagem de um sistema envolve a realização de uma análise, um estudo do projeto até chegar a sua implementação.

A metodologia OMT[45] é uma técnica empregada para a realização de uma análise orientada a objeto (OOA - Object Oriented Analisys). O desenvolvimento de projeto baseado em objetos trata o sistema de forma abstrata e independente da linguagem de programação, podendo assim, servir como um meio para especificação, análise, documentação, interface e programação. Dessa forma, essa metodologia pode ser adotada como meio de se construir a análise orientada a objeto para ser combinada com a especificação formal dos sistemas utilizando recursos orientado a objeto como apresentado na linguagem SDL'92 [36] [50].

A proposta de Rumbaugh constrói três modelos para se realizar a análise, que pode ser complementada à especificação em SDL'92:

1. *Modelo de Objetos* Esse modelo é representado graficamente por diagramas de objetos, que são organizados em níveis hierárquicos compartilhando estruturas e comportamentos comuns.

2. *Modelo Dinâmico* representa os aspectos temporais e comportamentais *de controle* de um sistema. Esse modelo é representado graficamente por diagramas de estados. Cada um desses diagramas mostra a seqüência de estados e de eventos permitidos em um sistema para uma classe de objetos. Esse modelo pode ser descrito em SDL e em diagramas MSC (Message Sequence Chart) [38], que mostram graficamente o funcionamento temporal dos sistemas.

3. *Modelo Funcional* representa os aspectos relativos às transformações *de funções* de um sistema. Esse modelo é representado por meio de diagramas de fluxo de dados, que mostram as dependências entre valores e o processamento dos valores de saída a partir dos valores de entrada e das funções, independente de quando ou se as funções são executadas. Na especificação em SDL, os processos descrevem o comportamento do sistema por meio de símbolos semelhantes a diagramas de fluxo de dados, podendo ser utilizado como modelo funcional.

Do modelo OMT foi utilizado o modelo de objetos para se estabelecer as dependências das classes, os atributos necessários e as operações que as classes devem executar. O modelo dinâmico e funcional pode ser documentado pelo uso dos diagramas MSC e na própria linguagem SDL que apresenta recursos gráficos disponíveis por essa linguagem e descritas nas padronizações do ITU-T Z.120 [38] e Z.100 [36] [37] respectivamente.

A utilização do modelo OMT aplicada a linguagem SDL'92, auxilia o desenvolvedor a projetar um sistema com maior coerência, visto que a maior parte das linguagens de programação orientadas a objeto não possuem recursos para realizar a análise, construir uma documentação para gerenciar o reuso, e realizar procedimentos para se construir classes de objetos bem projetadas. Dessa forma, a combinação de uma linguagem formal orientada a objeto como SDL'92 e uma metodologia como OMT, que orienta a especificação adotando essa linguagem, traz grandes vantagens ao desenvolvedor que os adota.

A metodologia que combina a utilização OMT e SDL'92 é utilizada para obter um sistema bem planejado, documentado para que possa ser compreendido por qualquer desenvolvedor. Após

a construção do modelo de objeto contendo as classes, atributos e métodos, é possível construir a especificação formal utilizando a linguagem SDL'92. Os conceitos existentes na OOA (Object Oriented Analysis) podem ser mapeados para mais de um conceito em SDL'92 (Ver apêndice A). O mapeamento proposto neste trabalho de tese entre OMT e SDL'92 é apresentado na tabela 3.1 .

OMT	**SDL'92**
Classe	*Process Type* (Alternativamente ADT - Abstract Data Type)
Objeto	Instâncias de *Process Type* (Alternativamente ADT - Abstract Data Type).
Atributos	Variáveis de processos (apenas *Process Type*).
Serviços	Sinais, pois pode enviar conteúdo dentro de sinais.
Entrada/Saída Externa	Sinais ao ambiente.
Create	Parâmetros formais e transição de estado de início de processo.
Grupos	*Block Type*. Essa estrutura pode conter um ou mais *Process Type*.
Cenário	Diagrama MSC (Message Sequence Chart)que visualiza graficamente o comportamento dinâmico do sistema.
Generalização/Especialização	Herança usando um sistema ou partes desse sistema. Uso do *package* para modularizar as partes em SDL para especializar somente as que serão herdadas.
Relação de Objetos (Descrito nessa metodologia como modelo de objetos)	Pode ser mostrada para blocos e processos, através da visualização hierárquica do sistema e dos blocos.
Diagramas de Serviços (descrito como um módulo de cada classe do modelo de Objetos)	Descrição informal dos processos para descrever a funcionalidade da classe. Uso de comentário em SDL.

Tabela 3. 1 - Diagrama de migração de OMT para SDL'92

Esse mapeamento da tabela 3.1 foi utilizado para realizar as especificações dos protocolos propostos descritos nesse capítulo. As notações e as características principais do modelo OMT e da linguagem SDL podem ser encontradas nos apêndices C e A respectivamente.

3.2.1 - Modelo de objetos em OMT

Os protocolos IP Clássico e NHRP (Next Hop Resolution Protocol) foram especificados utilizando características de orientação a objeto da linguagem SDL'92, para obter as vantagens proporcionadas por esse recurso. Visando construir classes com maior aproveitamento, algumas

características são necessárias para construção do projeto de objetos, assim, foi empregada a metodologia OMT (Object Modeling Technique) [45] para auxiliar no desenvolvimento das especificações.

Após o estudo dos protocolos, a análise dos sistemas a serem especificados e o projeto de objetos, obteve-se as classes necessárias para a realização da especificação do estabelecimento de conexão nos protocolos IP Clássico e NHRP. O projeto de classes resultou em um modelo que contém as características mínimas para a realização do estabelecimento da conexão. A partir da constatação do protocolo NHRP ser baseado no protocolo IP Clássico [3], pôde-se obter as classes com as características básicas e similares entre esses dois protocolos para que essas classes pudessem ser reutilizadas nas especificações posteriores em SDL.

O primeiro passo na construção de um projeto orientado a objetos, é definir as classes e as características necessárias para a especificação dos sistemas. Esse passo é conseguido a partir da análise das entidades necessárias para a construção dos protocolos de interconexão de redes locais e ATM. As entidades encontradas para a construção desse particular projeto de objetos são:

1. *Host* - gerencia os pedidos de conexão vindas do(s) usuário(s) para ser enviado para a subrede, ou recebe as enviadas a ele.

2. *Subrede* - gerencia os pedidos vindos do *Host* e do servidor, caso o host destino pertença à mesma subrede, esse sinal não necessitará sair dessa subrede não sobrecarregando o servidor.

3. *Servidor* - recebe o pedido de conexão de outro servidor através de roteadores, ou de uma de suas subredes. Consulta a memória cache para verificar os endereços. Verifica se o sinal deve ser enviado a uma das subredes que gerencia ou se o sinal deve ser encaminhado ao próximo servidor.

4. *Router* - encaminha o sinal ao próximo nó reconheça o seu sinal.

5. *Cache* - contém os endereços necessários para a consulta e, se necessário atualização, para a realização da resolução de endereço.

Através da definição das entidades envolvidas e do conhecimento do funcionamento do estabelecimento da conexão entre dois *hosts*, foi construído o modelo de objetos, apresentado na Fig. 3.1.

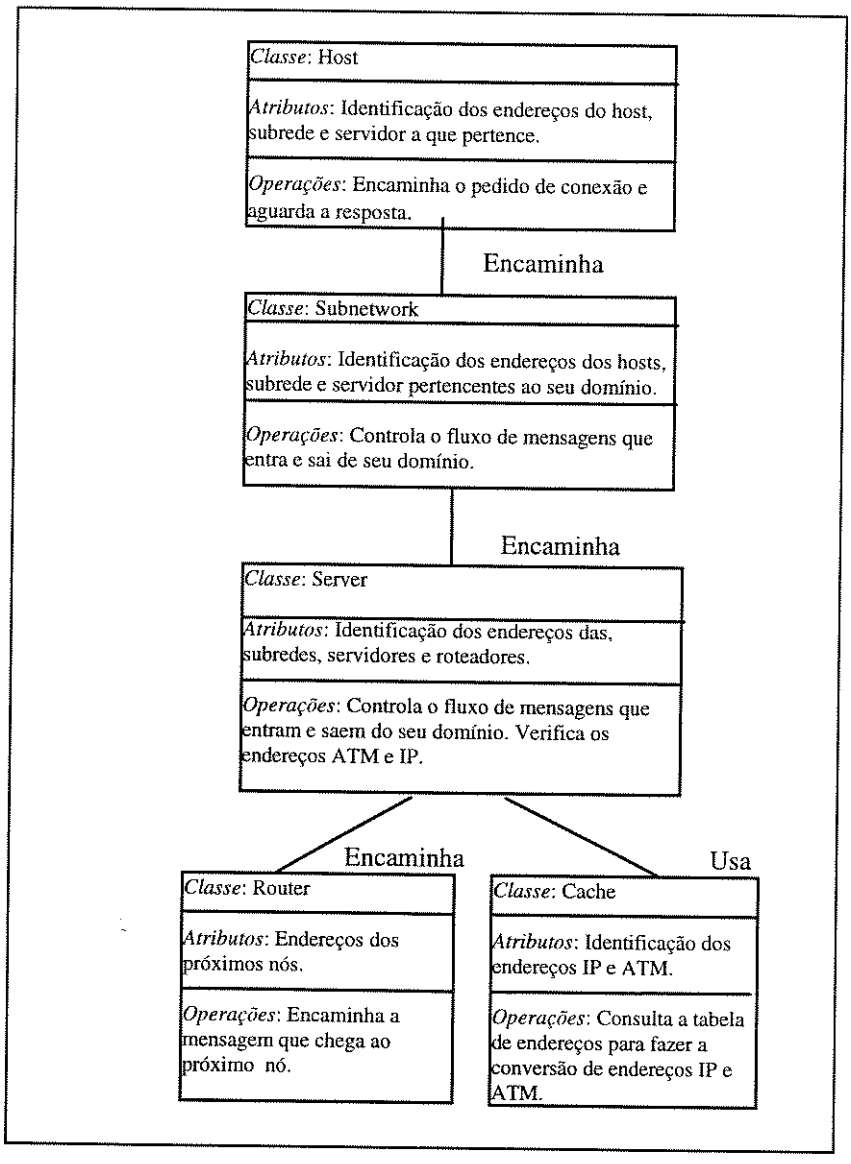

Figura 3. 1 - Modelo de objeto utilizado para especificação de sistemas

As classes apresentadas na Fig. 3.1 foram implementadas em SDL'92 primeiramente para formar o Sistema *Cloud*, que generaliza o funcionamento dos protocolos IP Clássico e NHRP tal que possa ser aproveitado nas especificações formais dos protocolos, assim como acompanhar a evolução desses. Os sistemas especificados referentes aos protocolos IP Clássico e NHRP, assim como o acréscimo de novas características a esses sistemas, utilizando conceitos orientado a objeto, e apresentados neste capítulo são:

1. ***Sistema Cloud,*** que apresenta uma generalização dos protocolos IP Clássico e NHRP, para que possa ser utilizada nos sistemas que especificam esses dois protocolos.

2. ***Sistema IP Clássico,*** que especifica o protocolo IP Clássico, proposto pelo RFC1577[28].

3. ***Sistema NHRP,*** que especifica o protolo NHRP apresentado em seus princípios básicos no RFC1577, e nos artigos de Armitage [3] e Ales[1].

4. ***Sistema NHRP utilizando Proxy-ARP,*** herda as características do Sistema NHRP e acrescenta a característica de comunicação entre duas nuvens ATM utilizando o protocolo *Proxy ARP* (Proxy - Adress Resolution Protocol), descritos no RFC1433 [26] e RFC1620[29] .

5. ***Sistema IP Clássico migrado para o protocolo NHRP,*** apresenta uma especificação de um draft [22] para migração do protocolo IP Clássico para o protocolo NHRP. Esse sistema herda características do Sistema Cloud e IP Clássico.

6. ***Sistema IP Clássico Migrado e NHRP,*** apresenta o Sistema IP Clássico Migrado que se comunica a uma nuvem ATM NHRP através do protocolo Proxy-ARP para realizar a comunicação entre duas nuvens ATM.

A ordem de apresentação dos sistemas propostos nesta tese, é referente à cronologia de desenvolvimento destes. A hierarquia de herança, obtida da reutilização das classes pode ser vista na Fig. 3.2.

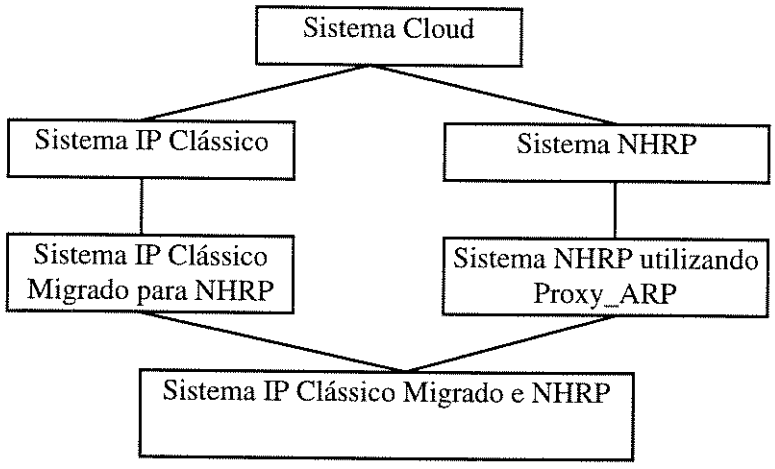

Figura 3. 2 - Hierarquia de herança entre os Sistemas Especificados

A hierarquia apresentada na Fig. 3.2 descreve os níveis de herança dos sistemas especificados em SDL. Nas próximas seções são apresentados os sistemas herdados e como foram realizados em SDL.

3.3 - Especificação em SDL'92

As especificações apresentadas neste trabalho possuem a característica de conexão através de um canal virtual permanente (PVC - Permanent Virtual Channel), com classe de serviço do tipo B, que de acordo com a tabela 2.1 é do tipo orientado a conexão para troca de mensagens e apresetnta taxa de bit variável para que possa ser empregado na transmissão de tráfego de áudio e vídeo.

Os sistemas que representam as especificações dos protocolos, apresentam a característica de serem especificados de acordo com o funcionamento da camada de rede, admitindo a montagem dos quadros e verificação de erros de transmissão que são realizadas nas camadas inferiores à nível de bit.

O endereço IP (Fig. 2.4) apresentado no capítulo 2 (descrição das arquiteturas IP e ATM), foi representado em um tipo abstrato de dados (ADT - Abstract Data Type) na linguagem SDL para a formatação dos sinais necessários para a construção das especificações. A Fig. 3.3 exemplifica a declaração desse ADT declarado na linguagem SDL'92.

```
NEWTYPE Addresses STRUCT
MA Pr_Type;
Network Pid;
Subnetwork Pid;
Host Pid;
ENDNEWTYPE Addresses;
```

Figura 3. 3 ADT utilizado na representação de endereço IP

Através da utilização das classes de objetos já planejadas (Fig. 3.1), foi criado inicialmente o Sistema *Cloud*, que apresenta características dos protocolos IP Clássico e NHRP (Next Hop Resolution Protocol), para que se pudesse aproveitar as características em comum entre esses protocolos e não necessitar modelar repetidamente as mesmas características em cada um dos sistemas.

3.3.1 - Sistema Cloud

O Sistema Cloud especifica de maneira abstrata o funcionamento da interconexão entre dois *hosts*. Esse sistema contém as classes necessárias para que se possa realizar a interconexão, implementadas de tal forma para que possam ser utilizadas pelos sistemas que especificam o funcionamento do protocolo IP Clássico e NHRP.

O mapeamento das classes apresentadas no modelo de objetos (Fig. 3.1), para suas representações em SDL no Sistema Cloud é apresentado na tabela 3.2.

Classes OMT	***Representação em SDL***
Host	*Process type Host*, apresentado dentro do *Block Type Host*.
Subnetwork	*Process type Subnetwork*, apresentado dentro do *Block type Subnetwork*.
Server	*Process type Server*, apresentado dentro do *Block type Server*.
Router	*Process type Router*, apresentado dentro do *Block type Server* por ser uma atividade controlada pelo Servidor.
Cache	*Process type Cache*, apresentando os processos realizados na memória cache, dentro do *Block Type Cache*.

Tabela 3. 2- Mapeamento entre o modelo OMT e SDL para Sistema Cloud.

A utilização das construções em SDL'92 *System Type, Block Type* e *Process Type*, são empregados para a reutilização dessas partes em outros sistemas (Apêndice C). Para a reutilização dos sistemas, blocos ou processos em SDL, é usada a construção *Package*, que armazena as classes que poderão ser reutilizadas em outros sistemas. O *Package* é uma biblioteca de classes que pode ser herdada por outros sistemas, blocos ou processos de acordo com sua hierarquia utilizada na linguagem. O emprego do *Package* em cada classe, pode ser útil para a construção de sistemas que possuem características de vários outros sistemas. A construção de *Packages* de partes menores como blocos e processos, determina uma maior modularidade, e um sistema que necessite apenas das características de determinadas classes, poderá utilizá-las sem necessitar herdar características de que não lhe serão úteis, eliminando características indesejáveis

Os *System Types*, não podem ser compilados, para que possam ser simulados e validados; essa estrutura deve ser englobada em um *Package*, como mostra a Fig. 3.4, e herdada em um sistema para que possa ser compilado. Esse sistema (Fig. 3.5) apresenta apenas a cláusula ***USE*** e o nome do Package utilizado, nos sistemas que irão herdar esse sistema através do Package, é apresentado na área textual do sistema (retângulo tracejado nas figuras), se utiliza a cláusula *inherits* e o nome do *System Type*; esses serão os casos do Sistema IP Clássico e o Sistema NHRP.

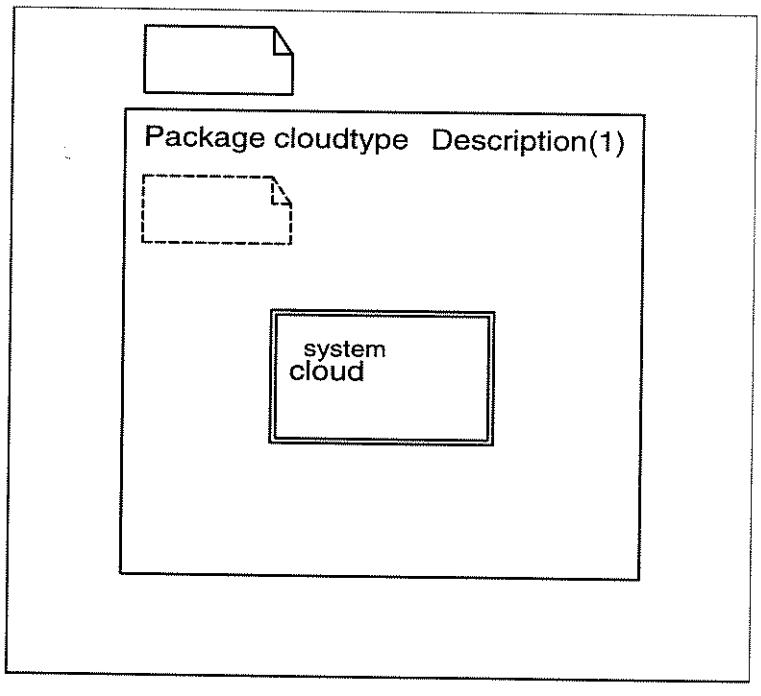

Figura 3. 4 - Package do Sistema Cloud

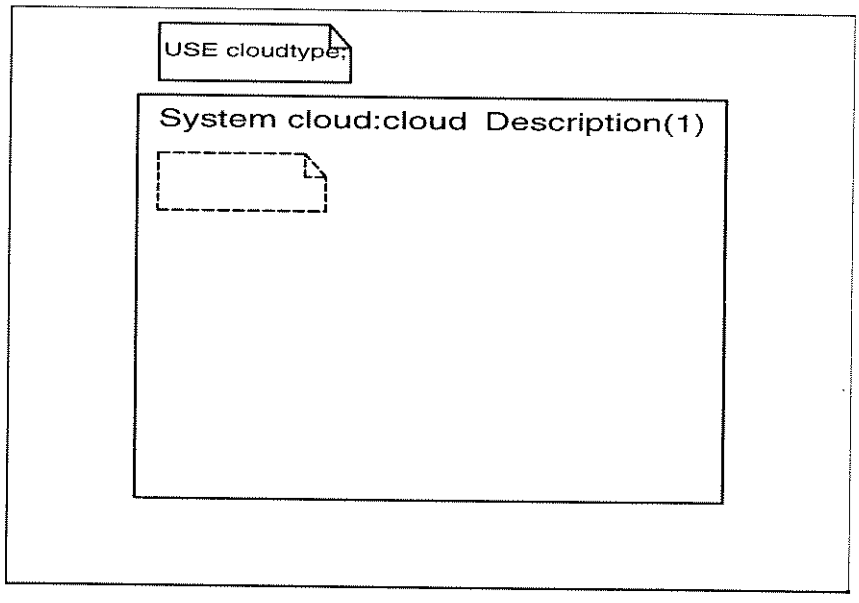

Figura 3. 5 - Sistema que utiliza o Package para compilar o Sytem Type Cloud

A construção *Package* também é utilizada para modularizar o sistemas em pequenos *packages* para que determinadas partes possam ser herdadas por outros sistemas, eliminando os aspectos desnecessários.

Os sinais utilizados possuem quadros, ou seja, campos dos sinais que são formatados como ADT (Abstract Data Type) e são herdados pelos Sistemas IP Clássico e NHRP. Esses tipos abstratos de dados são declarados utilizando *Newtype* nas especificações em SDL. A Fig. 3.6 mostra as declarações dos sinais e das construções dos tipos abstratos de dados para o Sistema Cloud. A figura apresentada apenas visualiza os sinais declarados a nível de sistema. Os sinais internos a cada bloco são declarados dentro do respectivo bloco.

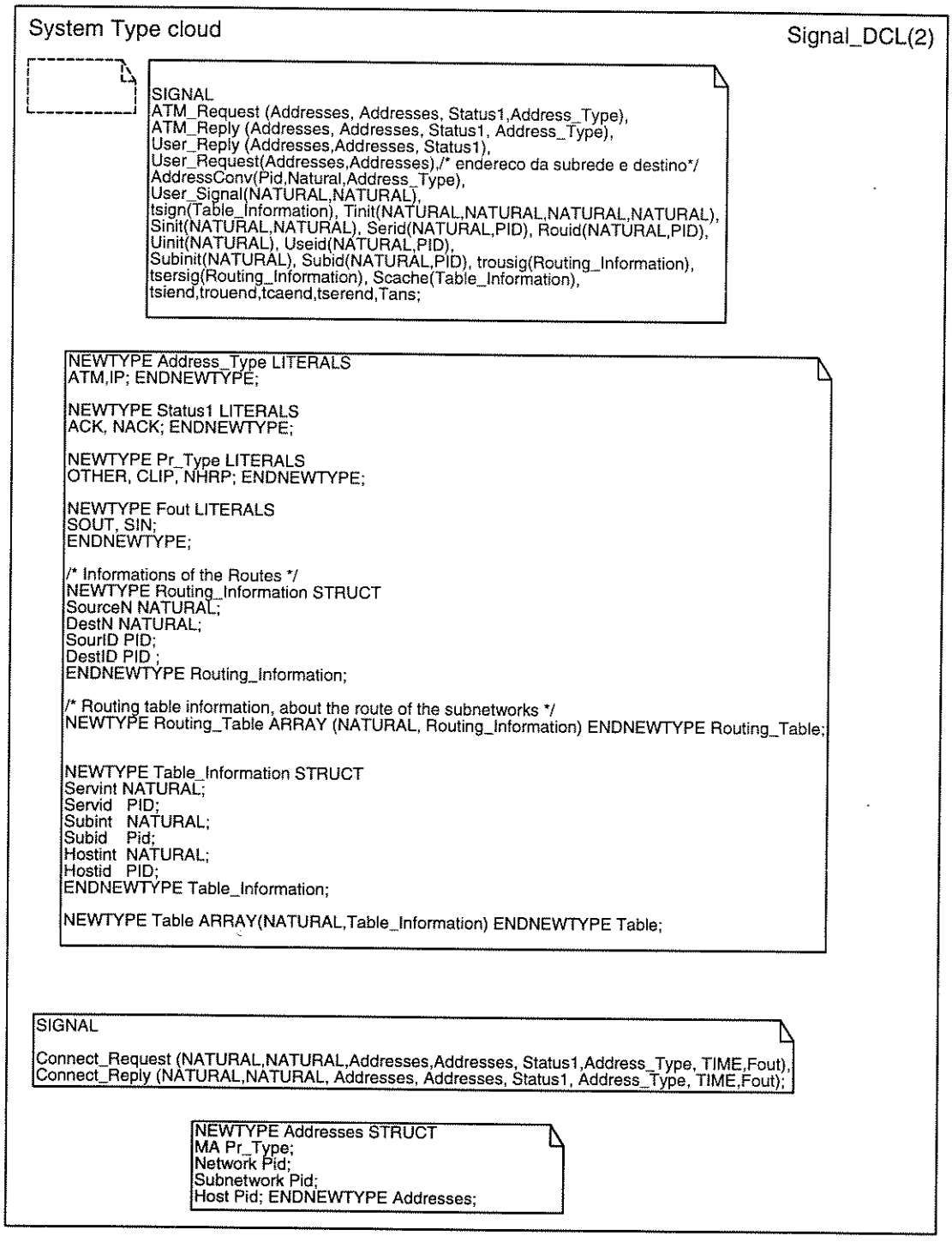

Figura 3. 6 - Declaração dos sinais utilizados no Sistema Cloud

O Sistema *Cloud* apresenta a implementação das classes definidas no modelo de objetos (Fig. 3.1). Esse sistema em SDL é mostrado na Fig. 3.7. A migração do modelo OMT para SDL da tabela 3.2 é visível nesta figura. O Sistema *Cloud* foi baseado no funcionamento dos protocolos IP Clássico e NHRP (Next Hop Resolution Protocol). A partir da constatação de que o modelo do protocolo NHRP é baseado no funcionamento do protocolo IP Clássico [3], o Sistema *Cloud,* descreve as classes com o funcionamento similar entre esses dois protocolos.

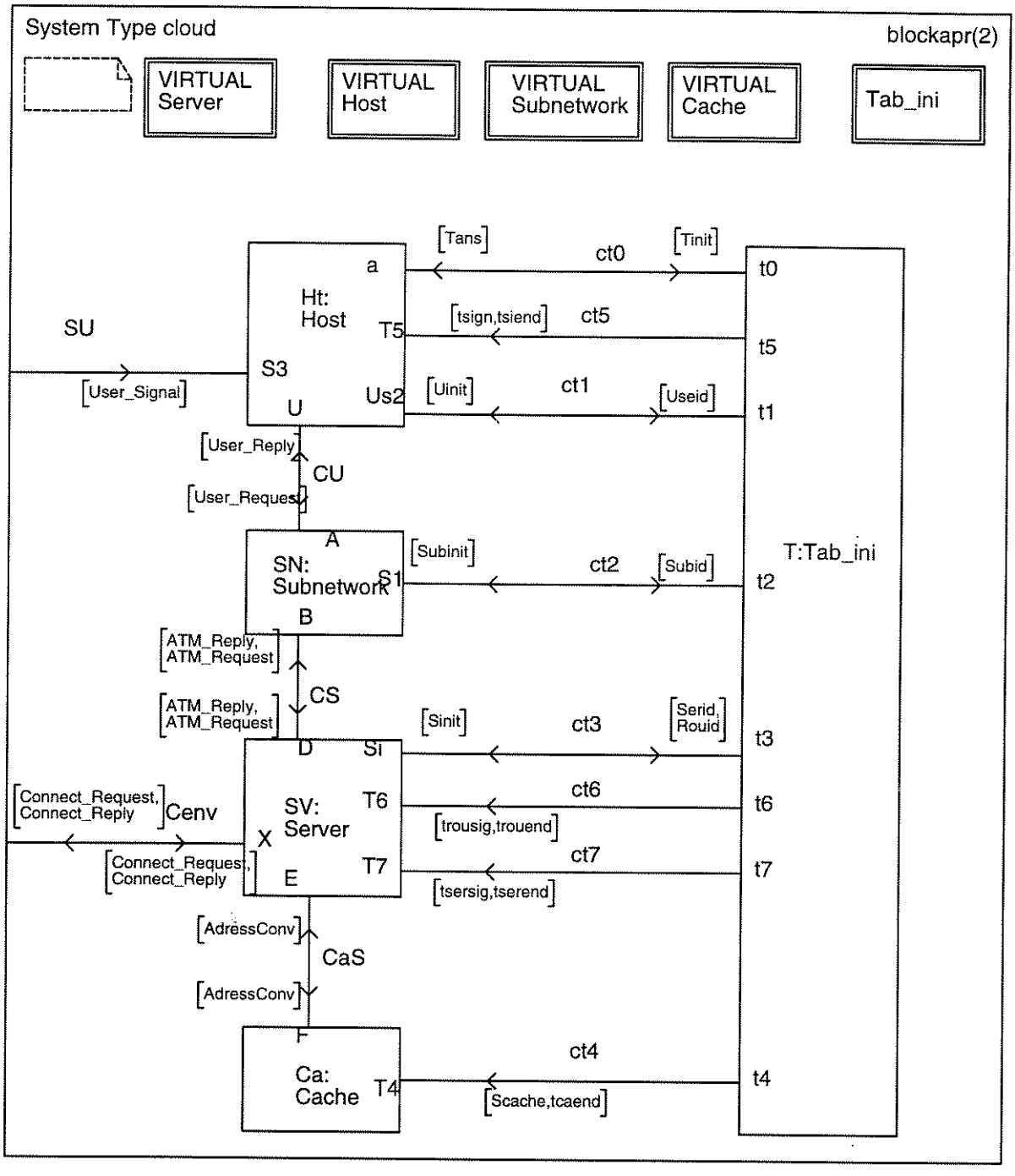

Figura 3. 7 - Especificação em SDL'92 do Sistema Cloud.

As classes descritas na Fig. 3.1 foram transformadas para SDL, de acordo com a tabela 3.2. Essas classes que foram implementadas na forma de *Process Type* são descritas dentro dos *Block Types* apresentados no Sistema *Cloud* (Fig. 3.7). O *Block Type Tab_Ini* (Fig. 3.8) não é apresentado como classe no modelo de objetos (Fig. 3.1), pois esse processo não será modificado em nenhum dos sistemas, ele tem apenas a função de criar as tabelas em suas devidas entidades, terminando com o processo após a criação dessas. O processo *Inicial_ta* (Ver Anexo D, Fig. D.1.16), especificada dentro do *Block Type Tab_Ini* da Fig. 3.8 é importante para a determinação

da configuração da rede a ser simulada, assim como na geração das tabelas, empregadas no roteamento e endereçamento. O *Block Type Tab_Ini* possui conexão com todos os outros blocos para obter os endereços dos processos, já que os endereços *(Pid)* são reconhecidos dentro dos blocos, necessitando passá-los por sinais para o *Block Type Tab_Ini* para a criação das tabelas utilizadas no Sistema *Cloud*.

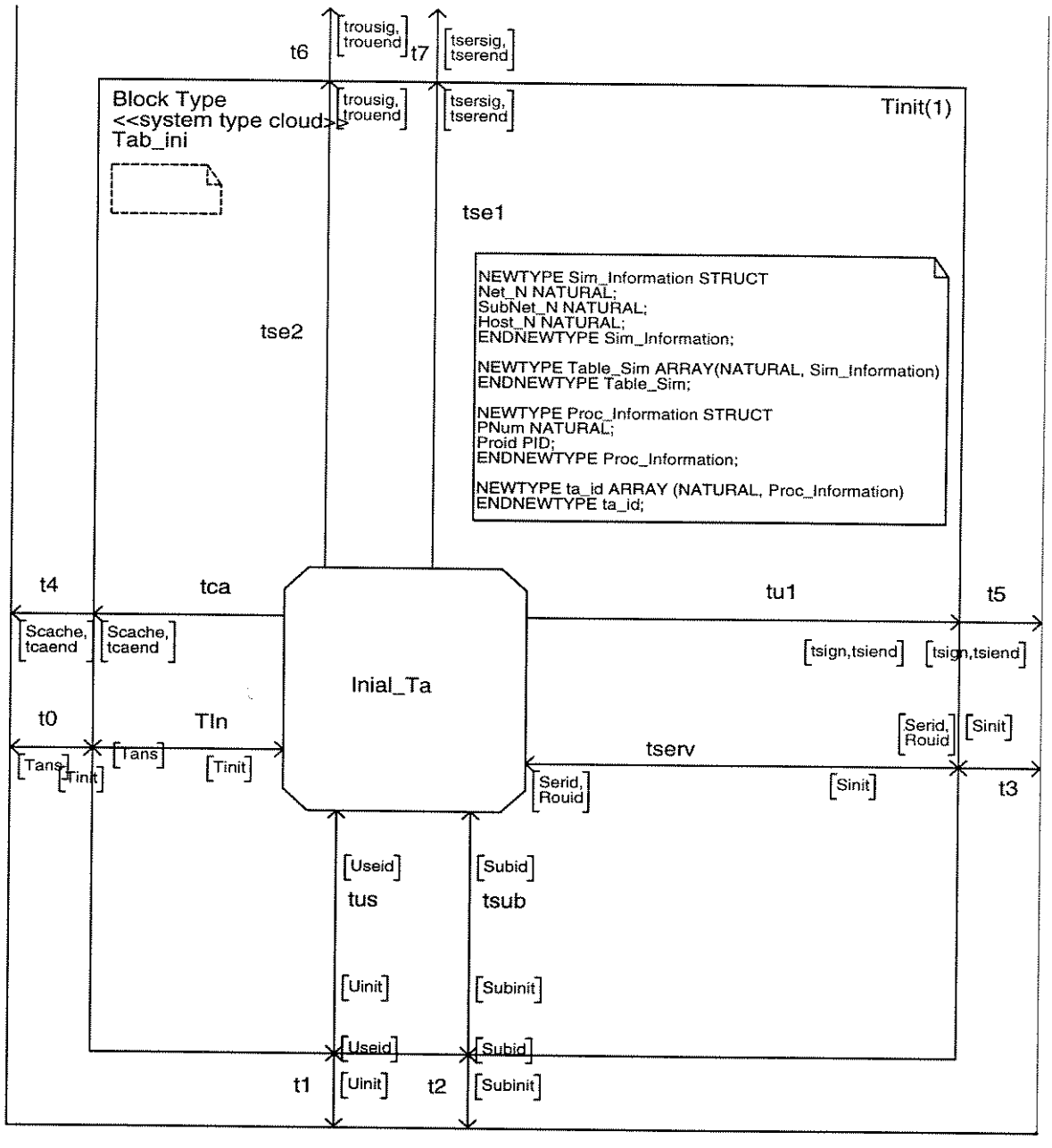

Figura 3. 8 - Bloco Tab_Ini do Sistema Cloud

Um *Block Type* através de seus *Process Type* pode representar um conjunto de classes do modelo de objetos. Apenas os processos declarados como *Type* são os que podem ser definidos como classes, pois podem ser herdados, redefinidos e intanciados. A Fig. 3.9 apresenta o *Block*

Type Cache do Sistema Cloud (Fig. 3.7) que contém o *Process Type Cache_Consulting*, declarado como Virtual para que possa ser redefinido.

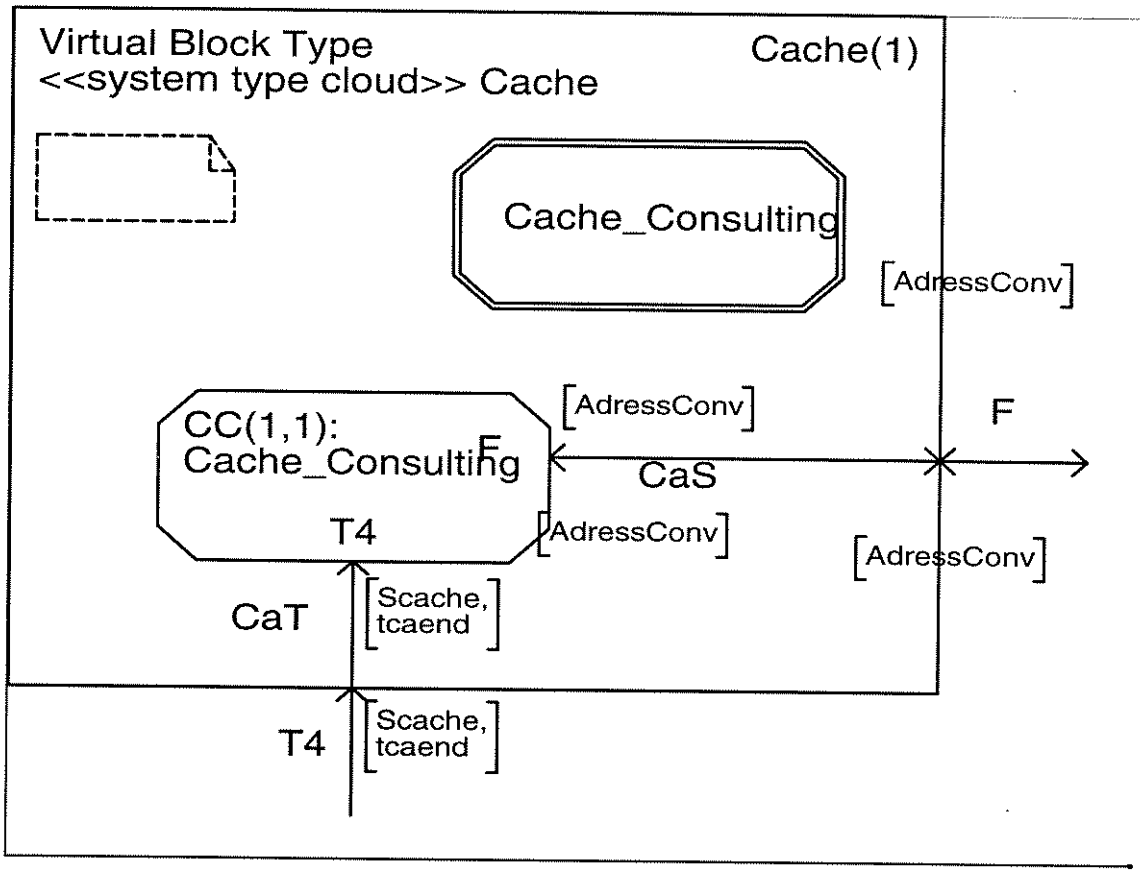

Figura 3. 9 - Block Type Cache do Sistema Cloud

Cada classe apresentada e especificada, representa o funcionamento das entidades necessárias para realizar o estabelecimento de conexão entre redes locais, através do meio ATM. O *Block Type Host* (Fig. 3.10), é um bloco que pode ser instanciado por ser declarado como *Type.* Esse bloco contém o *Process Type Host* que descreve o funcionamento de um *host*. O *host* recebe o pedido de conexão e encaminha esse pedido à devida subrede a que está submetida. O pedido enviado pelo *Process Type Host* é encaminhado ao *Process Type Subnetwork* que a contém, de acordo com a configuração da rede estipulada na inicialização do sistema, definida pelas tabelas; e esse, encaminha o pedido ao Servidor. Esse pedido circulará pela nuvem, de servidor a servidor, até que seja encontrado o destino pretendido.

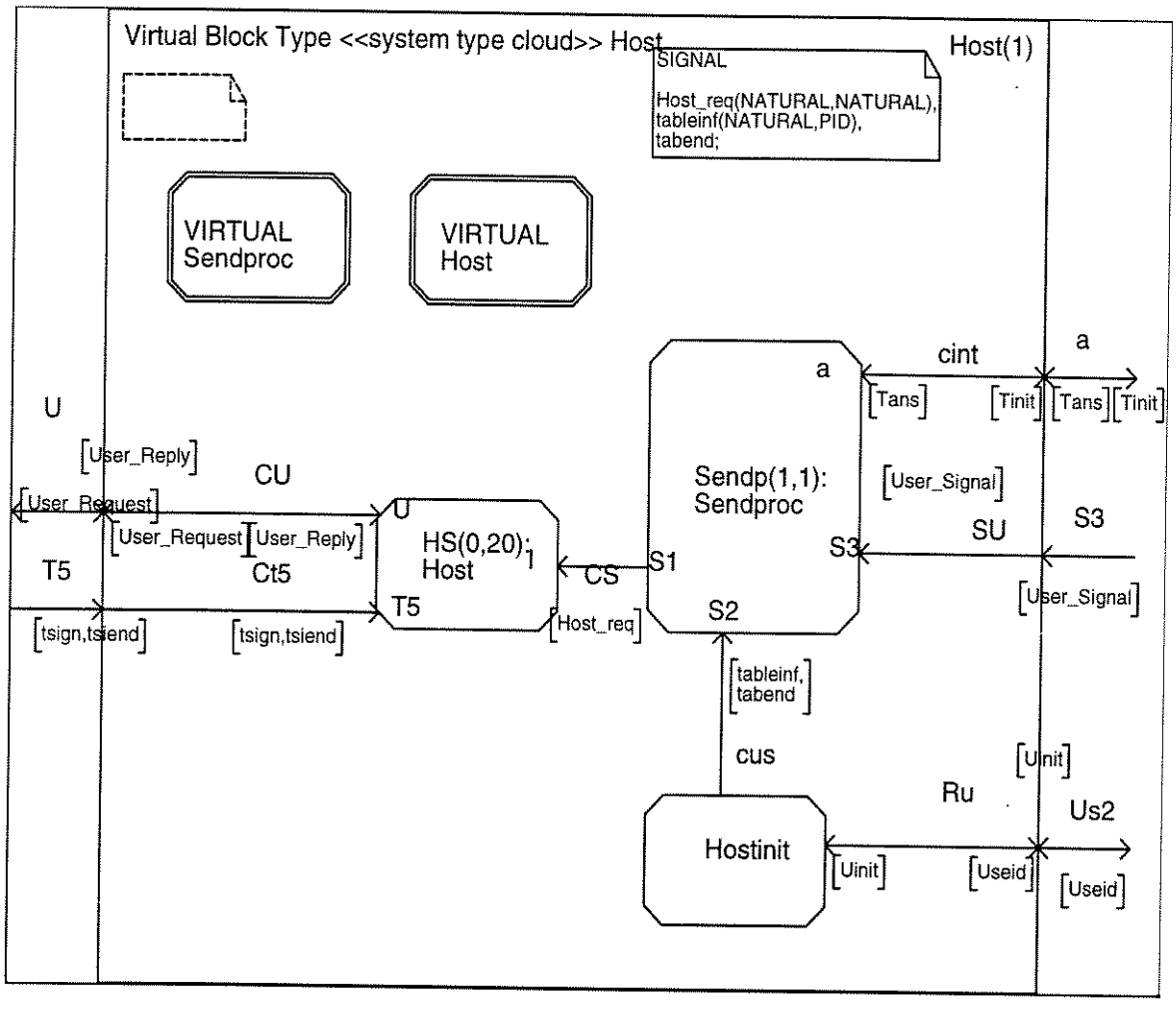

Figura 3. 10 - Bloco Host do Sistema Cloud

O *Block Type Subnetwork* (Fig. 3.11) possui os processos de controle da subrede (*Process Type Subnetwork)* e também o da criação das instâncias (*Process SubNinit)* para a geração de endereços (*Pid)* para que sejam enviados ao processo de inicialização das tabelas. Esse processo existe apenas durante a criação dos endereços, pois, após realizar sua função não é mais necessário manter esse processo. Esse processo é referente ao *constructor* no conceito de orientação a objeto, pois é esse processo que irá gerar as instâncias do *Process Subnetwork*, para que sejam gerados os *Pids* da subrede.

Todos os blocos especificados em SDL, apresentam um processo simples de inicialização (poe ex.: *Hostinit, SubNinit, SerInit*), esses processos são utilizados na criação das instâncias (*Host, SubNetwork, Server* e *Router*) para a criação dos endereços desses processos, e para que possam ser utilizados como endereços IP de cada membro da rede. Como pode ser visto nas nas figuras dos blocos Host, Subnetwork e Server, Fig. 3.10, 3.11 e 3.12 respectivamente, os processos Host, Subnetwork , Server e Router, apresentam números em parênteses dentro da declaração de

seus processos, esses números indicam o número de instâncias com que começam a partir do início do sistema, e quantas instâncias esses processos poderão admitir. No caso do processo Host, mostrado no bloco Host da Fig. 3.10; no início de seu sistema não existe nenhuma instância desse processo, e poderá assumir, num máximo de 20 processos durante o sistema. O processo *Hostinit*, é o processo de inicialização dessa classe, ele é quem irá criar as instâncias desse processo, gerar os *Pids* necessário, encaminhar ao processo *Inicial_ta*, para que esse possa gerar as tabelas e distribuí-las aos processos *Host, Subnetwork, Server* e *Router* gerados.

Assim como o processo *Hostinit*, o processo *SubNinit* é responsável por gerar as instâncias do processo Subnetwork, e o processo *SerInit*, é responsável de gerar as instâncias dos processos servidor e roteador. Esses processos de inicialização são finalizados em SDL após terem gerado as entidades necessárias para a simulação do sistema. Durante a validação e simulação dos sistemas, seria como se esses processos nunca tivessem existido, assim, como o bloco Tab_Ini, que só existe no início do sistema até a criação das instâncias.

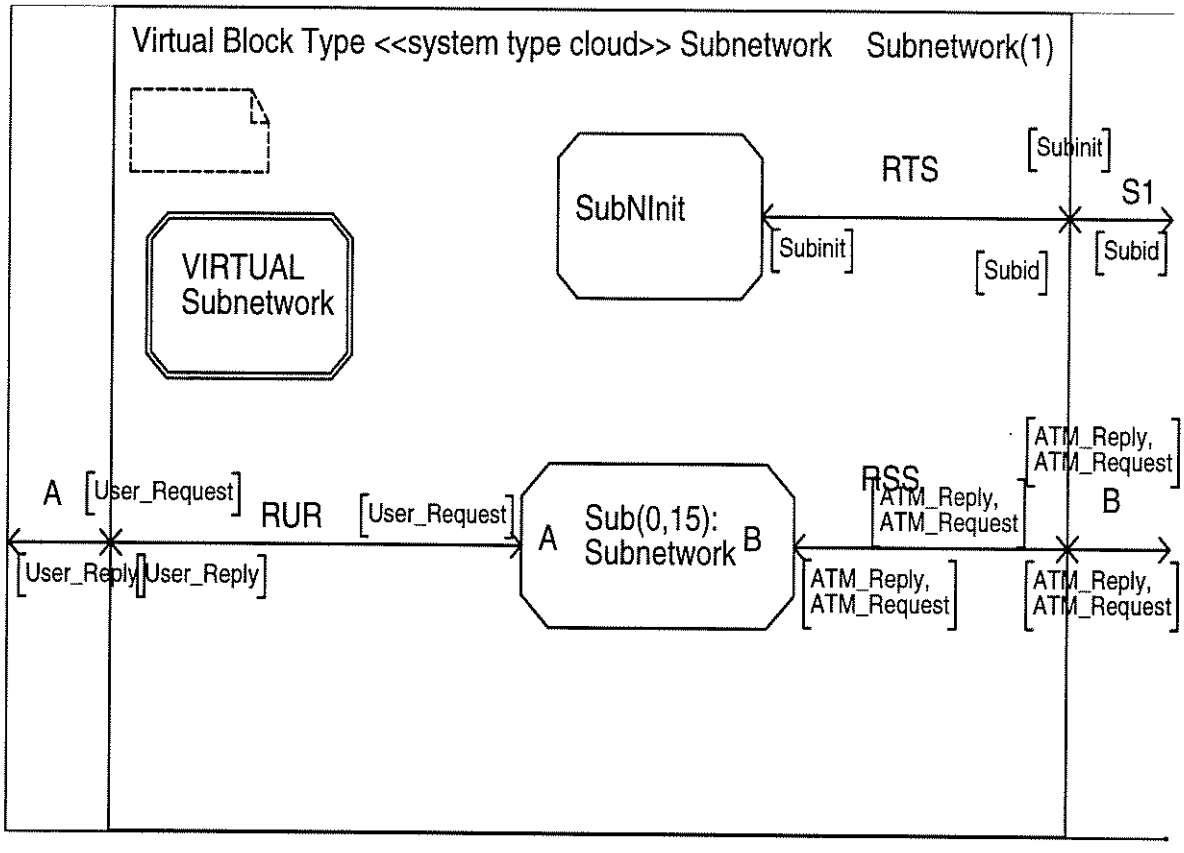

Figura 3. 11 - Bloco Subnetwork do Sistema Cloud

O bloco Subnetwork (Fig 3.11) do Sistema Cloud possui o *Process Type Subnetwork* que tem seu funcionamento implementado de forma abstrata para que pudesse ser modelado como uma

LIS (Logical IP Subnetwork) e NBMA (Non Broadcast Multi Access) no caso dos protocolos IP Clássico e NHRP.

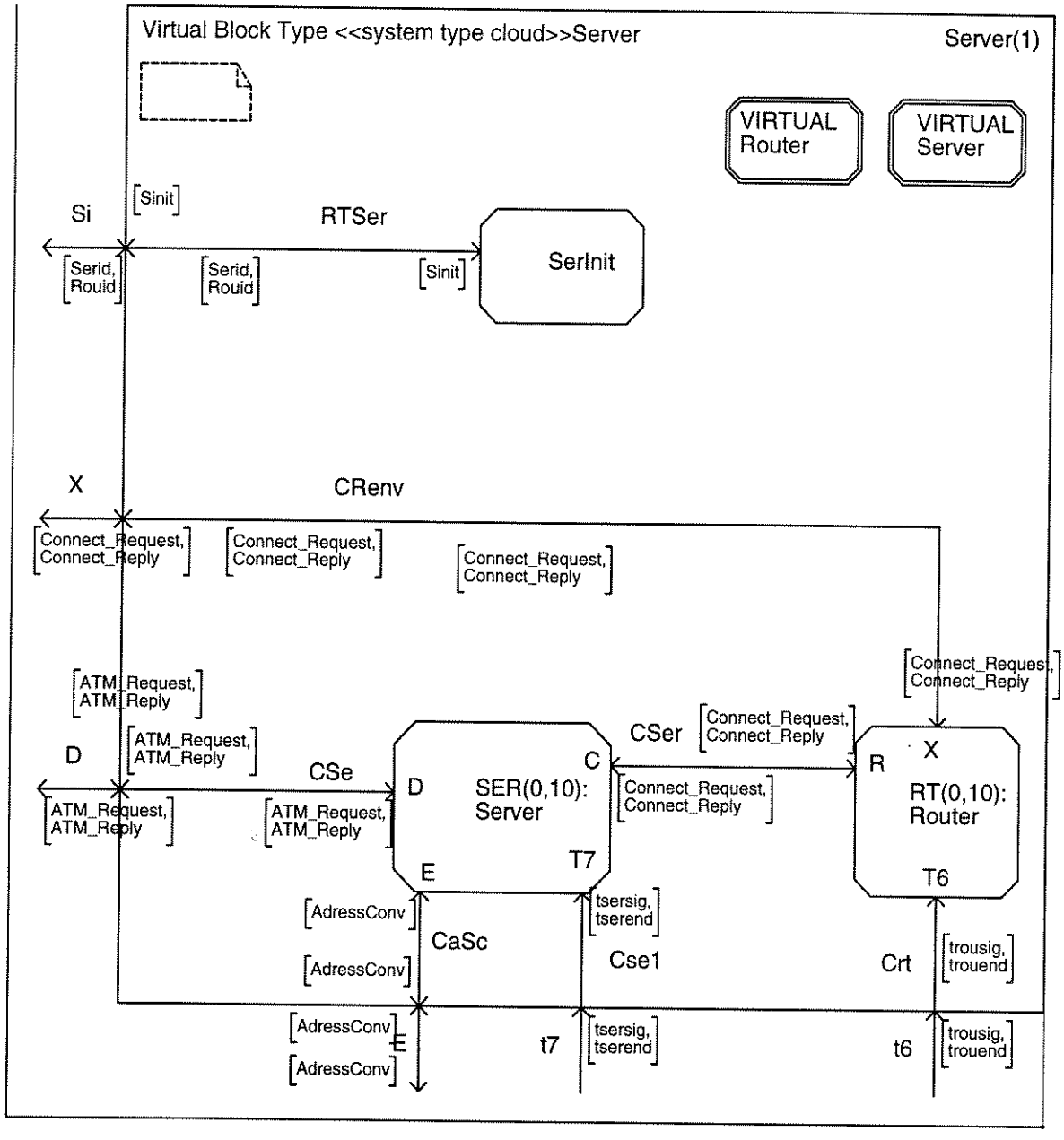

Figura 3. 12 - Bloco Servidor do Sistema Cloud

O *Block Type Server* (Fig. 3.12) apresenta três processos: o SerInit, que funciona como *constructor* do objeto Servidor, o *Process Type Router* e *Process Type Server.* O *Process Type Server* (Fig. 3.13) foi modelado abstratamente para que possa ser herdado pelos sistemas que especificam os protocolos IP Clássico e NHRP modelados. O *Process Type Router* (Fig. 3.14) foi modelado dentro do *Block Type Server* (Fig. 3.12) pois nas propostas dos RFCs, o roteamento é

uma tarefa realizada pelo servidor. Nas especificações dos sistemas, cada bloco representa assim as entidades necessárias para a discussão do pedido de conexão entre as redes.

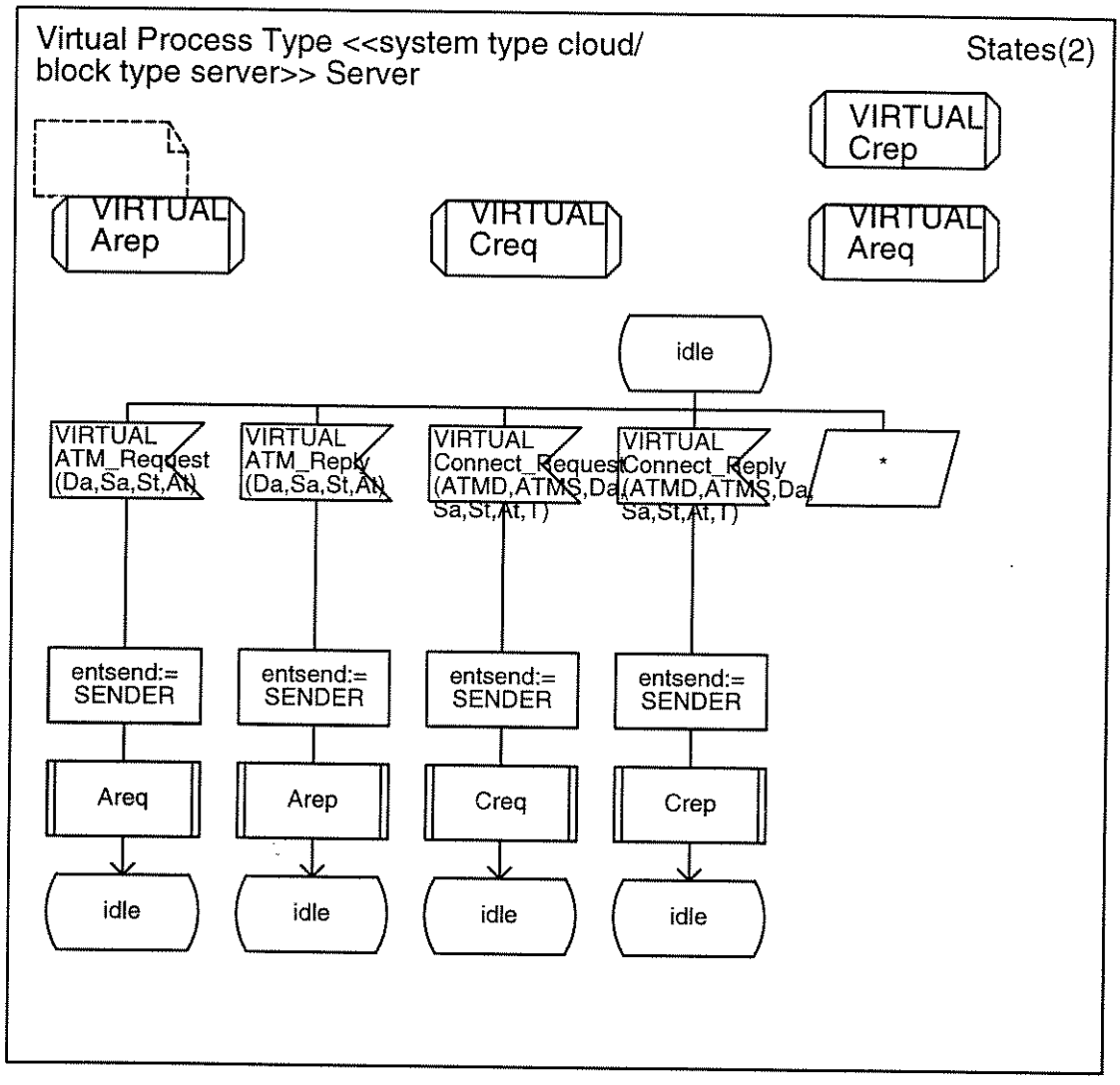

Figura 3. 13 - Processo Servidor do Sistema Cloud

O objetivo da criação das especificações do Sistema *Cloud* foi para que esse tivesse um alto grau de abstração como um dos itens de uma boa especificação de objeto, para que nenhuma de suas características pudesse limitar sua herança em outro sistema.

A Fig. 3.13 descreve uma parte da especificação do processo servidor do Sistema Cloud. A apresentação de um processo fornece dados mais detalhados sobre o funcionamento do sistema especificado. Nesse processo ainda é apresentado uma visão mais detalhada, a de *Procedure*, que também poderá ser redefinida como um processo. A visão hierárquica da linguagem SDL, nos

possibilita ter uma visão geral de um sistema, e entender cada parte separadamente em cada bloco, ou processo.

O processo Roteador (Fig. 3.14) utiliza um procedimento para buscar o próximo nó a ser encaminhado. Esse procedimento *Search_Hop*, procura em sua tabela o próximo nó. O roteamento é realizado a partir desse processo. O uso de procedimento para a busca do nó pode ser útil em uma eventual alteração de roteamento, pois assim, esta aconteceria apenas no procedimento *Search_Hop*.

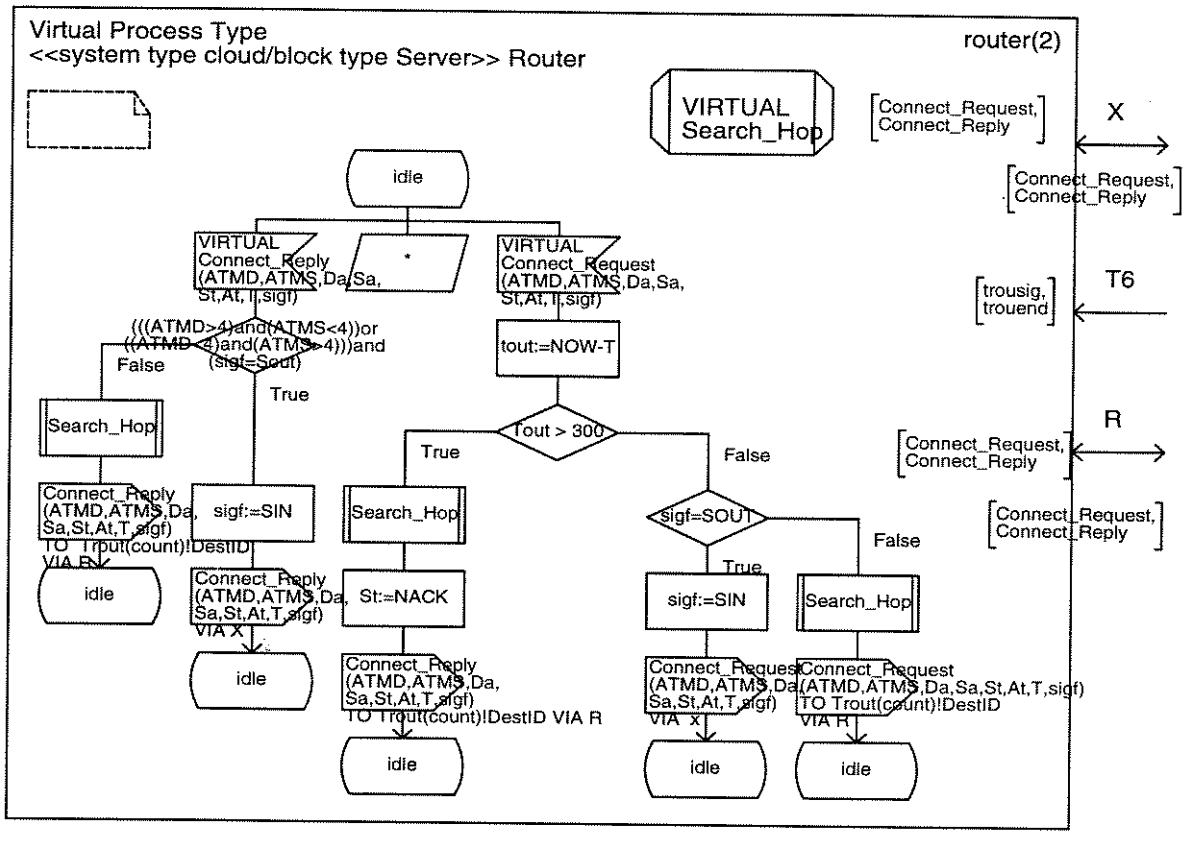

Figura 3. 14 - Processo Router do Sistema Cloud

Na linguagem SDL, havendo a necessidade de redefinir uma determinada parte de um processo, essa poderá ser feita a partir do ponto em que se necessite realizar essas modificações, a partir do ponto superior (estado, procedimento, entre outros) mais próximo que tenha sido declarado como *Virtual*. Se for em um determinado estado, somente é modificado o tratamento de um sinal no estado a ser redefinido, podendo os outros estados permanecerem como na super-classe. A herança de um sistema não limita a adição de novos processos, blocos, sinais e variáveis nos sistemas que herdam suas características.

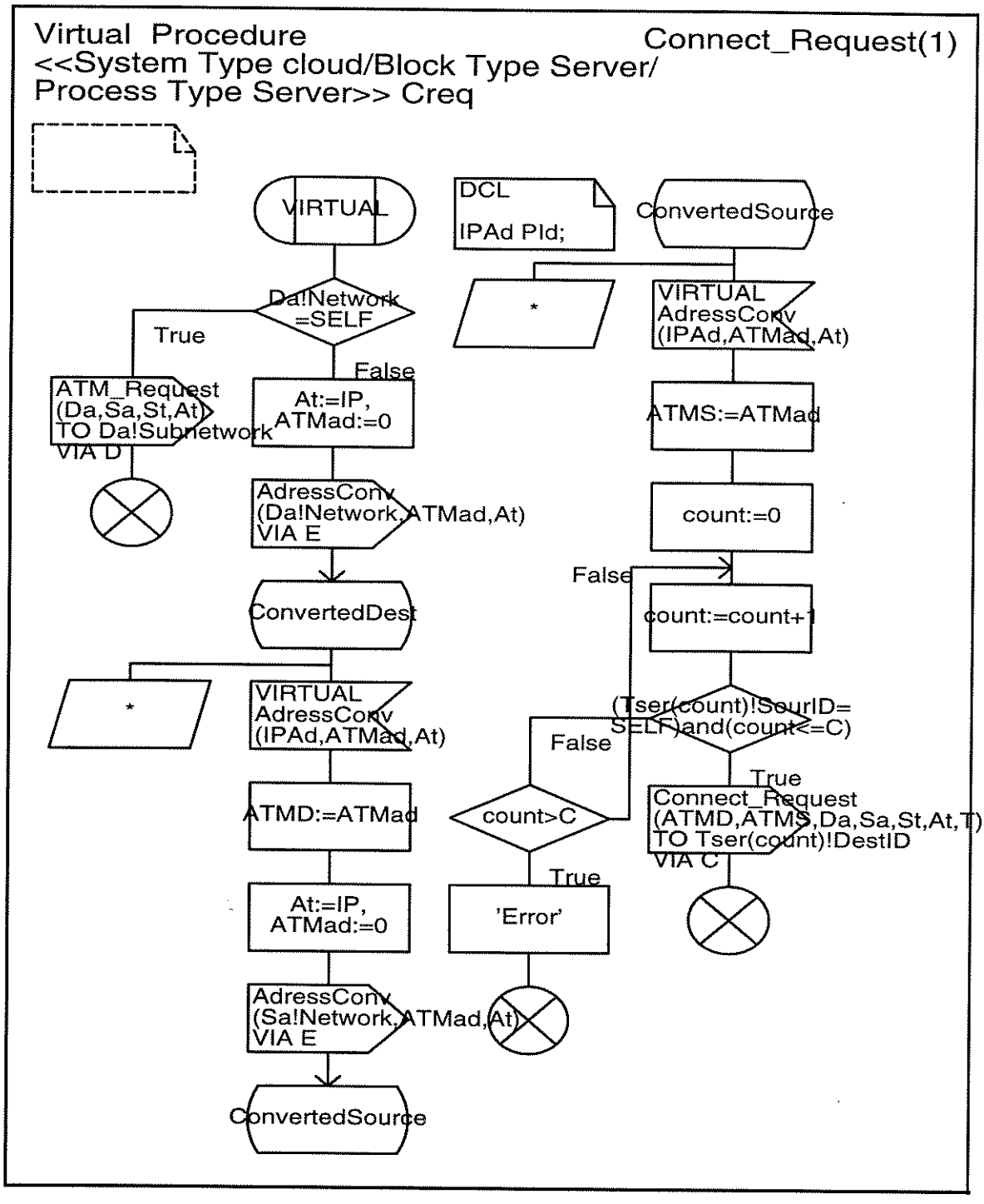

Figura 3. 15 - Procedimento Connect_Request do Sistema Cloud

De fato, o uso de procedimento (*Procedure*) no tratamento de sinal, pode ser uma característica vantajosa em se tratando de herança. No caso do processo Servidor (Fig 3.13), as alterações nos sistemas herdados, foram realizados nos procedimentos, não necessitando alterar o processo todo a partir do estado *idle*. Caso não se utilizasse a estrutura de procedimento, tudo que seguisse esse estado, deveria ser modificado. No processo Servidor o tratamento de sinal, foi realizado empregando os procedimentos *Areq, Arep, Creq* e *Crep*. A Fig. 3.15 descreve a

especificação de um procedimento que trata o sinal de *Connect_Request* recebido por uma entidade servidor (Fig. 3.13) e encaminha através de um router (Fig. 3.14).

As especificações dos sistemas que herdam o Sistema *Cloud*, apresentam apenas as partes em que houve a necessidade de realizar modificações referentes aos estados, sinais, entre outros. A especificação que herda características de outros sistemas constitui em uma especificação sucinta que apresenta apenas as partes alteradas, tanto em bloco como em processo, resumindo assim as novas especificações.

O Sistema *Cloud* proposto nessa seção, é a especificação formal da super-classe dos sistemas que são apresentados nesse trabalho. A herança do Sistema Cloud é empregada nos Sistemas NHRP e IP Clássico, além do desenvolvimento de novas características nas especificações já realizadas para construir um sistema que contém a comunicação de duas nuvens NHRP interconectadas para a implementação e verificação da troca de sinais fora do domínio do sistema, reutilizando as classes já modeladas no Sistema NHRP e Cloud. Essas evoluções do Sistema Cloud e outras são apresentadas nas próximas seções.

3.3.2. - Sistema IP Clássico

O Sistema IP Clássico herda todas as características apresentadas no Sistema *Cloud*, apenas adicionando ou modificando as partes necessárias. A hierarquia desse sistema é apresentada na Fig. 3.16. Através da utilização do termo *virtual* no Sistema *Cloud* é possível alterar partes que possuem esse termo. O adicionamento de novos blocos é sempre possível, mas o adicionamento de novos processos no sistema que está herdando as características depende da cláusula *virtual* no *Block Type* em que ele é especificado, pois a linguagem SDL trabalha em níveis, se um sistema herda outro sistema e o bloco do sistema herdado não permite realizar modificações, então, não será possível modificar tudo o que engloba esse bloco, tornando o bloco inalterável. Esse fato é uma característica que deve ser evitada nas especificações orientada a objeto, pois fere um dos princípios de se construir classes com maior grau de abstração para que possa ser adequada ao ambiente que poderá ser utilizado.

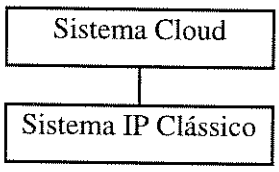

Figura 3. 16 - Hierarquia da Especificação do Sistema IP Clássico

A especificação do Sistema IP Clássico (Fig. 3.17) descreve apenas as alterações necessárias para que o Sistema *Cloud* (Fig. 3.7) apresente características do protocolo IP Clássico. As modificações realizadas são referentes ao adicionamento de novos sinais (*ex.: Inv_ATMARP, ATMARP_Rep, ATMARP_NAK)* entre os blocos, necessitando assim, alterar os blocos que enviam e recebem esses sinais. Os blocos em que houve a necessidade de serem alterados são apresentados em tracejados na nova especificação. Nota-se também que o sistema herda suas características do Sistema *Cloud*, devido a cláusula *"inherits Cloud"* apresentada na declaração do sistema.

Os *System Type, Block Type* e *Process Type*, são apenas as descrições dos tipos que poderão ser herdados, modificados e instanciados. As alterações são realizadas nas estruturas declaradas como *Type* que no Sistema IP Clássico são: *Redefined Server, Redefined Subnetwork, Redefined Host* e *Redefined Cache.* Dessa forma, são apresentados apenas os nomes das instâncias dos blocos (Ht, SN, SV, Ca) a serem modificados no Sistema IP Clássico, como mostrado na Fig. 3.17.

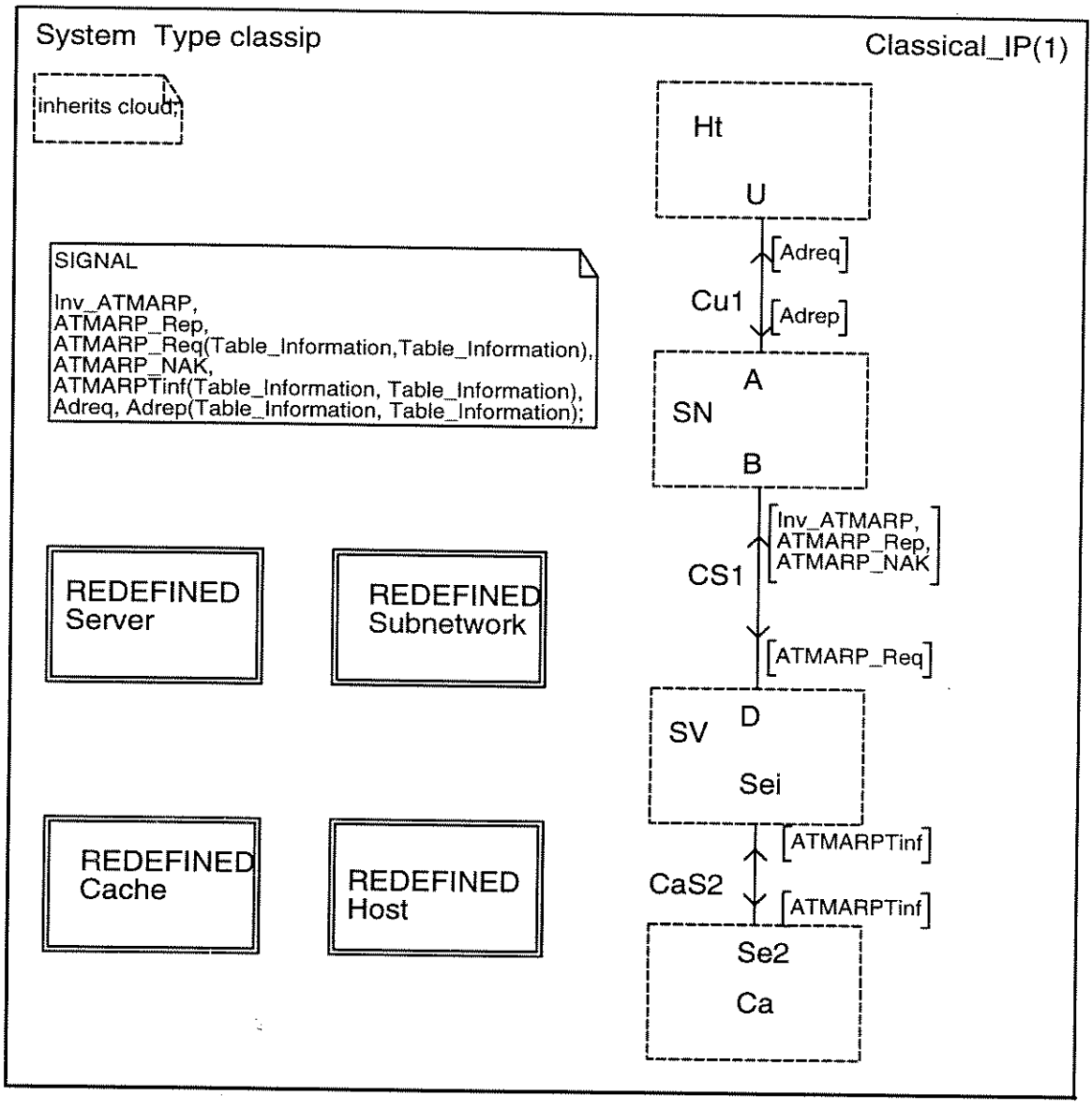

Figura 3. 17 - Sistema IP Clássico

O Sistema IP Clássico sofre modificações nesses blocos devido a adição de novos sinais entre esses blocos, necessitando assim, alterar todos os procedimentos envolvidos nessa troca de sinais, ou adicionar novos processos que tratarão esses sinais novos. Uma característica importante na especificação do sistema novo, está nos *gates* utilizados dentro de cada bloco. Os *gates* utilizados no Sistema IP Clássico são: U, A, B, D, Sei, Se2. Sendo que os *gates* U, A, B e D, são os mesmos utilizados no Sistema Cloud (Fig. 3.7), assim os novos *gates* (Sei e Se2), são criados para a comunicação de novos processos. A criação desses novos *gates* está relacionado a criação de um novo processo no bloco Cache, pois no protocolo IP Clássico existe a necessidade de atualização da Cache .

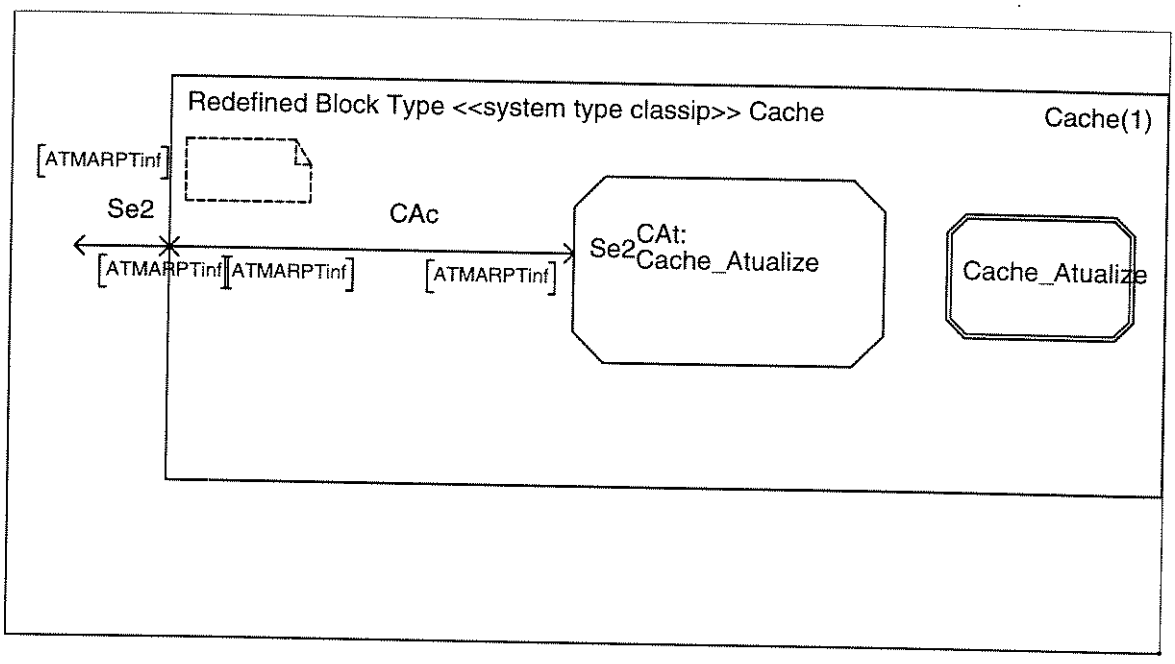

Figura 3. 18 - Bloco Cache do Sistema IP Clássico

Assim como nas redefinições dos blocos na especificação do *System Type*, os processos também são modificados da mesma forma. A redefinição dos blocos permite que se possa alterar os processos. O emprego de *Virtual* nos níveis acima, é que permite que as características dos elementos definidos dentro de cada bloco possa ser redefinido. As alterações nas instâncias (Ca,Ht,SV,SN) dos blocos: *Block Type Cache, Block Type Host, Block Type Server* e *Block Type Subnetwork* do Sistema IP Clássico, podem ser vistas nas Fig. 3.18, 3.19, 3.20 e 3.21 respectivamente.

O tipo (*Type*) determina apenas que um bloco ou processo possa ser herdado e modificado. Uma instância desses blocos e processos é apenas uma cópia do funcionamento desse tipo, possuindo características diferentes. Uma instância de um bloco é uma entidade diferente, é igual apenas no seu conteúdo, todas as instâncias podem ser de um determinado tipo. Nos processos a instanciação funciona da mesma forma apenas que cada processo terá uma identidade (*Pid)* diferente, ou seja um endereço diferente.

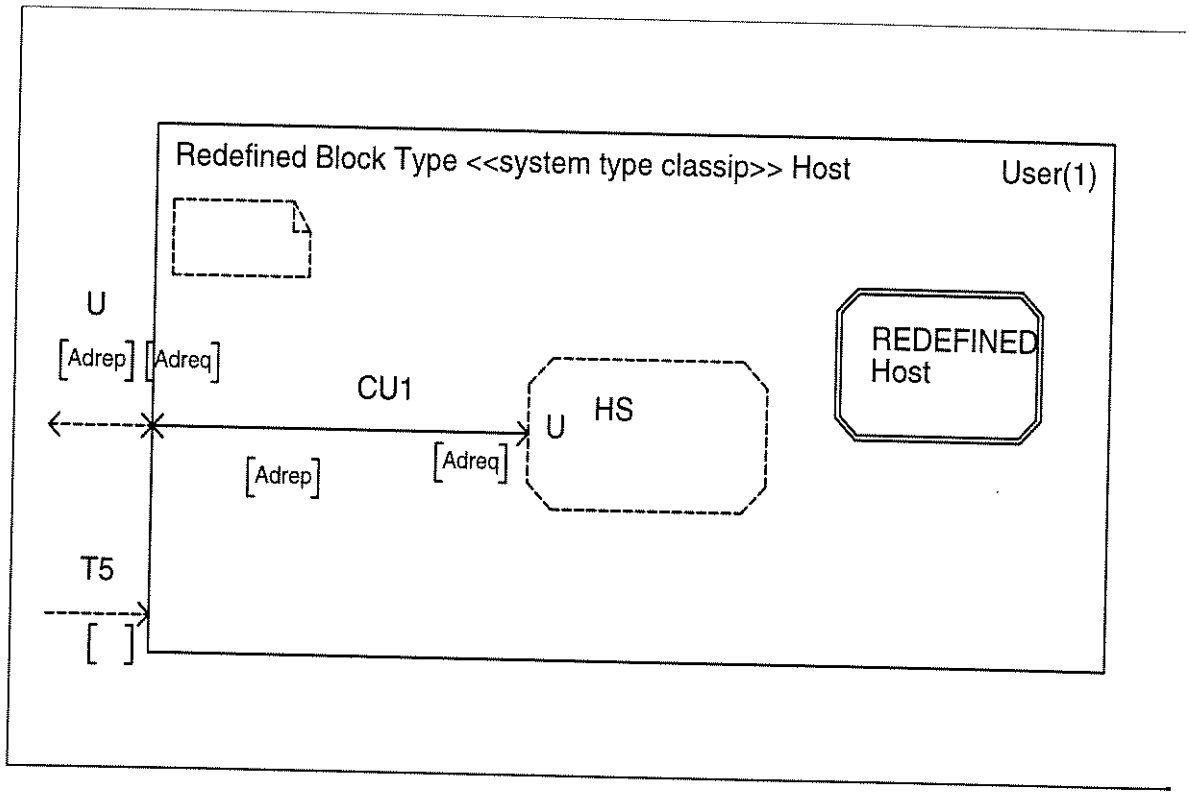

Figura 3. 19 - Bloco Host do Sistema IP Clássico

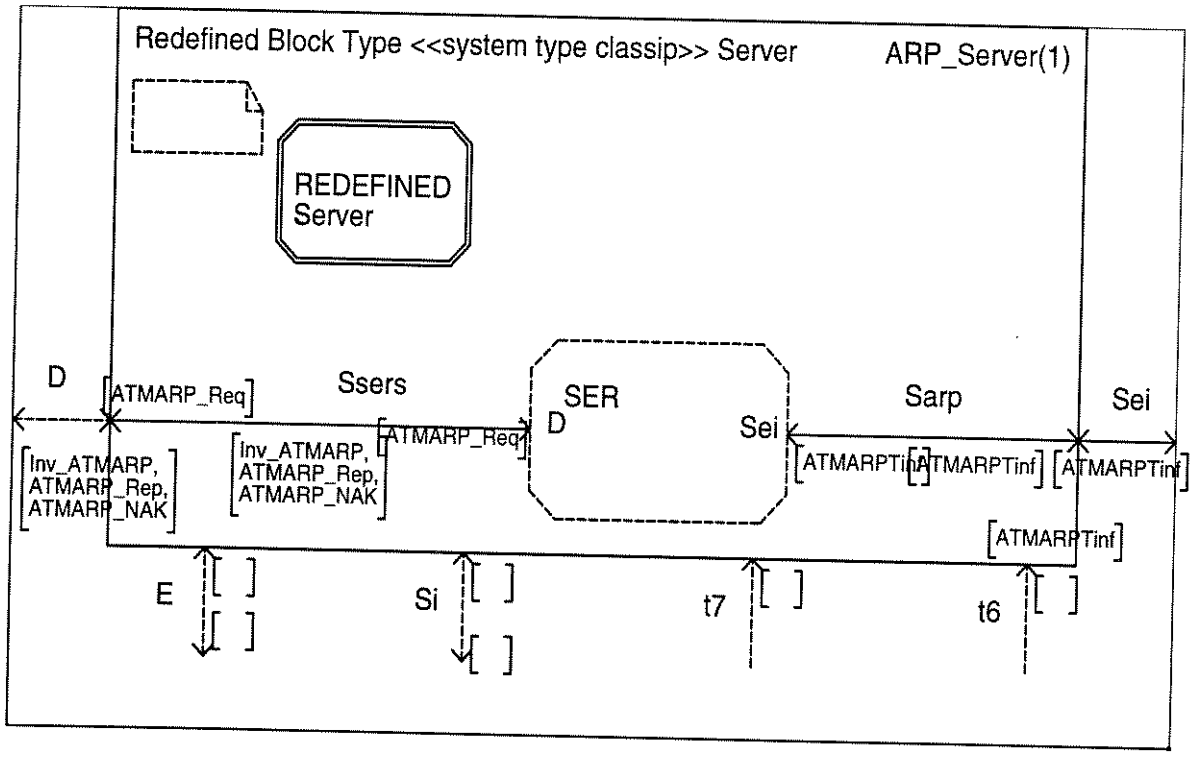

Figura 3. 20 - Bloco Server do Sistema IP Clássico

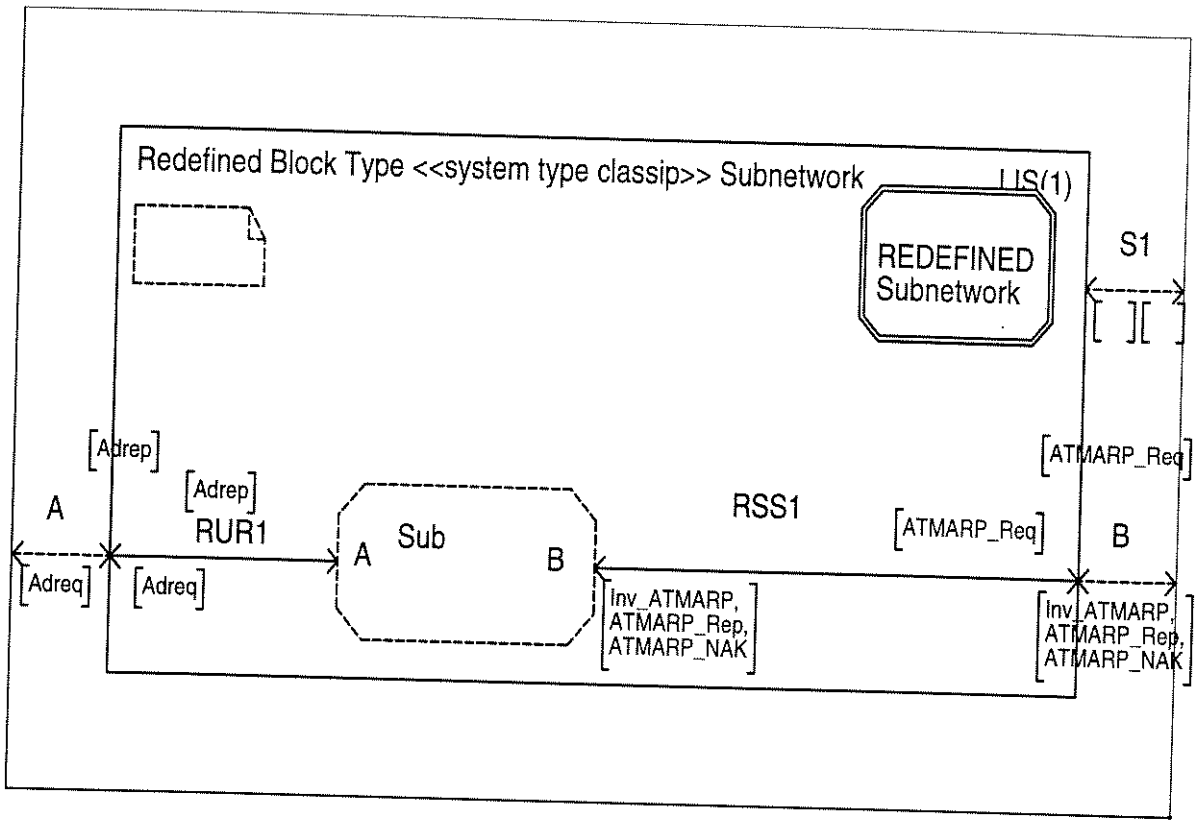

Figura 3. 21 - Bloco Subnetwork do Sistema IP clássico

Os blocos redefinidos apresentam apenas as modificações e a adição de novas características para a especificação do novo sistema. Todos os blocos e processos mostrados no Sistema Cloud permanecem presentes no Sistema IP Clássico, pois esse sistema herdou as características do Sistema Cloud completamente.

O *Process Type Subnetwork* do Sistema IP Clássico(Fig. 3.22), é redefinido de acordo com o funcionamento de uma LIS (Logical IP Subnetwork), e o *Process Type Server* (Fig. 3.23), é redefinido para apresentar características de funcionamento de um servidor ARP. Os processos originais do Sistema *Cloud*, podem ser vistos na Fig. D.1.14 (Apêndice C) e Fig.3.13 respectivamente. Na Fig. 3.22 do processo subrede do Sistema IP Clássico, são apenas acrescentados os tratamentos dos sinais novos (Inv_ATMARP, ATMARP_Rep,ATMARP_Req, ATMARP_NAK, Adrep, Adreq) acrescentados pelos *gates* A e B, enquanto que no processo servidor (Fig. 3.23) desse sistema, é acrescentado um novo *gate* Sei, que se conecta ao Bloco Cache.

Os *gates* apresentados em flechas tracejadas (Fig. 3.18, 3.19, 3.20, 3.21 e 3.23), são os herdados da super-classe Sistema *Cloud* e que permanecem no novo sistema. Os sinais antigos dos

gates nessas figuras ainda permanecem, necessitando apresentar apenas os sinais novos a esses *gates.*

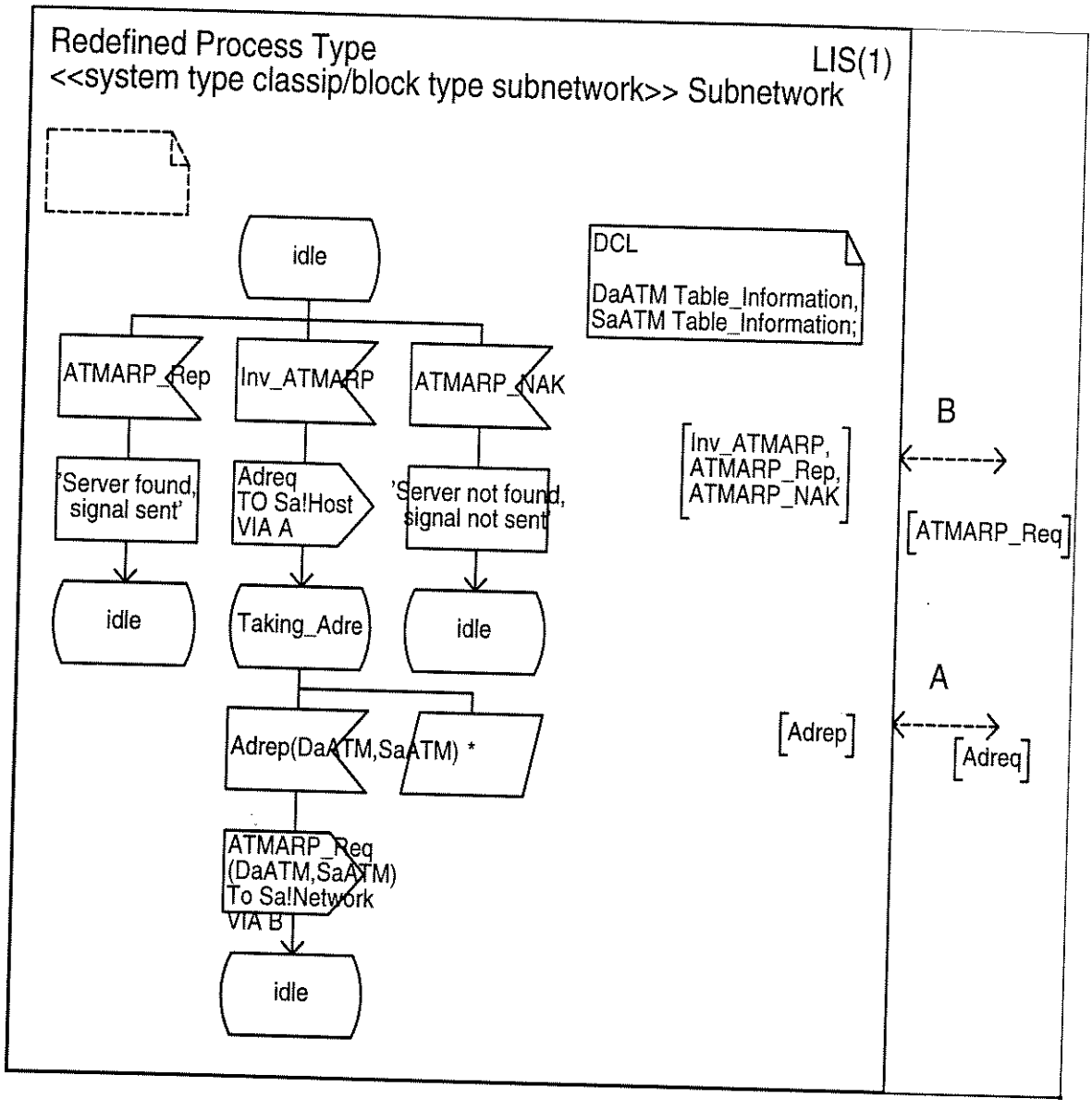

Figura 3. 22 - Processo subnetwork do Sistema IP Clássico

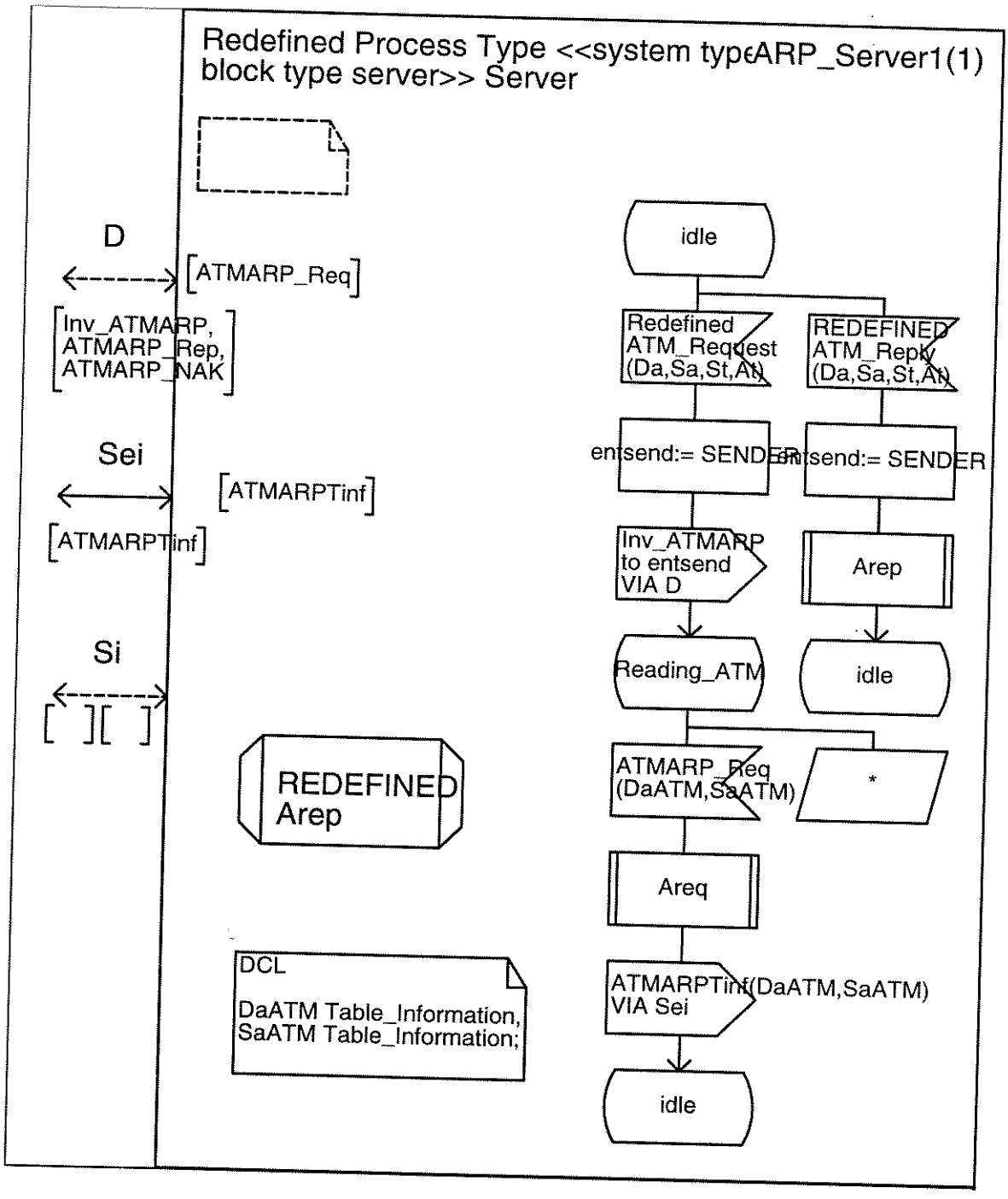

Figura 3. 23 - Processo Servidor do Sistema IP Clássico

No meio de transmissão TCP/IP existe a possibilidade de ocorrer *loops* e também outra característica presente nesse meio é a passagem por *hops* desnecessários. Esse problema está relacionado ao roteamento realizado. O processo de roteamento Internet pode ser implementado de diversas formas como o directed ARP proposto pelo RFC 1433 [26], além desse, existem diversos outros processos de roteamento, como os propostos pelo RFC 1620 [29], que tentam reduzir os números desnecessários de *hops* e também tentam controlar os *loops* encontrados nos protocolos de conexão, entre eles, o protocolo de redirecionamento *Hop-by-Hop* e o Proxy ARP estendido,

que foram estudados para serem utilizados no processo de roteamento dos protocolos IP Clássico e NHRP.

3.3.3 - Sistema NHRP

O Sistema NHRP apresenta o mesmo nível de hierarquia do Sistema IP Clássico, como pode ser visto na Fig. 3.24, que mostra a hierarquia desse sistema. Sua apresentação possui as mesmas características apresentadas no Sistema *Cloud*, adicionando e alterando o funcionamento em alguns aspectos. Assim como no Sistema IP Clássico, suas características foram modificadas para que se especifique as características do protocolo NHRP proposto pelo IETF [21] [30] e revisto no Capítulo 2.

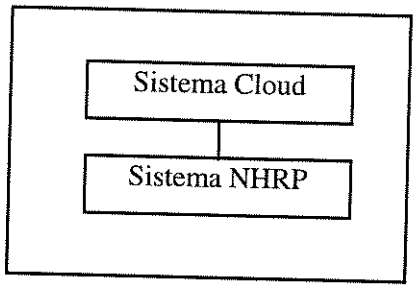

Figura 3. 24 - Hierarquia do Sistema NHRP especificado

A Fig. 3.25 descreve o Sistema NHRP, nele houve alterações apenas nos blocos *Server* e *Subnetwork* em relação ao Sistema *Cloud* (Fig. 3.7). Alguns sinais novos (ATMARP_Rep, ATMARP_NAK, ATMARP_Req) também foram adicionados na redefinição do sistema, situação análoga ao ocorrido no Sistema IP Clássico (Fig. 3.17).

Apesar da especificação do Sistema NHRP apresentar apenas dois blocos modificados (Bloco Servidor e Bloco Subrede), isso não quer dizer que tenha sofrido um menor número de modificações do que o Sistema IP Clássico; a adição de novos sinais nesse último, fez com que um maior número de blocos tivessem que apresentar alterações. No Sistema IP Clássico apenas se acrescentou o tratamento dos novos sinais no estado determinado, não necessitando realizar grandes modificações nas características já existentes. No caso do Sistema NHRP, apesar de terem sido acrescentados menos sinais do que no sistema anterior, algumas alterações também necessitaram ser realizadas, sendo concentradas nos blocos *Subnetwork* e *Server* devido ao processo de roteamento apresentado nesse protocolo.

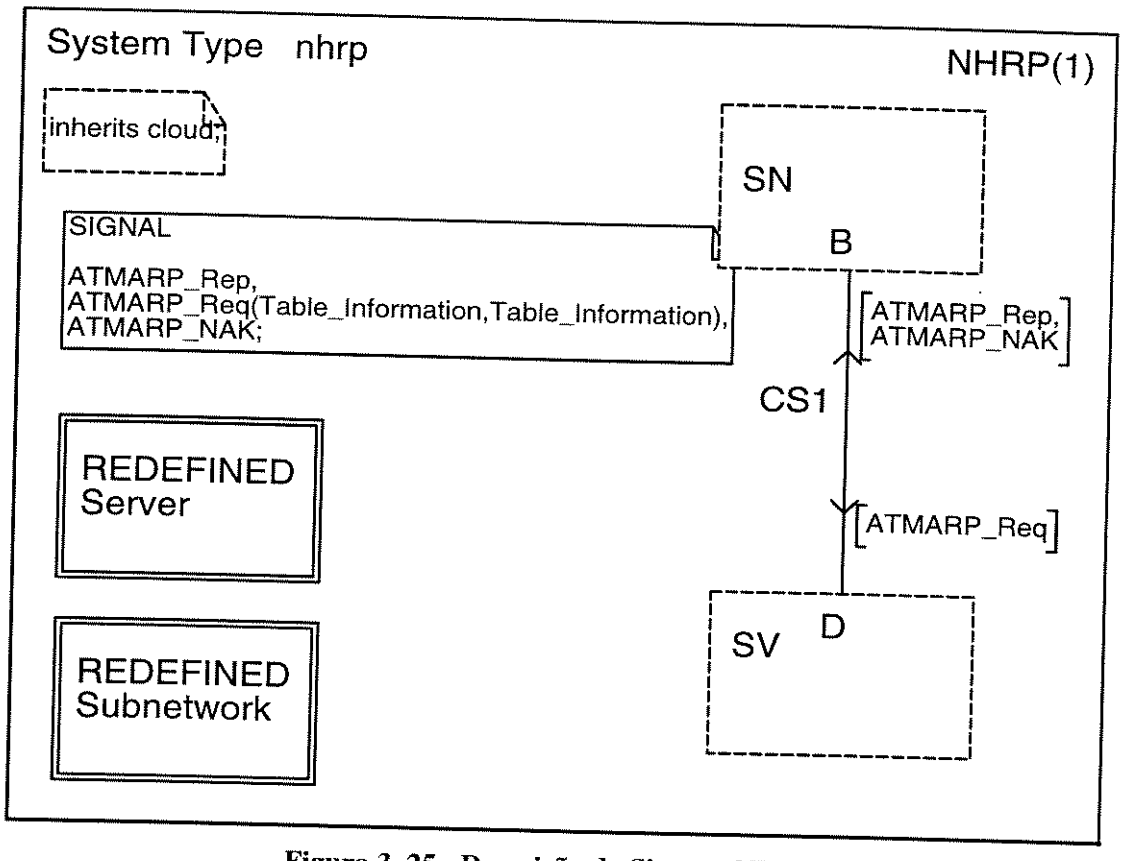

Figura 3. 25 - Descrição do Sistema NHRP

Nos Sistemas IP Clássico e NHRP, as alterações são realizadas principalmente nos Servidores LIS-para o Sistema IP Clássico, Fig. 3.20; e NHS para o NHRP, Fig.3.26; e suas comunicações com as subredes. Isso devido a implantação do reconhecimento do protocolo ARP (Adress Resolution Protocol) nos Sistemas IP Clássico e NHRP. Os blocos *Server* e *Subnetwork* do Sistema NHRP, estão apresentados nas Fig. 3.26 e 3.27 respectivamente.

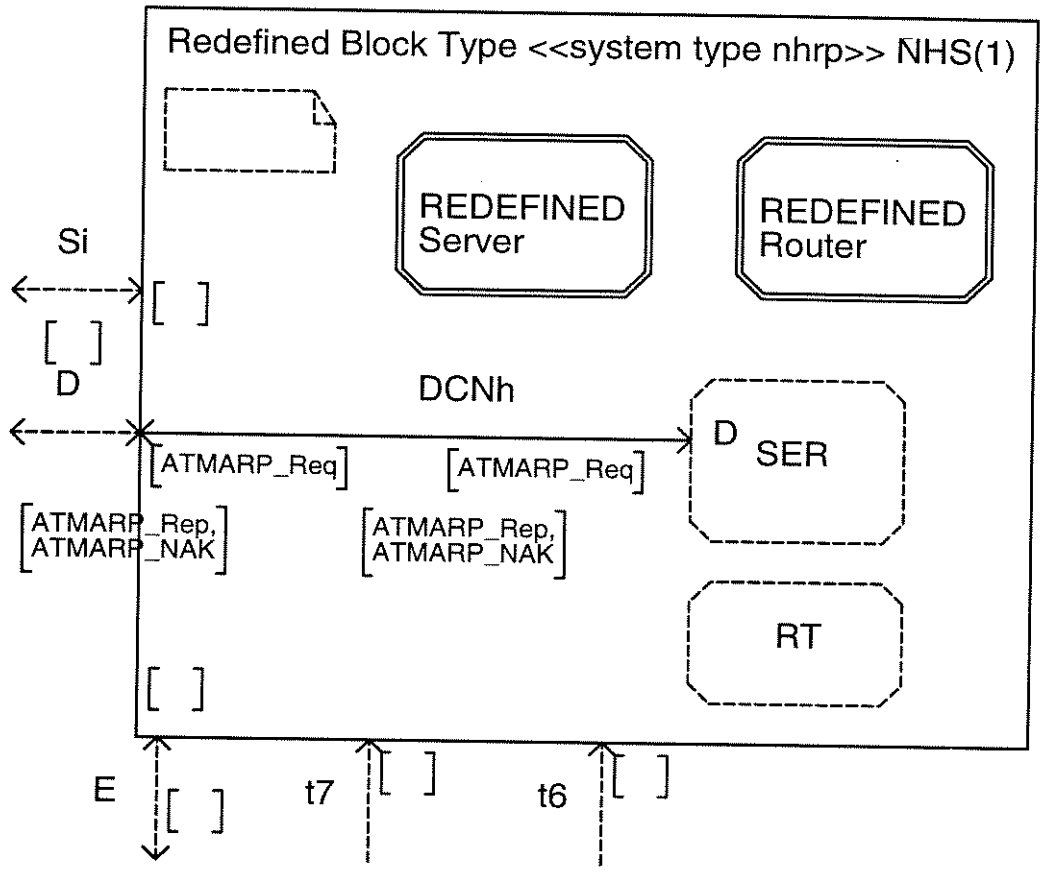

Figura 3. 26 - Bloco Servidor NHS do Sistema NHRP

O *Block Type Server* (Fig. 3.12) do Sistema Cloud, foi redefinido para que pudesse ser especificado com as características de um servidor NHS (NHRP Server). O bloco Server herdado é apresentado na Fig. 3.26. Nesse novo bloco foram alterados os processos Servidor e Roteador, sendo que nenhum sinal foi adicionado ao processo roteador , e foram adicionados novos sinais (Ex.: ATMARP_Req, ATMARP_Rep e ATMARP_NAK) ao processo servidor.

O *Block Type Subnetwork* (Fig. 3.27) apresenta a redefinição do *Process Type Subnetwork,* que foi alterado para realizar o funcionamento de uma NBMA (Non Broadcast Multi Access). Nesse bloco redefinido são adicionados novos sinais ao sistema, assim como os processos redefinidos. Nos blocos apresentados, nenhum processo novo foi acrescentado, pois não apresenta nenhum processo sem que seja tracejado como na super-classe (Sistema *Cloud*).

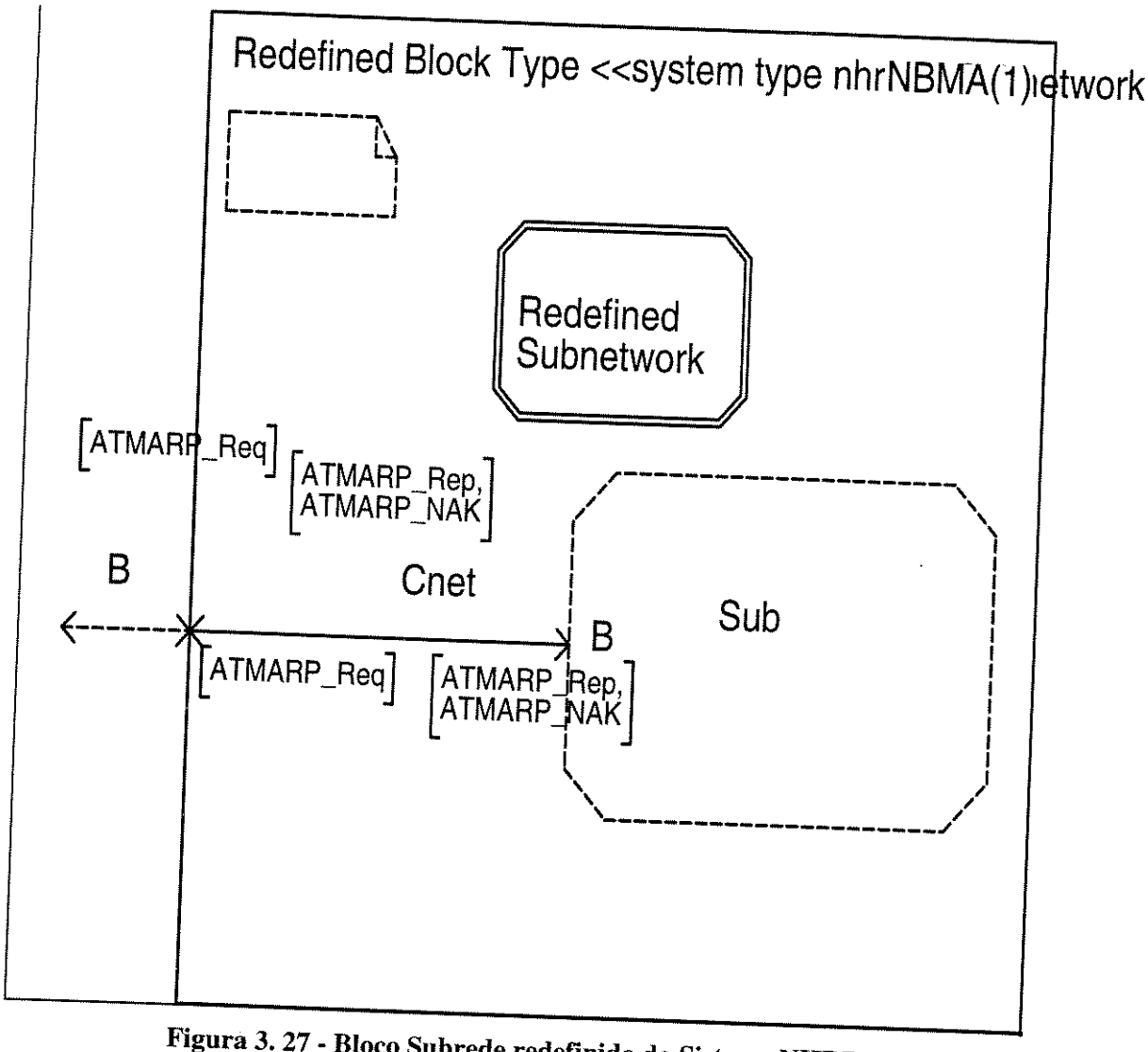

Figura 3. 27 - Bloco Subrede redefinido do Sistema NHRP

O protocolo NHRP, possui a desvantagem de possuir uma topologia pré-definida para realizar o roteamento, como descrito no RFC1755 [31], apesar disso, essa característica traz a vantagem de evitar os *loops* de roteamento dentro da rede. Seu roteamento, foi implementado primeiramente pelo protocolo *Hop-by-Hop*, redirecionando os endereços dos *hosts* para realizar a conexão direta entre eles. Para realizar a conexão externa a nuvem um novo sistema é proposto na próxima seção.

3.3.4 - Sistema NHRP que se comunica através do protocolo Proxy-ARP

O protocolo Proxy-ARP [29] é implementado junto ao protocolo NHRP para realizar o roteamento entre entidades externas a nuvem. Assim, esse protocolo foi especificado utilizando o protocolo NHRP especificado, como mostra a Fig. 3.28, que descreve a hierarquia dos sistemas utilizados. Essa hierarquia mostra a ligação de herança entre o sistema que será apresentado nessa seção, o Sistema NHRP apresentado na seção anterior e o Sistema Cloud apresentado na seção 3.3.1

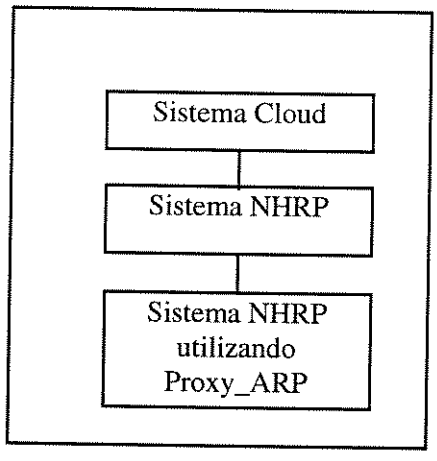

Figura 3. 28 - Hierarquia do Sistema NHRP que se comunica através do protocolo Proxy-ARP

O protocolo Proxy-ARP, foi criado para resolver o problema de resolução de endereçamento, em ambiente sem o mecanismo de *broadcast*. A Fig. 3.29, apresenta o funcionamento do protocolo Proxy-ARP, em que se apresentam dois roteadores (Router1 e Router 5), que realizam a troca de mensagens externas a nuvem. Para que um *host* (Ha, Hb, Hc e Hd) tente enviar uma mensagem fora da nuvem, o sinal deve percorrer a nuvem até encontrar um dos dois roteadores. Esse protocolo, pode ser bem empregado no protocolo NHRP, por apresentar restrições quanto sua topologia, definindo uma rede NBMA. Contudo as desvantagens de restrição da topologia e do percorrimento desnecessário de *hops* ainda permanecem, mas inibem a ocorrência de *loops* dentro da nuvem.

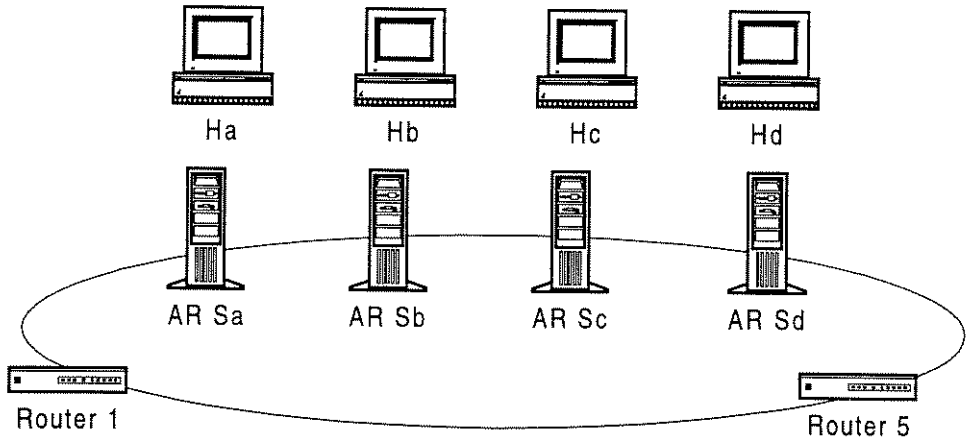

Figura 3. 29 - Meio compartilhado Proxy-ARP de uma única nuvem

O protocolo Proxy-ARP, realiza a conexão direta de uma NBMA (Non-Broadcast Multi-Access) no protocolo NHRP. Para realizar a troca de mensagem entre duas nuvens ATM, os blocos foram instanciado realizando assim a hierarquia não a nível de sistema, e sim a nível de blocos. A especificação desse sistema pode ser vista na Fig. 3.30.

A especificação desse novo sistema apresenta o funcionamento do estabelecimento da conexão entre dois sistemas NHRP com uma característica de roteamento referente ao Proxy-ARP para a realização do roteamento. Na Fig. 3.30 cada sistema NHRP é apresentado também com as caraterísticas do sistema Cloud que diferentemente da Fig. 3.25 a herança aqui é à nível de blocos, e não a nível de sistema, como foi o caso dos sistemas anteriores. O Sistema NHRP controla os *loops* de roteamento interno através da definição da topologia da rede. Tendo a topologia definida, os sinais que se destinam a sistemas fora de seu domínio devem passar pelos Servidores até que alcancem os roteadores que são encarregados de encaminhar as mensagens para fora de sua nuvem. Apesar desse protocolo possibilitar a conexão direta do protocolo NHRP, causa algumas desvantagens como o estabelecimento da topologia rígida e também o percorrimento de *hops* desnecessário até alcançar os roteadores que irão encaminhar as mensagens para fora da nuvem.

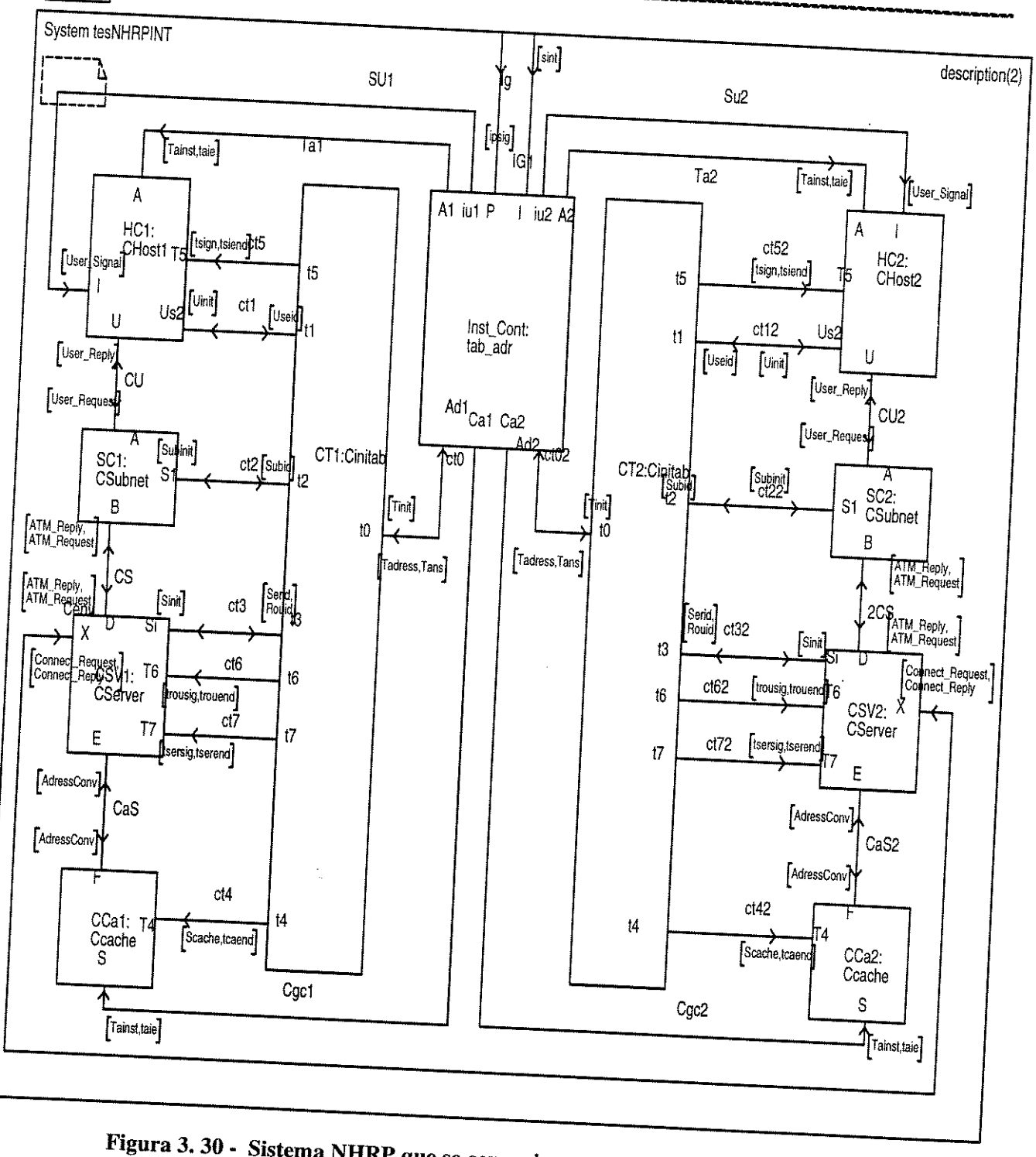

Figura 3. 30 - Sistema NHRP que se comunica através do protocolo Proxy-ARP

A não utilização da herança a nível de sistema para especificar a troca de sinais entre as duas nuvens nesse sistema apresentado nessa seção, é dado por vários fatores. Se a herança fosse feita pelo sistema, o novo sistema herdaria todas as características do sistema antigo, contudo existe a necessidade de se realizar modificações para a implementação do protocolo Proxy-ARP nesse sistema. Nesse caso seria possível utilizar o *Redefined.* O problema encontrado para se simular o Proxy-ARP está na aceitação de endereços que estão fora dessa nuvem, e essa

característica não é aceita pela varsão 3.02 da ferramenta SDT (SDL Design Tool) utilizada, tanto na validação quanto na simulação.

Uma outra solução seria instanciar os blocos, apresentando assim todos os blocos instanciados e os redefinidos. Essa solução apresenta o problema dos blocos instanciados possuírem as características do Sistema NHRP sem as modificações necessárias para que essa nuvem dada pelos blocos instanciados reconheça o funcionamento do protocolo Proxy-ARP.

A solução adotada foi da modularização do sistema, ao invés de criar o *package* com o sistema, se criou um package com os blocos do sistema. Dessa forma os blocos herdam as características dos blocos apresentados no Sistema NHRP, e as redefinições são realizadas uma única vez para a instância do bloco.

O sistema especificado (Fig. 3.30) apresenta a instanciação de blocos (Ex.: Block Type Ccache, instâncias: Cca1,Cca2), pois utiliza mais de um bloco do mesmo tipo. Na especificação existe apenas a especificação do *Block Type*, existindo apenas a especificação de um bloco, mas a instanciação permite a existência de vários blocos modelados uma única vez, diminuindo o tamanho da especificação. Um novo bloco (*Tab_Adr* Fig. 3.31*)* foi acrescentado ao novo sistema, pois todos os outros blocos já existiam nos Sistemas NHRP e Cloud especificados. Esse novo bloco foi criado para gerenciar os endereços das duas nuvens, visto que o bloco *Tab_ini* do Sistema *Cloud* (Fig. 3.8) gerencia apenas os endereços criados dentro da nuvem.

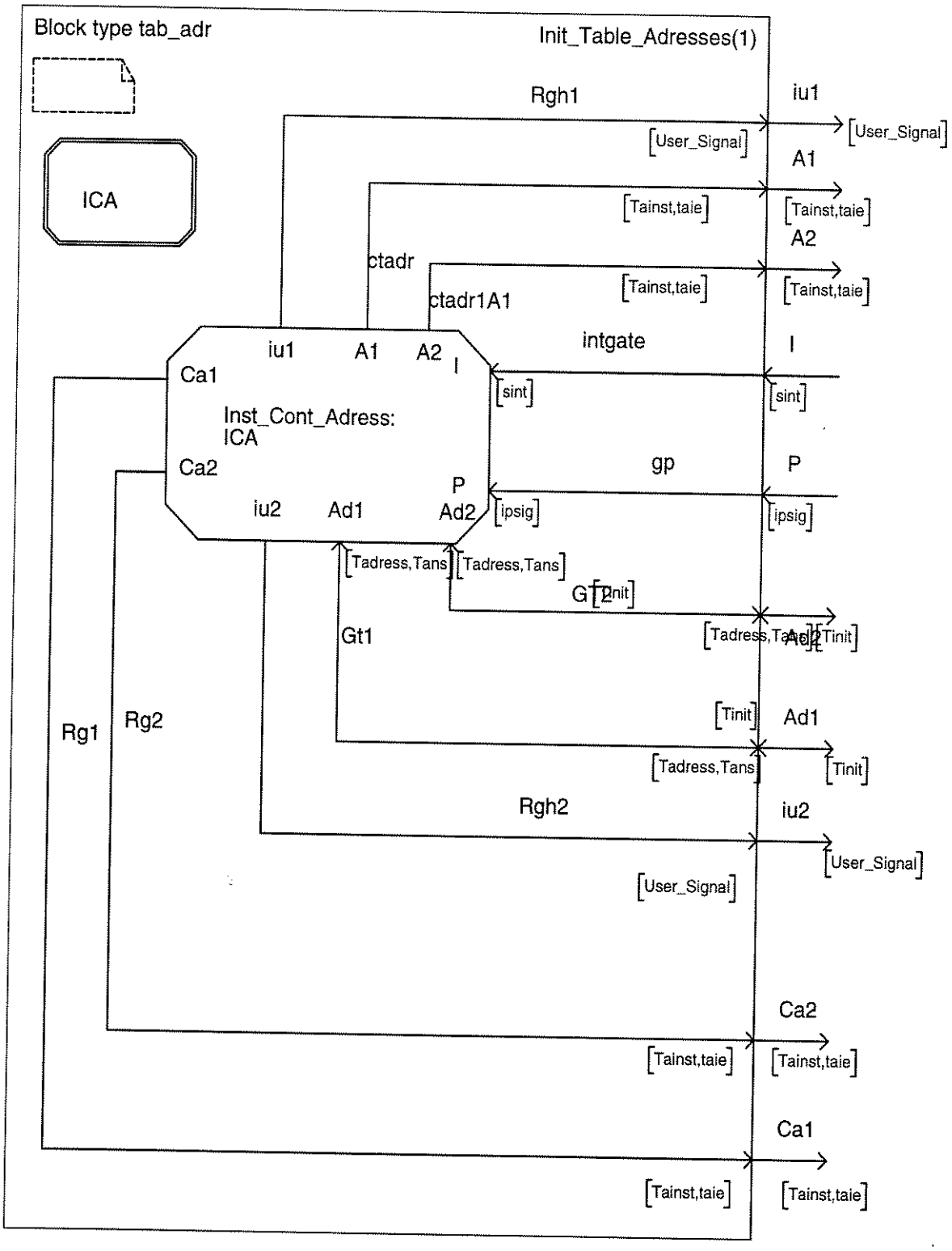

Figura 3. 31 - Bloco tab_adr que gerencia os endereços de duas Nuvens NHRP

O sistema apresentado na Fig. 3.30, exemplifica a conexão de duas nuvens ATM utilizando o protocolo Proxy-ARP para estabelecer a interconexão de duas nuvens ATM através do protocolo NHRP. A modularização dos blocos e processos em *packages*, permite uma maior reusabilidade, utilizando apenas as classes necessárias nos sistemas.

3.3.5 - Sistema IP Clássico Migrado para NHRP

A descrição do protocolo IP Clássico proposto pelo RFC 1577 [28], possui várias limitações, apesar dessa proposta ser simples em sua implementação. Sugestões para eliminar algumas das limitações foram sugeridas no RFC 1932 [32]; em uma delas, está a utilização do protocolo NHRP junto ao IP Clássico. Com o surgimento de novos drafts [19] [20] [21] apresentados pelo IETF, muitas da propostas sugeridas pelo RFC1932, não apresentaram muitas vantagens em seus desenvolvimento.

A partir dos drafts do IP Clássico e NHRP [20] [21], surgiu um novo draft que sugere a migração do protocolo IP Clássico para o protocolo NHRP [22]. Essa possibilidade resulta a implementação do servidor NHS (NHRP Server) junto ao servidor do protocolo IP Clássico, servidor ARP, assim, o protocolo IP Clássico poderia reconhecer o endereçamento IP pelo servidor ARP presente nesse protocolo e também endereçamento do protocolo NHRP, implementado pelo Servidor NHS no protocolo IP Clássico. O diagrama da hierarquia de herança desse sistema pode ser visto na Fig. 3.32, que descreve as características do protocolo IP Clássico, e o servidor NHS herdado do Sistema NHRP.

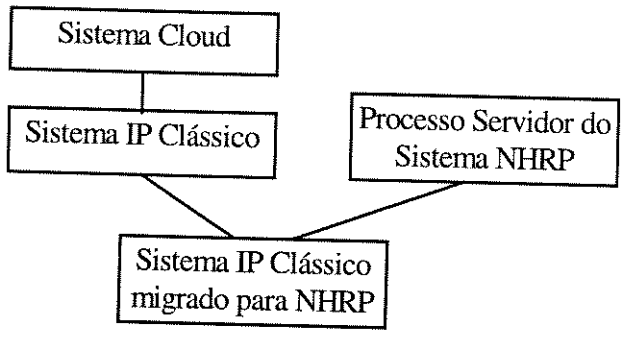

Figura 3. 32 - Hierarquia do Sistema IP Clássico Migrado para o NHRP

O System Type correspondente ao protocolo IP Clássico migrado está apresentado na Fig. 3.33. Esse protocolo apresenta algumas vantagens como a não necessidade de apresentar uma topologia definida como o protocolo NHRP, reconhecer endereços dos protocolos NHRP e IP Clássico, contudo, essa proposta de protocolo ainda não está robusto quanto a ocorrência de *loops*. Outra possibilidade é o de estabelecer uma topologia ao Sistema IP Clássico Migrado para o NHRP como no protocolo NHRP para evitar a ocorrência de *loops*.

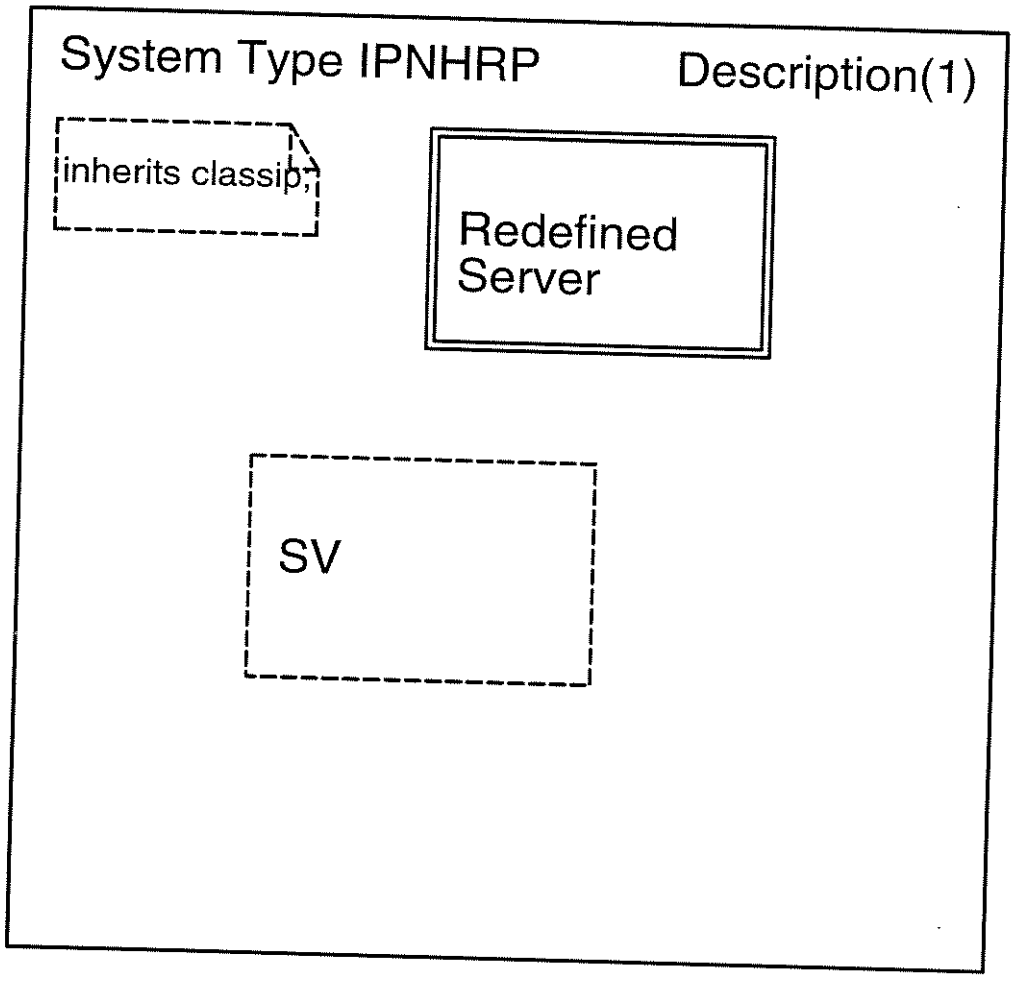

Figura 3. 33 - Sistema IP migrado para NHRP

No sistema migrado para o protocolo NHRP, permanecem as características do Sistema IP Clássico; assim, foi realizado as modificações apenas no *Block Type Server* (Fig. 3.34). A hierarquia do novo sistema reconhece as características do Sistema Cloud, as modificações do Sistema IP Clássico e as novas alterações do Sistema IP Clássico Migrado para o NHRP.

No *Block Type Server* (Fig. 3.34), foi alterado o *Process Type Server* e *Router* do Sistema IP Clássico e acrescentado dois novos processos, que representam o processo do servidor NHS (Processo NHS) do protocolo NHRP e um processo (Processo MANAG) para o reconhecimento do tipo de sinal que está sendo recebido, se é para o servidor ARP ou servidor NHS.

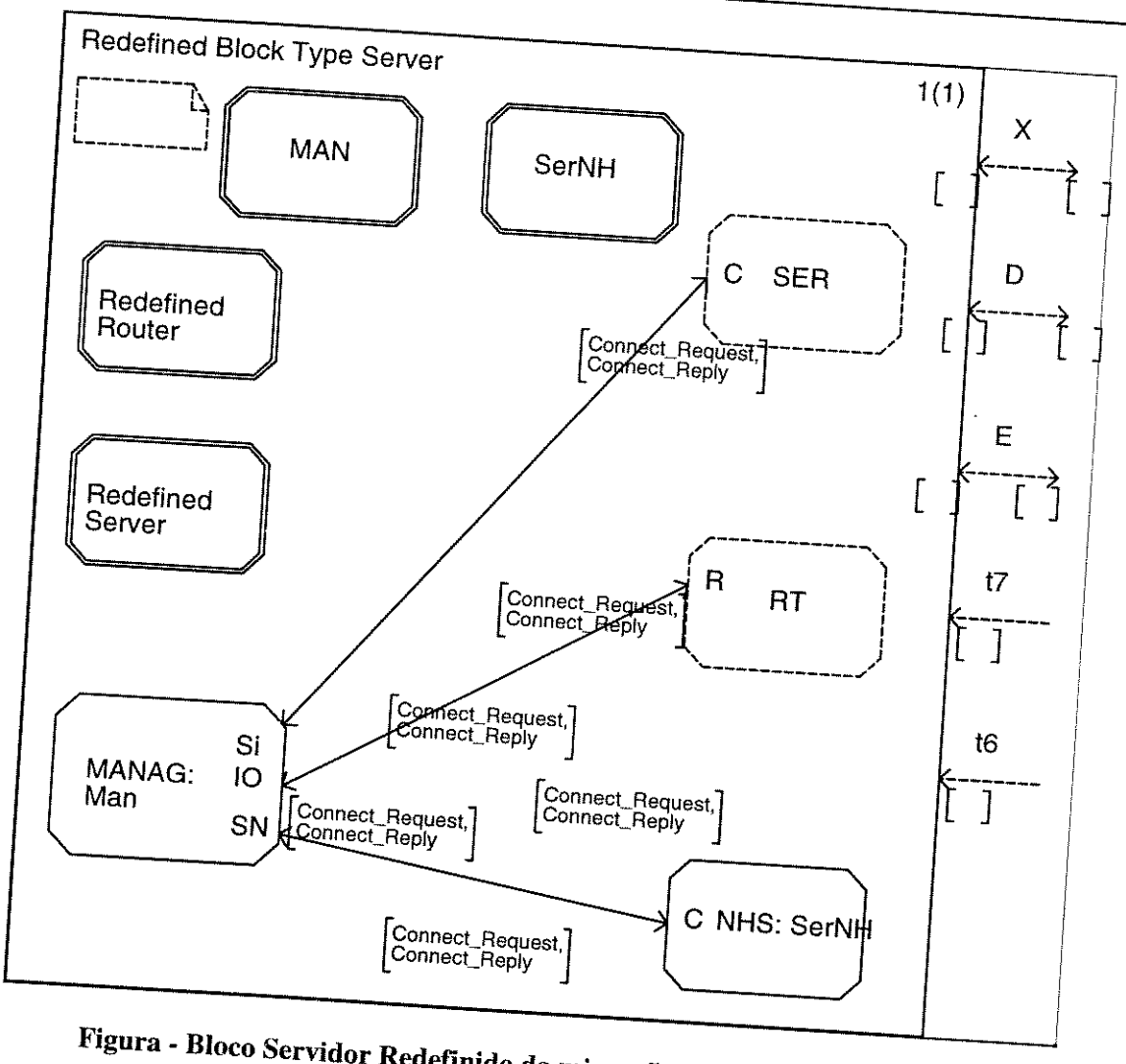

Figura - Bloco Servidor Redefinido da migração IP Clássico para NHRP

O funcionamento do protocolo IP Clássico migrado, não pode ser totalmente visualizado pois não apresenta uma comunicação com um protocolo NHRP, tal que suas características pudessem ser simuladas com a comunicação de uma outra nuvem ATM, para isso, foi apresentado o Sistema IP Clássico Migrado para o NHRP ao lado do funcionamento do Sistema NHRP, para verificar a troca de sinais entre essas duas nuvens. Essa especificação está apresentada no próximo item 3.3.6.

3.3.6 - Sistema IP Clássico Migrado e o NHRP

O Sistema IP Clássico migrado, é o sistema IP Clássico adicionado de um servidor NHS para o reconhecimento de endereços no formato IP e que possa realizar tanto o roteamento IP quanto o utilizado pelo pelo NHRP, contudo para analisar essas características é necessário que

esse sistema possua uma interface para verificação da comunicação entre esses dois sistemas diferentes. Para analisar essa característica, foi criado o Sistema IP Clássico Migrado para o NHRP que faz a interface com um Sistema NHRP, para verificar na execução desse sistema o reconhecimento de endereços e o roteamento realizado. A figura desse protocolo especificado pode ser visto na Fig. 3.35.

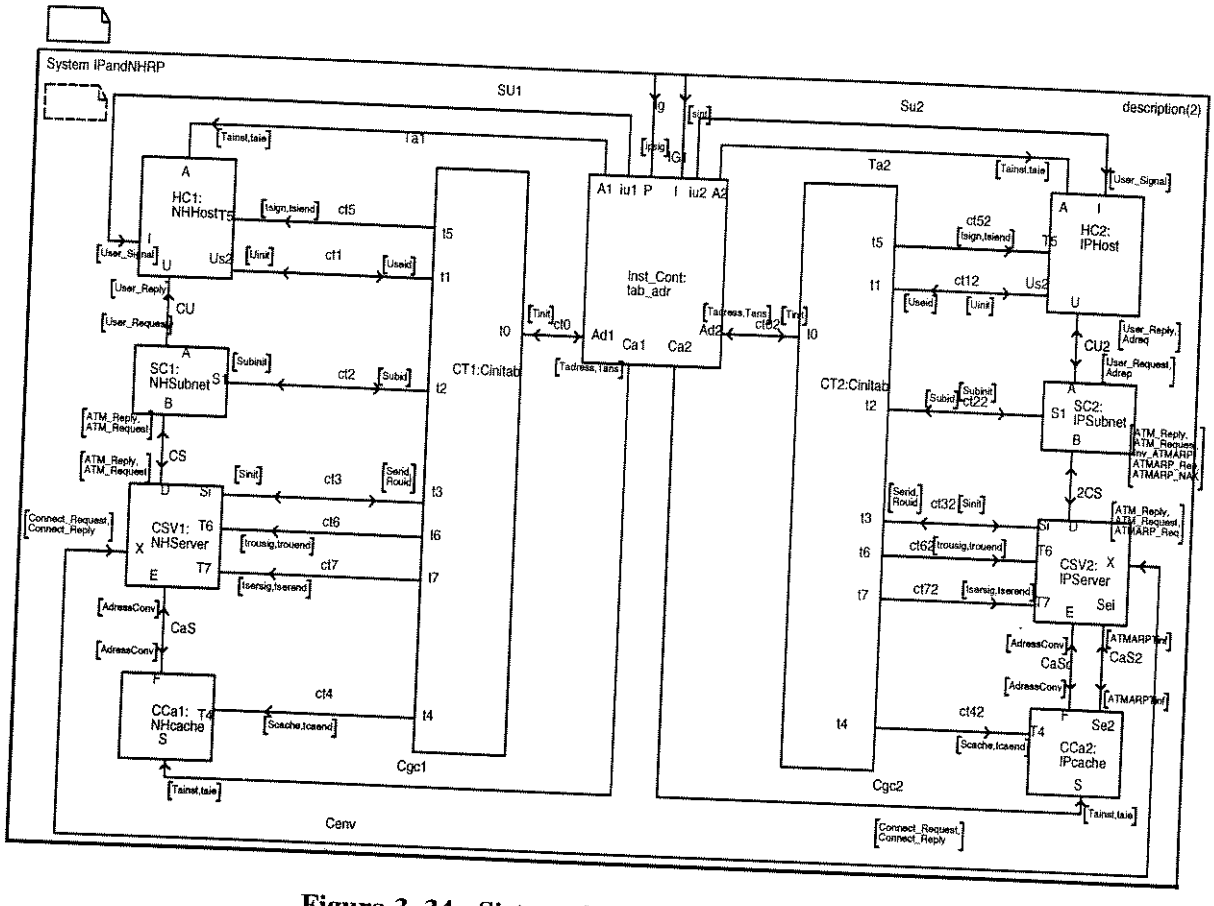

Figura 3. 34 - Sistema IP migrado para NHRP

A especificação desse sistema (Fig. 3.35) foi realizada semelhantemente ao sistema que mostra a comunicação entre duas nuvens NHRP. Os mesmos blocos apresentados no sistema da Fig. 3.30, estão presentes nesse novo sistema, sendo que não existe a instanciação dos blocos já que uma parte se refere ao Sistema IP Clássico migrado e o Sistema NHRP que se comunica através do protocolo Proxy-ARP.

O diagrama hierárquico apresenta o nível de herança desse sistema está mostrado na figura 3.36.

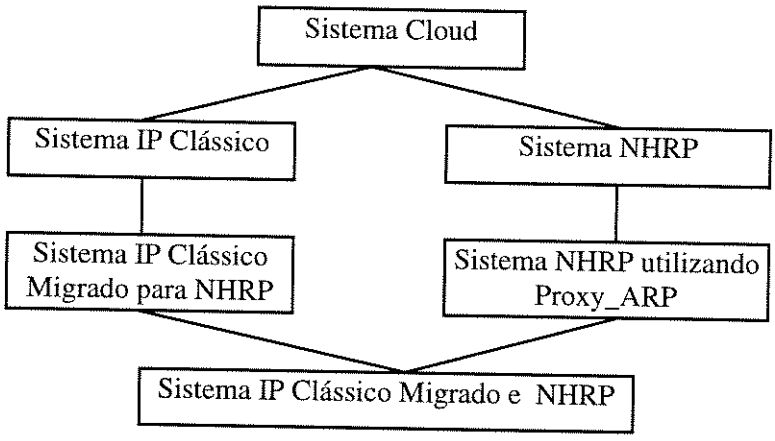

Figura 3. 35 - Hierarquia do sistema que realiza a comunicação entre IP Clássico migrado e o Sistema NHRP

A Fig.3.36 indica que o novo sistema especificado utiliza as classes definidas no Sistema IP Clássico migrado e o Sistema NHRP que se comunica através do protocolo *Proxy-ARP* para se conectar a outra nuvem ATM. Esse sistema utilizou a herança do protocolo IP Clássico, pois a migração é proposta nesse protocolo, e foi estabelecida a conexão com o protocolo NHRP para verificar se esse sistema realizaria a comunicação com o Sistema NHRP.

3.4 - Comentário sobre as Especificações

Durante a realização das especificações pôde-se perceber, que a maior dificuldade, e o maior tempo gasto durante o desenvolvimento das especificações foram empregados durante o estudo dos protocolos para a construção das especificações, e também da super-classe (Sistema *Cloud*) utilizada neste trabalho.

A partir da construção do Sistema *Cloud* se verificou que a dos outros sistemas, requereu menos esforço para realizar as novas especificações. Além disso, com o uso da validação, foi percebido que os erros encontrados eram apenas nas classes novas modificadas, garantindo que as classes antigas herdadas que já haviam passado nas validações e simulações permanecem sem erros, pois já haviam sido encontrados e corrigidos. Dessa forma, o acompanhamento do desenvolvimento em etapas, acrescentando novas características, faz com que se modele sistemas com um maior nível de abstração, validando, simulando e corrigindo os erros, assim, ao passo que esse sistema vai se tornando mais complexo, fica mais fácil de se descobrir e corrigir os erros apontados pelas validações e simulações.

Capítulo 4

Resultados das Validações e Simulações

4.1 - Introdução

A validação é um recurso utilizado para encontrar possíveis erros não percebidos durante a especificação. É uma ferramenta que torna o sistema especificado mais confiável, percorrendo todos os caminhos possíveis e gerando combinações de valores para analisar os procedimentos de uma especificação. A simulação estuda casos especiais na especificação. A simulação é adotada para acompanhar o funcionamento em casos específicos no estudo do comportamento dos sistemas.

A validação e simulação dos sistemas desse trabalho foram realizados utilizando o *Validator* e o *Simulator* monitorados pelo *MSC Editor* do package SDT , que auxilia o encontro de erros de especificação visualizando as instâncias dos processos para a deteção das falhas de especificação. Durante a simulação é possível acompanhar a execução da simulação passo a passo, utilizando tanto o *MSC Editor* como o *SDL Editor*, podendo analisar a execução detalhada do sistema.

A simulação é utilizada para visualizar determinada ação que o desenvolvedor deseje verificar detalhadamente. A validação também pode ser realizada passo a passo como na simulação utilizando o MSC (Message Sequence Chart) para acompanhar o funcionamento do sistema. Uma importante característica utilizada na simulação é a possibilidade de ser realizada parcialmente. Por exemplo, é possível acompanhar a simulação apenas de um dos blocos ou um dos processos, facilitando o encontro de erros localizados.

Uma outra diferença encontrada entre a validação e a simulação é a utilização do *Clock (variável TIME em SDL)* e do *Temporizador* (Variável do tipo TIMER em SDL). O *Clock* pode ser considerado tempo de duração, que pega o valor do tempo naquele determinado momento, um relógio interno, que só é utilizado dentro simulação. O *Temporizador*, a partir do momento que for definido, será sempre incrementado, esse tipo pode ser utilizado tanto na validação como na simulação. Nas especificações apresentadas neste trabalho de tese foram utilizados apenas o

Clock para a detecção de *Time out*. Para determinar a ocorrência de *Time out*, é utilizado duas variáveis de *Clock* que verifica o máximo de tempo que esse sinal pode trafegar na rede. A determinação do tempo em que o sinal está trafegando, é calculado fazendo a diferença do tempo local e o tempo em que o sinal saiu de seu domínio, a partir desse valor se verifica se esse sinal está perdido ou não, detectando assim o *Time out*.

4.2 - Validação

A ferramenta *Validator* utilizada pelo SDT apresenta três algoritmos para eventuais validações: *Random Walk, Exaustive Exploration* e o *Bit State Exploration.* O *Random Walk* é um algoritmo simples de percorrimento que tem como resultados: o número de estados gerados (Generated States), o número de mensagens (Number of reports), que podem ser referentes a erros como perda de sinais, ou apenas avisos como vários destinos para um sinal (principalmente quando se envia o sinal para fora do sistema). Esse algoritmo de exploração apresenta também o número mínimo (Min Depth) e o máximo de profundidade (Max Depth) dos ramos percorridos, e a percentagem de símbolos percorridos (Symbol Coverage).

O algoritmo *Exaustive Exploration* apresenta a lista de resultados análoga ao do *Random Walk,* sendo que algumas vezes os resultados são diferentes pois esse algoritmo não possui as mesmas prioridades admitidas pelo *Random Walk,* que dá maior prioridade no tratamento aos sinais de dentro do sistema, enquanto que no *Exaustive Exploration* todas as prioridades são as mesmas, tanto para os sinais que chegam fora do ambiente quanto para os sinais internos.

O algoritmo *Bit State Exploration* possui a característica de ser mais complexo, o resultado desse algoritmo resulta nos dados estatísticos percorridos do sistema especificado. Se executado após o *Random Walk,* esse algoritmo retorna os dados estatísticos referente ao percorrimento do algoritmo *Random Walk.*

Uma característica apresentada pelos algoritmos de validação encontrada, é a inesgotável execução da validação, apesar de já terem sido utilizados os mesmos valores que os sinais possam assumir. Esse fato ocorre em sistemas em que os processos não são finalizados, ou seja, esses processos sempre existirão no sistema, forçando a execução contínua da validação, e sempre estarão gerando sinais a esses processos. Esse foi o caso dos sistemas especificados, pois a

finalização de um único processo significaria a perda da configuração da rede que foi especificada, impossibilitando a troca de sinais e mensagens entre as entidades do sistema.

Entre os algoritmos de validação apresentados, para se construir uma análise mais detalhada, o Bit State Exploration é o que fornece maiores detalhes, enquanto que o Random Walk e o Exaustive Exploration, apresentam resultados superficiais, melhores apenas, por serem mais simples e mais rápidos, e mais utilizados nos primeiros passos das validações no encontro de erros de especificação. A seguir serão mostrados os resultados obtidos para os sistemas durante a validação, e discutidos os resultados mais interessantes.

```
            Sistema Cloud
** Random Walk**
Number of reports: 0
Generated States: 10.100
Max Depth: 300
Min Depth: 110
Symbol Coverage: 93.72

** Bit State Exploration **
Number of reports: 0
Generated States: 101
Truncated path: 0
Unique System States:101
Size of hash table: 8.000.000
(1.000.000 bytes)
Number of bits set in hash table:202
Colision risk:0%
Max depth:100
Current depth:-1
Min State size:2304
Max state size: 4892
Symbol Coverage: 93.72
```

Figura 4. 2 - Resultados da Validação (Random Walk e Bit State Exploration) do Sistema Cloud

```
        Sistema IP Clássico


** Bit State Exploration **
Number of reports: 0
Generated States: 360
Truncated path: 0
Unique System States:360
Colision risk:0%
Max depth:100
Current depth:-1
Min State size:720
Max state size: 4976
Symbol Coverage: 84.79
```

Figura 4. 1 - Resultado do Bit State Exploration do Sistema IP Clássico

```
Sistema NHRP

** Bit State Exploration **
Number of reports: 0
Generated States: 594
Truncated path: 0
Unique System States:584
Colision risk:0%
Max depth:249
Current depth:-1
Min State size:572
Max state size: 4252
Symbol Coverage: 81.78
```

Figura 4. 3 - Resultado do Bit State Exploration do Sistema NHRP

```
Sistema NHRP que se comunica
através do protocolo Proxy-ARP

** Bit State Exploration **
Number of reports: 0
Generated States: 901
Truncated path: 0
Unique System States:901
Colision risk:0%
Max depth:500
Current depth:-1
Min State size:916
Max state size: 7960
Symbol Coverage: 88.70
```

Figura 4. 4 - Resultado apresentado pelo Sistema NHRP que se comunica através do protocoloque se Proxy-ARP

```
Sistema IP Clássico Migrado para
NHRP

** Bit State Exploration **
Number of reports: 0
Generated States: 6.000
Truncated path: 0
Unique System States:5979
Colision risk:0%
Max depth:500
Current depth:468
Min State size:592
Max state size: 4.164
Symbol Coverage: 78.38
```

Figura 4. 5 - Resultado do Sistema IP Clássico Migrado para o NHRP

```
Sistema IP Clássico Migrado e NHRP

** Bit State Exploration **
Number of reports: 0
Generated States: 2.000
Truncated path: 0
Unique System States:1999
Colision risk:0%
Max depth:500
Current depth:481
Min State size:1040
Max state size: 8900
Symbol Coverage: 89.43
```

Figura 4. 6 - Resultados do Bit State Exploration do Sistema IP Clássico Migrado e o NHRP

O Sistema Cloud, é a super-classe especificada para que todos os outros sistemas especificados herdassem suas características. A Fig. 4.1 exemplifica os resultados obtidos com a validação desse sistema, que apresenta detalhes dos resultados dos algoritmos *Random Walk* e *Bit State Exploration*. Interpretando os resultados da validação do sistema Cloud, o número de mensagens (*Number of Reports*) igual a 0 significa que nenhuma situação de erro ou aviso foi encontrada; nenhum caminho foi truncado (*Truncated Path*), o número de estados únicos do sistema (*Unique System States*), se refere ao número real de 101 estados do sistema, o número de

estados gerados (*Generated States)* compreende-se que a validação percorreu todos os estados existentes do sistema. O risco de colisão (*Colision Risk*) do sistema foi de 0%. O número de estados únicos é o número referente ao números de estados declarados durante a especificação. A existência de um número maior de estados gerados ocorre quando existe fila de sinais em um estado pois um estado pode ler mais de um sinal.

O sinal -1 de profundidade de busca (*Current* Depth) dos ramos da árvore gerada, significa que a validação foi completada. Contudo, a validação foi forçada a terminar, pois os algoritmos de exploração não terminam pois sempre estará entrando sinal a ser processado até que o processo seja terminado. Como na especificação essa característica não ocorre, os manuais do SDT [56] [58] sugere que se valide por aproximadamente 20 minutos para que o algoritmo percorra a árvore gerada pela validação. Em todos os resultados do *Bit State Exploration* terá valor -1 por ter sido terminado pelo usuário e os resultados obtidos por esse algoritmo serem baseados no *Random Walk* realizado. O *Hash Table* não foram mencionados nos outros resultados, por apresentarem o mesmo tamanho da tabela, e não ser interessante para o estudo desse trabalho.

De acordo com a Fig. 4.1 foram cobertos 93,72% dos símbolos (*Symbol Coverage*). Isso ocorreu devido ao fato de não terem sido alcançados os símbolos de tratamento de erros, como os de perdas de sinais. Esses símbolos puderam ser alcançados durante a realização de simulações, através do envio de sinais com conteúdos de erros para analisar essas situações, e foi constatado que o sistema tratava essas situações completamente.

Outro motivo do percorrimento de 93,72% dos símbolos foi observado pelo *Coverage View* (Visualização da árvore de percorrimento), que apresentam os diagramas dos símbolos do sistema e mostra quais os símbolos que foram e os que não foram alcançados. Os símbolos em branco apresenta os símbolos que não foram alcançados como mostra a Fig. 4.7 que descreve quais as instâncias dos blocos não foram utilizadas.

Os resultados das validações apresentados nesse capítulo, são os resultados finais após terem sido corrigidos os erros apontados pelos algoritmos utilizados. Um dado interessante apresentado pelo *Bit State Exploration* está nos números de estados gerados e números de estados únicos do sistema. O número de estados únicos sempre é menor, ou igual ao número de estados gerados.

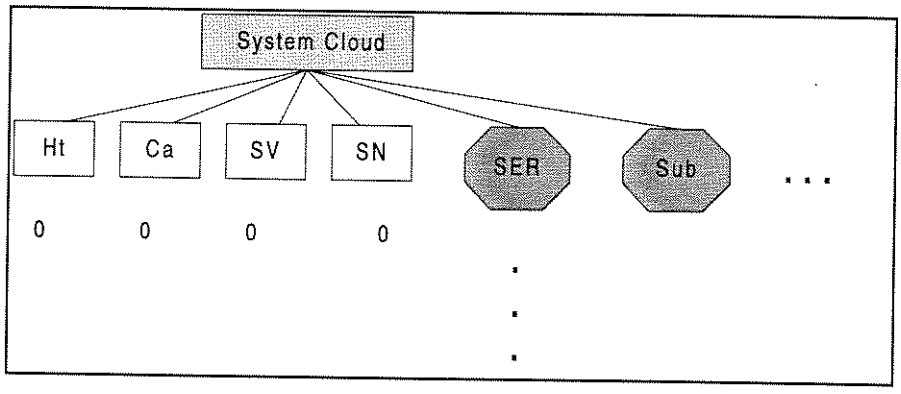

Figura 4. 7 - Parte do Coverage View do Sistema Cloud

O *Coverage View* mostra que as instâncias dos blocos sempre são colocadas como não utilizados, evitando que os sistemas cheguem a 100% de área percorrida, isso porque o sistema percorre o *Block Type* e não a instância da qual ela faz parte desse tipo. Quando o sistema herda de outros sistemas, seus blocos herdados também não constam como área percorrida, diminuindo assim sua percentagem de percorrimento.

Para percorrer todos os caminhos dentro da validação, pode ser utilizado o comando *ANY* para criar valores aleatórios, entretanto este recurso não foi utilizado nas especificações, pois não é possível criar *Pids* aleatórios, assim se manteve a especificação original e se utilizou a simulação para verificar esses símbolos não percorridos durante a validação.

A especificação do Sistema *IP Clássico* herda características do Sistema *Cloud*, assim apresenta no *Coverage View* uma instância a mais (System CLIP, que representa o nome físico do Sistema IP Clássico) que não está sendo utilizada diminuindo ainda mais o percentual de símbolos percorridos pelo sistema. Outra característica que explica a percentagem de símbolos percorridos é que se mantém os símbolos redefinidos da super-classe, mesmo que esses símbolos não sejam mais existentes no novo sistema, diminuindo nas classes herdadas o percentual dos símbolos percorridos. Esse fato pode ser visto comparando os resultados das validações dos sistemas (Fig. 4.1 à 4.6) e comparando o *Coverage View* da Fig. 4.7 e Fig. 4.8.

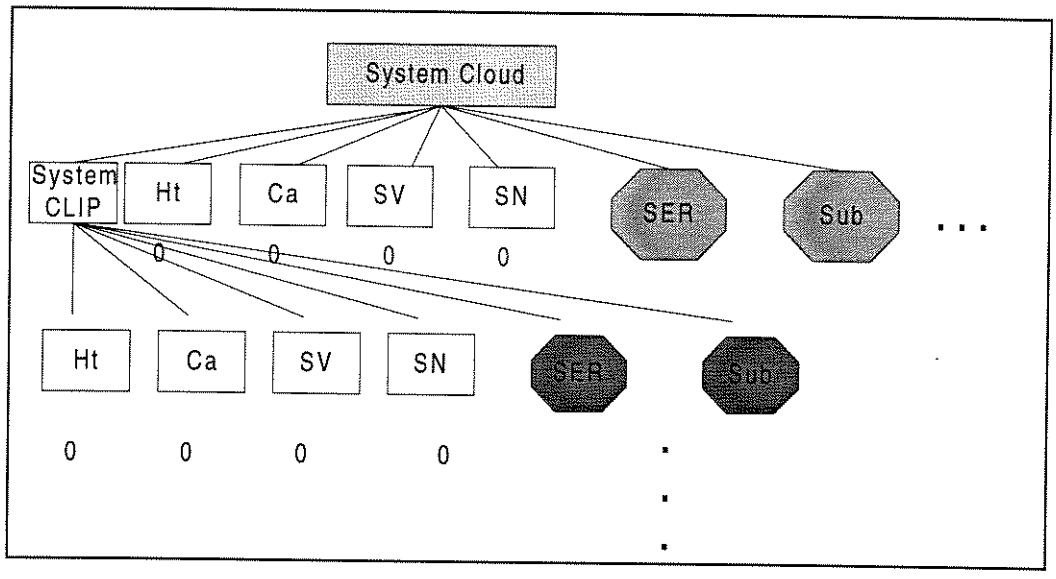

Figura 4. 8 - Parte do Coverage View do Sistema IP Clássico

A percentagem de símbolos percorridos (Symbol Coverage) do Sistema NHRP que se comunica através do protocolo *Proxy-ARP* (Fig. 4.4) é maior do que no Sistema NHRP. Isso porque, nesse sistema especificado, a herança não é direta do Sistema NHRP, a herança é dada à nível de blocos.

4.3 - Simulação

Após a realização das validações e identificadas as inconsistências existentes no sistema, são realizadas simulações de casos críticos, como por exemplo, casos em que a validação não cobriu. Através do uso do MSC (Message Sequence Chart) é possível visualizar o funcionamento da especificação e verificar se o sistema funciona de acordo com o funcionamento previsto durante o estudo dos protocolos propostos. A visualização gráfica do funcionamento da especificação do sistema é um recurso importante para identificar falhas realizadas durante a especificação.

A configuração de rede adotada para simulação e validação foi a mesma adotada em todos os sistemas, sendo essa, definida pelo bloco Tab_Ini definido no Sistema *Cloud* (Fig. 3.17) e herdado em todos os outros sistemas. Essa configuração é apresentada na Fig. 4.9. A mesma configuração foi adotada para se realizar uma melhor comparação entre as validações e simulações entre cada sistema.

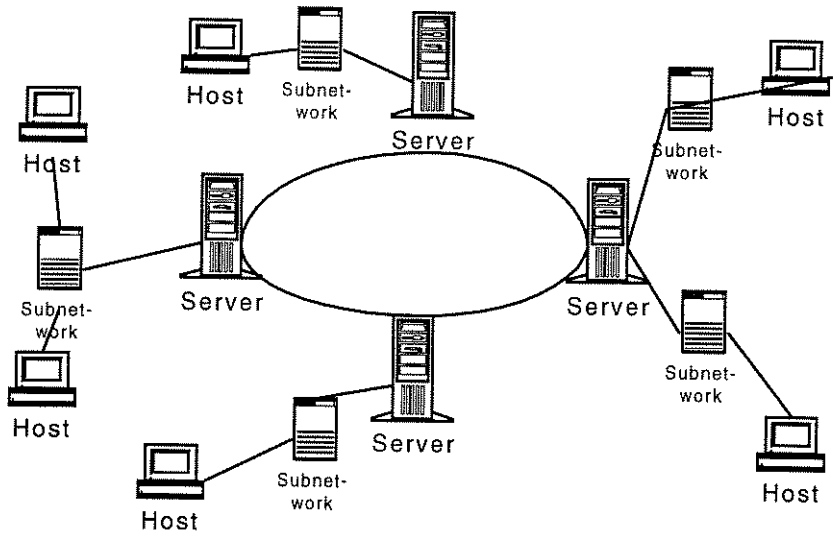

Figura 4. 9 - Configuração de rede utilizada para simulação e validação dos sistemas

A simulação pode ser realizada localmente em um bloco, ou no sistema completo para dar uma visão total do sistema. Em certos casos, era interessante estabelecer a simulação local, para testar casos que não poderiam ser gerados pela validação. Um desses casos foi a verificação de *Time Out* no tratamento de um sinal esperado, ou na verificação de uma rede que não possa ser alcançada. Para realizar a simulação desse último caso, o endereço da rede que não possa ser alcançada, deve ser tirada das tabelas de roteamento, fazendo com que o sinal deva ser transmitido a outro roteador, causando o *Time Out*, pois o sinal permaneceria procurando esse endereço indefinidamente.

Nos sistemas apresentados no capítulo anterior, é possível verificar a chegada de um sinal do ambiente. Esse sinal é muito importante na simulação, pois é a partir dele que se determina os testes que serão realizados. O sinal que chega do ambiente, e que uma instância de *Host* deve receber, determina a fonte e o destino da simulação. Por exemplo, na nuvem ATM apresentada pela Fig. 4.9, existem 6 *Hosts*, se a cada um deles possui um endereço, que no caso do SDL seria o *Pid* de cada instância, sendo cada *Pid* único, havia um endereço a cada entidade existente nessa figura. Quando um sinal chega do ambiente, esse estabelece a origem do *Host* que deseja efetivar a conexão, e o *Host* destino, a quem será feita a conexão, mas para estabelecer essa conexão, existem diversos intermediários, entre servidores, roteadores e gerenciadores de subredes.

Os sistemas tratados nesse trabalho, apresentam o estabelecimento de uma conexão PVC (Permanent Virtual Connections), apresentada no capítulo 2, esse tipo de conexão determina que

deve ser estabelecida a conexão antes da troca de mensagens entre os nós. O formato dos endereços foi mostrado na Fig. 2.3, e a especificação desses sinais em SDL foi apresentada na Fig. 3.3. Essas características são importantes a serem observadas para a interpretação das simulações.

A configuração de rede da Fig. 4.9, foi criada para que se pudesse observar as ações, e as trocas de sinais entre cada processo, como troca de sinais dentro de uma mesma rede, entre uma mesma subrede, fora da rede do servidor e em alguns sistemas, fora da nuvem ATM. Uma característica importante analisada que envolve a linguagem de especificação utilizada, foi quanto a simulação de vários *hosts* tentando estabelecer conexão a outros nós, uma característica que não é verificada quando se trata apenas a conexão de um *host* a outro *host.*

O caso estipulado, é que um servidor pode estar tentando estabelecer conexão a outro nó, enquanto isso, esse servidor, pode estar recebendo uma requisição de conexão e também ser um nó intermediário enquanto aguarda o estabelecimento de sua conexão. Esse caso não é tratado quando se analisa apenas o estabelecimento entre dois nós, mas na tentativa de conexão entre vários nós, que seria uma característica semelhante ao funcionamento real das redes de comunicação.

Com o objetivo de exemplificar as simulações e alguns casos interessantes de serem analisados, serão analisados alguns casos de estabelecimento de conexão, utilizando pedaços de MSC (Message Sequence Chart), para melhor compreender a ação dos processos. O exemplo do caso 1 apresentado na Fig. 4.10. Nesse caso o host1 deseja estabelecer a conexão ao host4, para isso ele passa pelo servidor1 a que pertence, direciona o sinal ao servidor3, que envia ao servidor2 e finalmente chega ao servidor4 a quem o host4 pertence. Esse caso, é apenas um exemplo simples de estabelecimento de conexão.

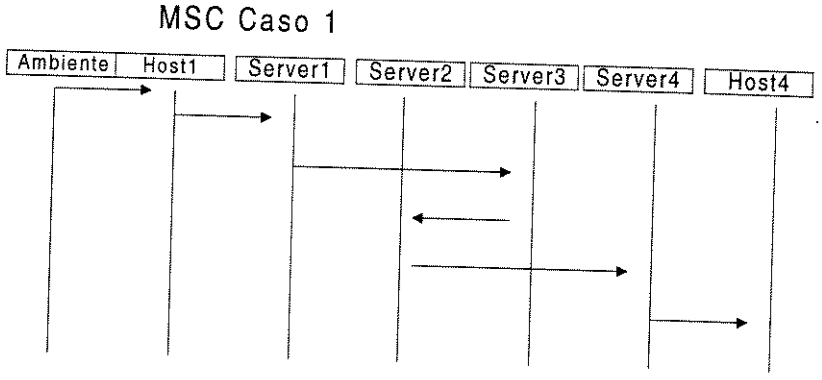

Figura 4. 10 - Caso 1 - estabelecimento de conexão

No caso 2 de simulação, é apresentado um exemplo típico encontrado nas especificações. A ausência de um símbolo de *SAVE* na especificação em SDL pode comprometer toda a especificação, pois como no exemplo da Fig. 4.11, a perda de sinal apresentada pela seta em "* ", significa que esse sinal foi perdido, isso porque seu estado de tratamento é no estado IDLE e não no estado encontrado *taking_adress*. A correção desse problema, é apresentado no MSC na figura 4.12 após a correção da especificação SDL correspondente.

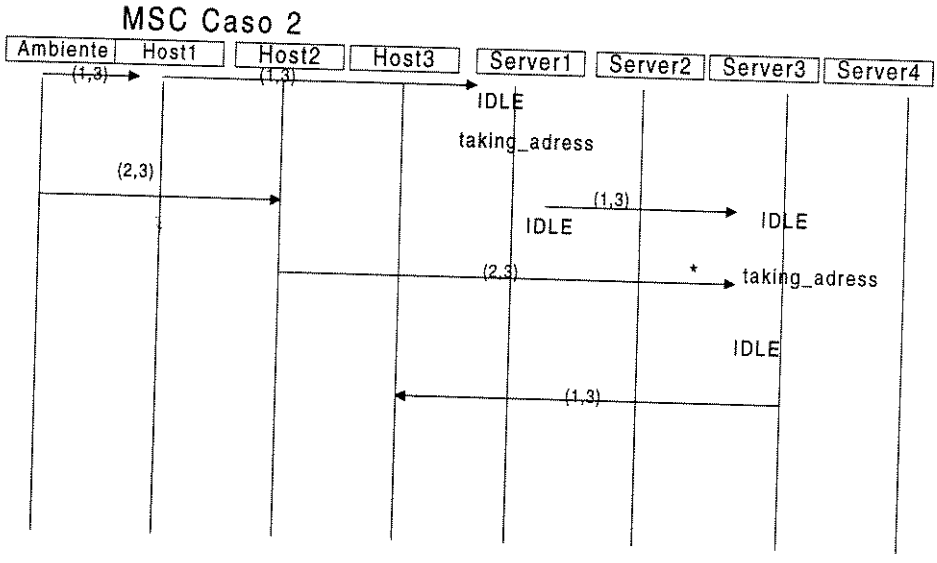

Figura 4. 11 - Caso 2 - perda de sinal

A perda de sinal apresentada no caso 2 é facilmente resolvida na especificação em SDL, mas que por muitas vezes, essa característica pode passar despercebida, pois essa característica só é descoberta a partir do momento em que uma mesma instância de processo possa estar sendo requisitada por mais de uma outra instância, tanto do mesmo tipo de processo, como de outro. A análise de dois casos em paralelo também auxilia no descobrimento de novos erros, o que estaria aproximando uma simulação à realidade.

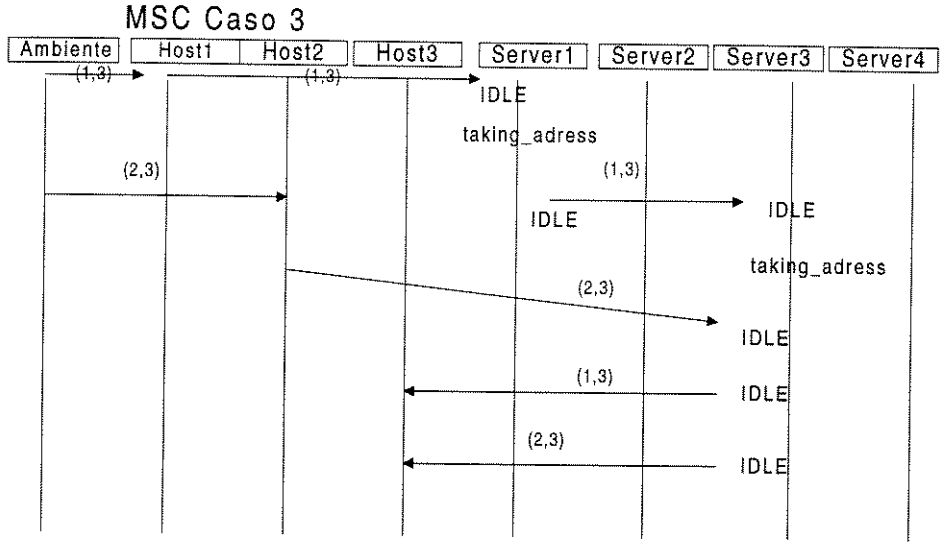

Figura 4. 12 - Caso 3 - Correção de especificação

Os casos de simulação apresentados nesses exemplos, foram casos encontrados no decorrer do desenvolvimento desse trabalho. Os pedaços de MSC mostrados nesse capítulo, são apenas pequenas amostras do funcionamento, não apresentando uma cópia do MSC gerado na validação, pois essa não seria possível ser visualizada totalmente por apresentar diversas instâncias. Nesse caso, foram apresentados apenas uma visão do funcionamento realizado nas simulações desse trabalho. Nos casos apresentados na Fig. 4.10, 4.11 e 4.12, foram mostrados apenas a troca principal de requisição de conexão, não entrando detalhadamente no conteúdo ou na resposta enviada após o nó ter sido encontrado.

Através da realização das simulações, podem ser analisados os protocolos especificados. O protocolo IP Clássico, é o protocolo mais simples de ser implementado, pois não interfere na configuração da rede existente, realizando o roteamento realizado nas redes TCP/IP. Os tipos de roteamento propostos para otimizar os nós percorridos e para tentar estabelecer uma conexão direta, podem otimizar o tempo no estabelecimento da conexão. Contudo, esses mecanismo empregados não inibem a possibilidade de ocorrência de *loops*. Um exemplo do modo de funcionamento do Sistema IP Clássico é mostrado na Fig.4.13. A Fig. 4.14 exemplifica um *loop* detectado após a desativação de um roteador que ligava o Server 3, assim, a única conexão existente, seria a conexão ao roteador do Server 4, que nesse exemplo, entraria em *loop*.

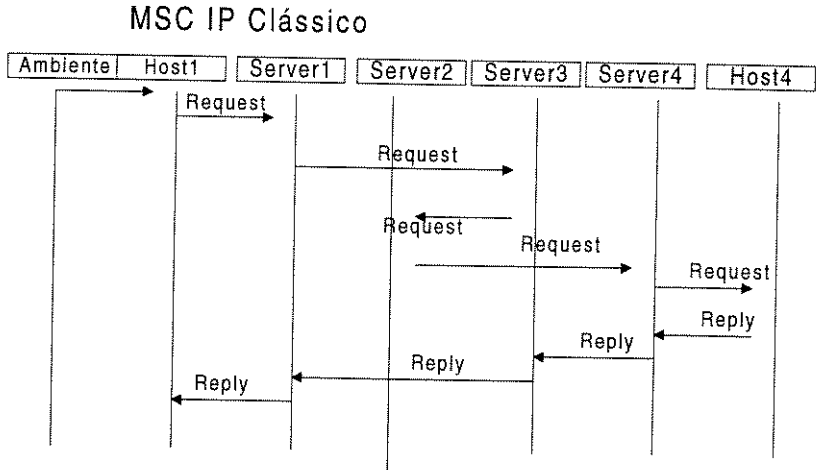

Figura 4. 13 - Exemplo de funcionamento do Sistema IP Clássico

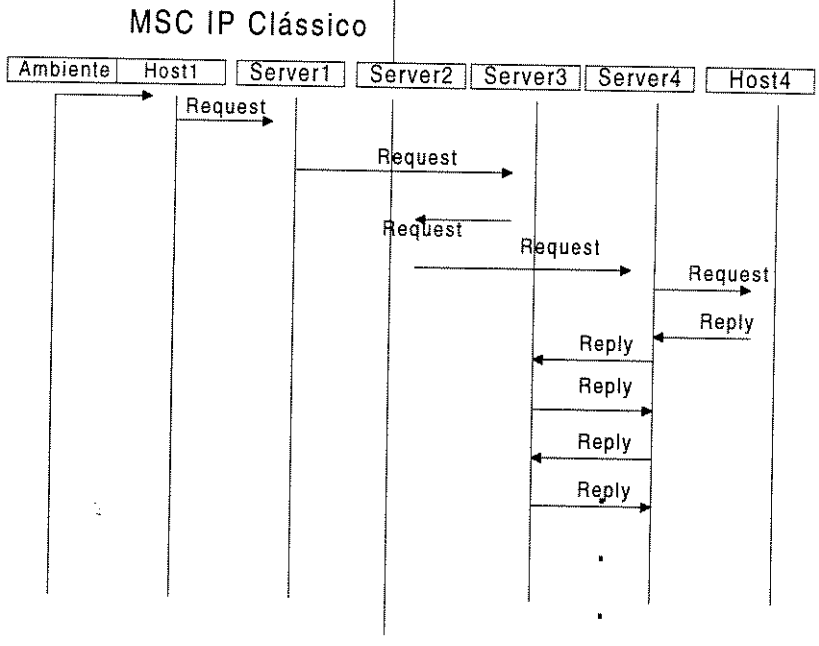

Figura 4. 14 - Loop detectado no funcionamento do Sistema IP Clássico

No protocolo NHRP a existência de *loops* não é encontrada (Fig. 4.15), porém para que isso ocorra, esse protocolo sofre algumas limitações como a configuração da topologia da sua rede. A utilização do *Proxy-ARP* para realizar a troca de mensagens fora de uma nuvem ATM, força a mensagem a percorrer os nós até o encontro do roteador que encaminhará a mensagem para fora da nuvem.

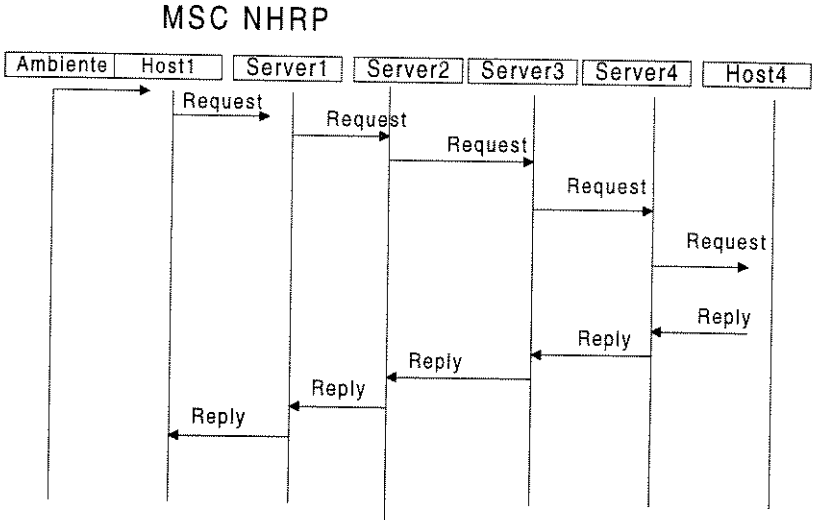

Figura 4. 15 - Exemplo de funcionamento do Sistema NHRP

A migração do protocolo IP Clássico para o protocolo NHRP propõe a implementação do servidor NHS junto ao servidor ARP (Address Resolution Serever) do protocolo IP Clássico. Esse recurso auxilia na resolução de endereços entre os dois protocolos. Assim, realizando o roteamento da nuvem IP, no mesmo modo do protocolo IP Clássico, o problema de *loops* ainda permaneceria, mas para que o roteamento realizado seja como no protocolo NHRP, essa nuvem deve ter também uma topologia estipulada, trazendo essas desvantagens.

4.2 - Conclusão

A validação e a simulação são recursos utilizados para encontrar possíveis erros de especificação e para melhor visualizar o funcionamento das especificações utilizando recursos gráficos disponíveis como MSC (Message Sequence Chart) [38], como visto nas Fig. 4.10, 4.11 e 4.12, que apresenta a perda de um sinal que chega em um estado onde não é tratado na especificação, e que foi resolvido após a verificação desse erro.

A validação auxilia o desenvolvedor a encontrar erros das especificações e também a visualizar o funcionamento das especificações. A validação percorre todos os caminhos possíveis dos sinais, encontrando possíveis erros como perda de sinais, não tratamento de erros, entre outros. A simulação com auxílio do MSC, ajuda ao desenvolvedor a estipular os caminhos que os sinais devem percorrer, permitindo ao desenvolvedor a analisar o sinal que deseja.

A opção de validação determinada pelo *Bit State Exploration* se constitui em um algoritmo eficiente para sistemas SDL de porte médio, enquanto que o *Random Walk*, é um algoritmo mais simples que pode ser utilizado em sistemas SDL mais complexos[16].

A utilização da validação e da simulação, realizadas passo-a-passo, e verificando suas ações através do MSC, auxilia o desenvolvedor, a encontrar automaticamente pelo uso da ferramenta SDT, detalhes como perda de sinais e tratamentos de erros, que possam ter sido esquecidos ou não tratados durante a fase de análise do sistema a ser modelado. A utilização desses recursos são importantes para a implementação dos projetos, pois implica no desenvolvimento de um sistema confiável.

Capítulo 5

Conclusão

Há alguns anos atrás, as propostas de rede ATM, eram vistas apenas como uma solução para integração de diferentes tipos de tráfegos. Hoje, a maior preocupação é o de interligar redes locais a essa rede ATM, já que ela pode ser utilizada na comunicação de redes públicas como a B-ISDN.

Além do ITU-T, existem órgãos como o IETF que faz as padronizações dos protocolos que são executados sobre a rede Internet, e o ATM Forum, criado pelas empresas privadas para desenvolver e acelerar o desenvolvimento dos dispositivos em redes ATM. Esses órgãos estão preocupados com a compatibilização do endereçamento entre essas duas redes.

A Internet é uma rede que interliga mundialmente diferentes tipos de redes locais, possui uma grande importância por ser uma rede já utilizada amplamente. A rede ATM por sua vez, possui diversas vantagens na sua utilização, tal como qualidade de serviço, possibilidade de atender tráfegos diversos como alta velocidade, entre outros. A conversação entre essas duas redes é muito importante para obter na Internet as vantagens oferecidas com ATM.

Com o surgimento de várias propostas de interconexão, como IP Clássico, NHRP, MARS, LANE e MPOA, os problemas de interconexão de cada uma dessas propostas tem sido estudadas. Os protocolos IP Clássico e NHRP, apresentados nos RFC1577, RFC 1735 e alguns *drafts* do IETF, foram escolhidos neste trabalho por apresentarem grupos abertos a discussão desses protocolos e não possuírem uma especificação concluída desses protocolos podendo assim explorar melhor suas características.

A proposição de uma metodologia de projeto orientado a objeto aplicada a uma linguagem formal como SDL'92 (Specification and Description Language, versão de 1992, ITU-T) para especificação de protocolos, trouxe grandes vantagens como no tempo gasto para a construção das especificações, que teve sua maior parte dedicada na análise dos protocolos e na realização do Sistema *Cloud* que foi herdado por todos os outros sistemas. A dedicação maior na validação e simulação desse sistema, também se explica pelo motivo de garantir uma especificação com maior

grau de corretude para ser aplicada em outras especificações. O uso da modularidade das classes, também auxilia no processo de herança dos sistemas, permitindo que os sistemas herdassem apenas as classes que lhes eram necessárias.

A utilização de uma ferramenta CASE como o SDT (SDL Design Tool, Telelogic - Suécia), traz grandes vantagens na realização das especificações. Além de facilitar a realização das especificações utilizando recursos gráficos da linguagem SDL, a ferramenta auxilia em um melhor entendimento do funcionamento da especificação, assim como o de localizar erros não tratados nas especificações, ou na realização incorreta dessa, encontradas com o auxilio do validador, e/ou simulador utilizando gráficos temporais MSC (Message Sequence Chart, ITU-T) como ferramenta de auxílio.

Neste trabalho, baseando-se nas recomendações dos protocolos IP Clássico e NHRP, foram especificados, utilizando projeto orientado a objeto e SDL, sistemas que descrevem o funcionamento desses protocolos. As simulações e validações desses sistemas, possibilitam além de encontrar erros de especificação, analisar o comportamento desses protocolos mediante a tratamento de sinais, *loops*, roteamento, entre outros.

A reutilização das especificações apresentadas nesse trabalho, podem ser aproveitadas para trabalhos futuros como:

Especificação de um MARS (Multicast Address Resolution Server) para implementar o *multicasting* no protocolo IP Clássico.

Especificação do protocolo MPOA (Multi Protocol Over ATM) visto que esse protocolo apresenta a funcionalidade dos protocolos IP Clássico, NHRP, MARS e LANE.

Construir um trabalho comparativo das funcionalidades desses protocolos, a partir das especificações desses protocolos para determinar as vantagens e desvantagens de cada protocolo em determinadas situações.

Em resumo, este trabalho mostra que o emprego de uma metodologia aqui proposta de projeto orientado a objeto como OMT para construir um projeto de objetos, auxilia num melhor desenvolvimento de especificação de sistemas utilizando uma linguagem de especificação formal como SDL que possui recursos gráficos e de uma ferramenta que auxilia o desenvolvedor a

realizar projetos com maior confiabilidade, desenvolver características para sua evolução em um prazo menor de tempo, são fatores importantes para a modelagem, teste e implementação de sistemas para um mercado competitivo em que a tecnologia evolui muito rapidamente.

Indice Remissivo

A

AAL, 11; 12; 14; 25
ATM Forum, 1; 3; 5; 17; 18; 22; 26; 27; 88

B

B-ISDN, 1; 5; 10; 17; 26; 88
Block Type, 33; 38; 39; 42; 44; 45; 47; 51; 52; 54; 61; 66; 69; 79

C

Cache, 23; 35; 38; 44; 52; 53; 54; 56
camada ATM, 11; 12; 14
camadas ATM, 1
Classe A, 12
Classe B, 12
Classe C, 13
Classe D, 13

E

especificação formal, 2; 3; 28; 30; 31; 33; 51; 90

H

Host, 9; 10; 19; 20; 23; 25; 34; 38; 44; 45; 46; 52; 54; 81

I

IETF, 1; 2; 3; 5; 17; 18; 24; 27; 88
instâncias, 45; 46; 52; 54; 66; 78; 79; 84
ISDN, 1
ITU-T, 5; 17; 88; 89

L

LANE, 1; 17; 18; 26; 27; 88; 89
LIS, 19; 21; 23; 24; 25
loops, 23; 24; 58; 62; 63; 64; 68; 84; 85; 86; 89

M

MAC, 17; 18
MARS, 1; 18; 25; 26; 27; 88; 89
Modelo de objetos, 34
MPOA, 1; 18; 26; 27; 88; 89
MSC, 2; 30; 32; 33; 74; 80; 82; 84; 86; 87; 89

N

NBMA, 22; 23; 24; 47; 61; 63; 64
NHS, 23; 24; 86
NNI, 15; 17; 22

O

OMT, 2; 3; 4; 5; 22; 90

P

Package, 30; 39; 40
Pid, 43; 45; 46; 54; 79; 81
Process Type, 33; 39; 42; 44; 45; 47; 52; 56; 61; 69
protocolo IP Clássico, 2; 21; 22; 23; 25; 27; 84; 86; 89
protocolo NHRP, 2; 21; 23; 24; 85; 86
protocolo TCP/IP, 1
Proxy-ARP, 3; 4; 22; 28; 36; 63; 64; 65; 66; 67; 71; 72; 77; 80; 85
PVC, 16; 19; 21; 82

Q

QoS, 1; 10; 14; 18; 19

R

rede ATM, 1; 5; 10; 14; 15; 16; 17; 19; 20; 21; 27; 88
rede NBMA, 23; 24
RFC 1577, 22; 24; 25; 26; 68
RFC 1735, 22; 24; 88
RFC 1932, 22; 24; 68
Router, 34; 38; 45; 47; 63; 69

S

SDL, 2; 3; 4; 5; 22; 74; 81; 82; 83; 87; 88; 89; 90
SDL'92, 3; 28; 30; 31; 32; 33; 34; 35; 37; 39
SDT, 2; 3; 4; 75; 78; 87; 89
Servidor, 21; 23; 34; 38; 44; 47; 53; 59; 60; 64; 68; 70
Sistema *Cloud*, 3; 4; 76; 77; 79; 80; 88
Sistema IP Clássico, 3; 4; 76; 77; 79; 84
Sistema IP Clássico Migrado e NHRP, 36
Sistema IP Clássico Migrado, 3; 4
Sistema IP Clássico migrado para o protocolo NHRP, 36
Sistema NHRP, 3; 4; 77; 80
Sistema NHRP que se comunica através do protocolo *Proxy-ARP*, 4
Subrede, 7; 34; 59
SVC, 16; 19; 21

T

TCP/IP, 1; 2; 5; 6; 7; 9; 84

U

UNI, 2; 10; 15; 17; 25; 26

V

VCI, 11; 16; 22
VPI, 11; 16; 22

Siglas e Acrônimos

AAL - ATM Adaptation Layer

ABR - Avaible Bit Rate

ADT - Abstract Data Type

ARP - Address Resolution Protocol

ASN.1 - Abstract Syntax Notation - One

ATM - Asynchronous Transfer Mode

ATMARP - ATM Address Resolution Protocol

BRISA - Sociedade Brasileira para Interconexão de Sistemas Abertos

BT - Burst Tolerance

BUS - Broadcast/Unknown Server

CASE - Computer Aided Software Engineering

CBR - Constant Bit Rate

CCITT - Comité Consultatif International Téléphonique

CDV - Cell Delay Variation

CLP - Cell Loss Priority

CLR - Cell Loss Ratio

CORBA - Common Object Request Broker Architecture

CTD - Cell Transfer Dalay

EGP - Exterior Gateway Protocol

ELAN - Emulated LAN

FDDI - Fiber Distributed Data Interface

FDT - Formal Description Technique

FSM - Finite State Machine

FTP - File Transfer Protocol

GFC - Generic Flow Control

HEC - Header Error Check

ICMP - Internet Control Message Protocol

IDL - Interface Definition Language

IEEE - Institute of Electrical and Eletronics Engineers

IETF - Internet Engineering Task Force

IGP - Interior Gateway Protocol

IP - Internet Protocol

IPX - Internetwork Packet Exchange

ISDN - Integrated Services Digital Network

ISO - International Starndartization Organization

ITU-T - International Telecommunication Union, Telecommunication Standatization Sector

LAN - Local Area Network

LANE - LAN Emulation

LE - ARP - LAN Emulation - Address Resolution Protocol

LEC - LAN Emulation Client

LECS - LAN Emulation Configuration Server

LES - LAN Emulation Server

LIS - Logical IP Subnetwork

LLC - Logical Link Control

LOTOS - Language of Temporal Ordering Specification

MAC - Medium Access Control

MARS - Multicast Address Resolution Server

MCR - Minimum Cell Rate

MPOA - Multiprotocol Over ATM

MTU - Maximum Transmission Unit

NBMA - Non Broadcast Multi Access

NDIS - Network Driver Interface Specification

NHRP - Next Hop Resolution Protocol

NHS - NHRP Server

NIC - Network Interface Cards

NNI -Network-Node Interface

NPDU - Network Protocol Data Unit

NSAP - Network Service Access Point

ODI - Open Datalink Interface

OMT - Object Modelling Technique

OOA - Object Oriented Analisys

OOB - Out Of Band

OSI - Open Systems Interconection

OSPF - Open Shortest Path First

PCR - Peak Cell Rate

PDU - Protocol Data Unit

PNNI - Private NNI

PTI - Payload Type Identifier

PVC - Permanent Virtual Connections

QoS - Quality of Service

RDSI - Rede Digital de Serviços Integrados

RDSI-FE - RDSI de Faixa Estreita

RDSI-FL - RDSI de Faixa Larga

RFC - Request For Comments

RIP - Routing Information Protocol

ROLC - Routing Over Large Clouds Working Group

SAA - Service Aspects and Applications

SCR - Sustained Cell Rate

SDL - Specification and Description Language

SDT - SDL Design Tool

SMDS - Switched Multimegabit Data Service

SMTP - Simple Mail Transfer Protocol

SNAP - SubNetwork Access Protocol

SNMP-Simple Network Management Protocol

SOMT - SDL-oriented Object Modelling Technique

SPF - Shortest Path First

SVC - Switched Virtual Connections

TCP - Transmission Control Protocol

TCP/IP - Transmission Control Protocol/Internet Protocol

TTCN - Tree and Tabular Combined Notation

UDP - User Datagram Protocol

UNI - User-Network Interface

VBR - Variable Bit Rate

VC - Virtual Connection

VCI - Virtual Channel Identifier

VPI - Virtual Path Identifier

WAN - Wide Area Network

Referências Bibliográficas

[1] - Ales,Antony; *ATM Internetworking*. Cisco Systems, Inc. 1995. http://www.cisco.com/warp/public/614/12.html.

[2] - Andrews, Eric; *MPOA Ties It All Together*, Data Communications on the web. Abril 1996. http://www.data.com/Tutorials/MPOA_Ties_It_All_Together.html

[3] - Armitage, G.J.; *Multicast and Multiprotocol Support for ATM based Internets*. ACM SIGCOMM - Computer Communication Review, april 1995. pp. 34 - pp.46.

[4] - ATM Forum - *Baseline Text for MPOA*. June 1996. ftp.atm.forum.com pub/conbtributions/atm95-0824.txt.

[5] - ATM Forum - *LAN Emulation Client Management Specification Version 1.0*. September 1995. ftp.atm.forum.com pub/conbtributions/af-lane-0050.000..txt.

[6] - ATM Forum - *LAN Emulation Over ATM Version 1.0*. January 1995. ftp.atm.forum.com pub/conbtributions/af-lane-0021.000..txt.

[7] - ATM Forum - *LAN Emulation Server Management Specification Version 1.0*. March 1996. ftp.atm.forum.com pub/conbtributions/af-lane-0057.000..txt.

[8] - Belina, F., Hogrefe, D. e Sarma, A.; *SDL with Applications from Protocol Specification*. Prentice-Hall, 1991.

[9] - Booch, Grady; *Objetc-Oriented Design With Applications*, 1991. The Benjamin/Cummingins Publishing Company, Inc.

[10] - Carvalho, Tereza Cristina Melo de Brito; *Arquitetura de redes / Brisa*. Organização e coordenação:. Editora Makron Books. 1994.

[11] - Diaz, M., Ansart, J. P., Courtiat, J. P., Azama, P., Vijaya C., *The Formal Description Technique Estelle*. North-Holland, 1989.

[12] - Eijk, P. H. J. van, Vissers C. A., Diaz, M. *The formal Description Technique LOTOS* Editora North Holland, Elsevier Science Publishers, 1989.

[13] - Færgemand, Ove e Olsen, Anders; *New Features in SDL-92*. TFL - Telecommunications Research Laboratory. http://www.tdr.dk/public/SoftwareEngineering /SDL/doc/sdltutor.ps.

[14] - Garalski, Walter J.; *Introduction to ATM Networking*, 1995. McGraw-Hill.

[15] - Grael, Pedro; *Suporte ao Serviço não orientado à Conexão na Rede Digital de Serviços Integrados de Faixa Larga*. Tese de mestrado, Faculdade de Engenharia Elétrica, UNICAMP. Maio de 1994.

[16] - Henninger, O., Barbeau, M. e Sarikaya, B.,; *Specification and Testing of the behavior of Network Management Agents Using SDL-92*. IEEE/ACM Transations on Networking, volume 4, número 6. December 1996. pp. 951 - pp. 962.

[17] - Holzmann, Gerard. J. , *Design and Validation of Computer Protocol*. Prentice-Hall, 1991.

[18] - Hüni, H., Johnson, R. e Engel, R.,; *A Framework for Network Protocol Software*. ACM SIGPLAN NOTICES, OOPSLA'95 Conference Proceedings, Object-Oriented Programming Systems, Languages and Applications. pp. 358 - pp.369.

[19] - IETF - Draft - Smith, J. e Armitage, G. <draft-ietf-ion-bcast-00.txt> *IP Broadcast over ATM Networks*. August 1996. http://ds.internic.net/ds/rfc.index.html.

[20] - IETF - Draft - Laubach, Mark; <draft-ietf-ion-classic2-00.txt> *Classical IP and ARP over ATM.* June 1996. http://ds.internic.net/ds/rfc.index.html.

[21] - IETF - Draft - Luciani, J. V. , Katz, D., Piscitello, D. e Cole, B.; <draft-ietf-rolc-nhrp-10.txt> *NBMA Next Hop Resolution Protocol NHRP.* March 1997. http://ds.internic.net/ds/rfc.index.html.

[22] - IETF - Draft - Luciani, J. V.; <draft-ietf-ion-transition-00.txt> *Classical IP to NHRP transition.* March 1997. .http://ds.internic.net/ds/rfc.index.html.

[23] - IETF - Draft - Talpade e Ammar; <draft-ietf-ion-marsmcs-00.txt> *Multicast Server Architecture for MARS-based ATM multicasting.* July 1996. http://ds.internic.net/ds/rfc.index.html.

[24] - IETF - RFC 1112 - Deering, S.; *Host Extensions for IP Multicasting.* August 1989. http://ds.internic.net/ds/rfc.index.html.

[25] - IETF - RFC 1180 - Socolofsky, T. e Kale, C.; *A TCP/IP Tutorial.* January 1991. http://ds.internic.net/ds/rfc.index.html.

[26] - IETF - RFC 1433 - Garret, J. , Hagan J. e Wong, J. ; *Directed ARP.* March 1993. http://ds.internic.net/ds/rfc.index.html.

[27] - IETF - RFC 1483 - Heinanen J.; *Multiprotocol Encapsulation over ATM Adaptaion Layer 5.* July 1993. http://ds.internic.net/ds/rfc.index.html.

[28] - IETF - RFC 1577 - Laubach, M.; *Classical IP and ARP over ATM.* January 1994. http://ds.internic.net/ds/rfc.index.html.

[29] - IETF - RFC 1620 - Braden B., Postel J. e Rekhter Y. *Internet Architecture Extensions for Shared Media.* May 1994. http://ds.internic.net/ds/rfc.index.html.

[30] - IETF - RFC 1735 - Heinanen, J.; *NBMA Adress Resolution Protocol (NARP).* December 1994. http://ds.internic.net/ds/rfc.index.html.

[31] - IETF - RFC 1755 - Perez, M., Liaw, F., et al. *ATM Signaling Support for IP over ATM.* February 1995. http://ds.internic.net/ds/rfc.index.html.

[32] - IETF - RFC 1932 - Cole, R., Shur, D. e Villamizar, C.; *IP Over ATM*: A Framework Document. April 1996. http://ds.internic.net/ds/rfc.index.html.

[33] - IFIP - *Proceeding of the IFIP WG.6.1 Ninth Symposium on Protocol Specification, Testing, and Verification IX*, Enschede, The Netherland 6-9 June, 1989. Elsevier Science Publishers, North-Holland, 1990.

[34] - IFIP - *Proceeding of the IFIP WG.6.1 Eighth Symposium on Protocol Specification, Testing, and Verification VIII*, Atlantic City, New Jersey, USA, 7-10 June, 1980. Elsevier Science Publishers, North-Holland, 1989.

[35] - Interfase Corporation. *An Overview of LAN Emulation.* http://www.iphase.com/Public/Products/Technology/ATM/WP/LAN.html

[36] - ITU - T Recomendação Z.100. CCITT *Specification and Description Language* (SDL), Helsinki, 1993 .

[37] - ITU - T Recomendação Z.100. CCITT *Specification and Description Language* (SDL), Helsinki, 1989 .

[38] - ITU - T Recommendation Z.120 - *Message Sequence Chart (MSC)* Helsinki 1993.

[39] - Jacobson, I. et al; *Object-Oriented Software Engineering*, Addison-Welwy, 1992.

[40] - Marshall, G.; *Classical IP Over ATM: A Status Report.* Data Communications on the web. December 1995. Tech Tutorial. http://www.data.com/Tutorials/ Classical_IP_Overl_ATM.html

[41] - Martins, J., Hubaux, J.P., Saydam, T. e Znaty, S.; *Integrating Performance Evaluation and Formal Specification* . 1996 IEEE International Conference on Communications. Volume I. pp. 1803 - 1807.

[42] - Peeters, J., Cuypers, L. e Jonckers, V.; *Using OMT in Telecommunications Systems Requirements Analysis; an Experience Report and Critical Appraisal.* June 1995. Insyde Project.

[43] - Peeters, J., Cuypers, L., Schoovaerts, K. e Jonckers, V.; *Capturing Telecommunications Systems Requirements in OMT; an Experience Report.* September 1995. Insyde Project.

[44] - Prycker, Martin de; *Asynchronous Transfer Mode: solution for broadband ISDN.* terceira edição. Editora Prentice-Hall, 1995.

[45] - Rumbaugh, James et al. *Modelagem e projetos baseados em objetos.* Editora Campus. 1994.

[46] - Saracco, Robert, Smith, J. R. W e Reed, Rick; *Telecommunications Systems Engineering using SDL* North-Holland, 1989.

[47] - Sharma, R. e Keshav, S. *Signaling and Operating System Support for Native-Mode ATM Applications.* ACM - SIGCOMM Communication Review, august 1994, pp. 149 - pp. 157.

[48] - Siu, Kai Yeung e Jain, Raj. *A brief overview of ATM: Protocol Layers, LAN Emulation, and Traffic Management.* ACM SIGCOMM - Computer Communication Review, april 1995, pp. 6 - pp. 20.

[49] - Soares, Luis Fernando Gomes. *Redes de Computadores*. Editora Campus, 1995.

[50] - Takahara, Jhuli M. e Borelli, Walter C. *Especificação de Protocolos de Interconexão de Redes Locais e ATM utilizando OMT e SDL'92*, Anais do Simpósio Brasileiro de Telecomunicações, setembro 1997.

[51] - Tanembaum, Andrew S. *Computer Networks*. Terceira edição. Editora Prentice-Hall. 1996.

[52] - Tanembaum, Andrew S. *Computer Networks*. Segunda edição. Editora Prentice Hall. 1989.

[53] - Taylor, Ed *The McGraw-Hill internetworking handbook*, McGraw-Hill, 1995.

[54] - Telelogic AB(Malmö, Sweden *Combining Object-Oriented Analysis and SDL Design,* Malmö - 1996. http://www.telelogic.se/.

[55] - Telelogic AB(Malmö, Sweden), Ek, Anders *The SOMT Method,* Malmö - September 1995. http://www.telelogic.se/ .

[56] - Telelogic AB(Malmö, Sweden), *Getting Started with SDT3.02* Malmö - 1995.

[57] - Telelogic AB(Malmö, Sweden), *Getting Started with SDT3.1, Part2 - SOMT and CORBA Tutorials* Malmö - 1996.

[58] - Telelogic AB(Malmö, Sweden), *Methodology SDT3.02* Malmö - 1995.

[59] - Telelogic AB(Malmö, Sweden), *Methodology Guidelines, SDT3.1, Part1 - The SOMT Method,* Malmö - 1996.

[60] - VIVID Newbridge, *Building Next Generation LAN Internetworks.* http://www.vivid.newbridge.com/documents/Eric.html/network.html

[61] - VIVID Newbridge. *Multiprotocol Over ATM (MPOA).* http://www.vivid.newbridge.com/documents/Eric.html/MPOAarticle.html

Apêndice A ___ *102*

Linguagem SDL (Specification and Description Language) ___ **102**

A.1 - Introdução ___ 102
A.2 - SDL'88 ___ 103
A.3 - SDL'92 ___ 104
A.4 - Notação SDL ___ 107

Apêndice B ___ *111*

SDT (SDL Desing Tool) ___ **111**

Figura A. 1 - Estrutura de Especificação em SDL ... 103
Figura A. 2 - Exemplo de conexão N-M ... 106
Figura A. 3 - Exemplo do uso do any e de Procedimento que retorna valor ... 106
Figura A. 4 - Símbolos do SDL [46] ... 109

Figura B. 1 - Visão geral do Funcionamento do SDT [58] ... 112
Figura B. 2 - Metodologia SOMT [54] ... 114

Tabela A. 1 - Tabela de equivalência de conceitos orientados a objeto e SDL'92 [58] ... 107

Apêndice A

Linguagem SDL (Specification and Description Language)

A.1 - Introdução

A linguagem SDL (Specification and Description Language) [36] [37] foi publicada primeiramente como uma recomendação do CCITT (Comité Consultatif International Téléphonique), no começo dos anos 70. Atualmente o órgão CCITT é representado pelo ITU-T (International Telecommunication Union, Telecommunication Standatization Sector).

As características básicas da linguagem SDL são apresentados como SDL'88 , pois sua especificação [32] foi consideravelmente expandida e utilizada em ferramentas gráficas como o SDT 2.0 [3] . Com a crescente utilização de recursos orientado a objeto nas linguagens de programação, a linguagem SDL, também ganhou esses recursos, e passou a ser chamada de SDL'92 [36].

Devido a sua facilidade de aprendizado e disposição de ferramentas gráficas, além dos vários recursos disponíveis, a linguagem SDL tem sido amplamente difundida, não somente na área de telecomunicações, mas também em áreas de Automação, Banco de Dados e Engenharia de Software e sistemas de tempo-real. A linguagem SDL é uma linguagem formal utilizada para representar o modelo comportamental de um determinado sistema.

A apresentação do SDL abriu um horizonte maior para especificação de sistemas, pois com suas novas características tornou-se possível especificar vários sistemas. A característica orientado a objeto trazida na nova especificação (SDL'92), trouxe grandes vantagens na modelagem de sistemas, como a possibilidade de reuso.

[3] Versão da Ferramenta SDT que apresentava recursos da linguagem SDL'88.

A.2 - SDL'88

A linguagem SDL é uma linguagem de especificação formal cujo comportamento é modelado através de interação de máquinas de estados finitos [46] [8]. A especificação do sistema a ser modelado é divida em níveis hierárquicos, de acordo com seu nível de abstração:

1. Sistema
2. Bloco
3. Processo
4. Serviço
5. Procedimento e Macro

A estrutura dessas hierarquias pode ser vista na figura abaixo:

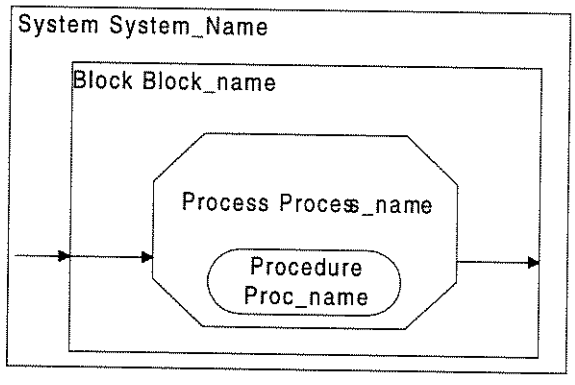

Figura A. 1 - Estrutura de Especificação em SDL

O sistema apresenta os blocos e os sinais que serão trocados pelos blocos. Dentro dos blocos são apresentados os processos de cada bloco, apresentando os sinais por eles trocados. Os processos possuem um maior detalhamento da especificação, é neles e nos procedimentos que são apresentados o comportamento do sistema.

A versão SDL'88 foi adotada para utilização de ferramentas gráficas para a especificação de sistema de telecomunicações. O recurso gráfico da linguagem facilita o entendimento da programação. Apesar das vantagens apresentadas na especificação com o auxilio de uma

ferramenta CASE [50], a linguagem SDL'88 apresenta muitas limitações que dificulta a programação. Para resolver essas limitações, é que foi criada a nova versão da linguagem, conhecida como SDL'92.

A.3 - SDL'92

A recomendação SDL que apresenta recursos orientado a objeto, é bastante conhecida como SDL'92. Seu funcionamento básico é conhecido como SDL'88. O funcionamento das antigas estruturas de sua recomendação como blocos, processos entre outros, permanecem sem alterações, apenas conceitos novos como *System Type, Block Type* e *Process Type* foram adicionados [11]. Essas novas estruturas foram adicionadas para permitir o reuso, através da instanciação ou herança.

A hierarquia da linguagem SDL com as novas estruturas e:

1. Sistema
2. *Package*
3. *System Type*
4. *Block Type* ou Bloco
5. *Process Type* ou Processo
6. Macro e procedimento

A instanciação é feita pela descrição de vários blocos ou processos de um único tipo, criando vários processos e blocos, sem a necessidade de modelá-los novamente. A herança é proporcionada pelo uso do ***inherits*** na estrutura (sistema, bloco ou processo) em que irá herdar as características. Uma estrutura herda determinadas características que estão armazenadas nos *Packages*, que funciona como uma biblioteca de tipos que poderão ser reutilizadas. No sistema que utiliza herança, deve-se especificar com o ***use*** o *package* utilizado.

Um sistema, bloco ou processo pode herdar características de um *System Type, Block Type* e *Process Type* respectivamente. A herança dessas estruturas em SDL não limita a adição ou modificação nas especificações em SDL. O uso da cláusula ***Virtual*** na super-classe, indica que uma estrutura, estado, procedimento, pode ser modificado a partir do ponto em que aparece essa cláusula.

Características novas para tratamento de dados também foram acrescentadas. Essas características são muito importantes para a utilização de ADTs (Abstract Data Type). A definição de tipos em SDL é dada pelo uso do operador ***Newtype***. Esse operador é utilizado para definir tipos como *arrays, Literal,* entre outros. A versão SDL'92 adicionou à definição de novos tipos a possibilidade de especificar esses tipos definições algorítmicas e alternativas., utilizando operadores.

Os blocos, processos e símbolos que são redefinidos são apresentados como ***Redefined***, o que indica a possibilidade de adicionar e modificar certas características. Quando utilizado o ***Finalized*** no lugar do ***Redefined*** significa que a partir desse ponto não é possível realizar alterações.

Uma outra característica nova que foi implementada no SDL'92 é o uso dos ponto de conexões (*gates*). Os *gates* são utilizados para diferencias a entrada ou saída de sinais de cada instância de bloco ou processo. Nas especificações do SDL'88 não existia a presença dos *gates* dessa forma, nos limites de blocos e processos poderia ser apenas de 1-N, pois como não haviam pontos de conexão, não era possível que os blocos pudessem enviar sinal por um mesmo canal utilizado por outro bloco. Na especificação SDL'92, a presença dos *gates* possibilita a conexão dos pontos de N-M, pois é possível que vários blocos que conectem através de um mesmo ponto de conexão. Essa característica foi resolvida pelo uso do termo ***VIA*** na saída dos sinais dos processos.

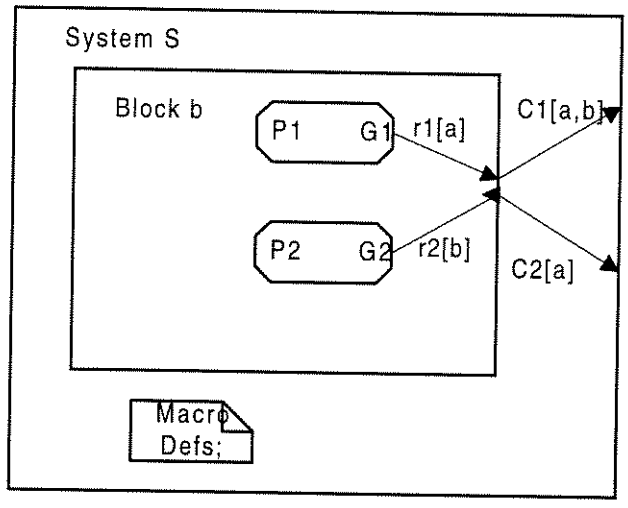

Figura A. 2 - Exemplo de conexão N-M

A versão SDL'92 possibilita construir procedimentos globais, eliminando assim a duplicação de procedimentos em vários processos diferentes. Outra característica nova apresentada é a utilização do ***Any*** que gera aleatoriamente qualquer valor do tipo especificado. A possibilidade de um procedimento retornar valores também é uma característica nova a linguagem graças ao uso dos parâmetros ***in/out***.

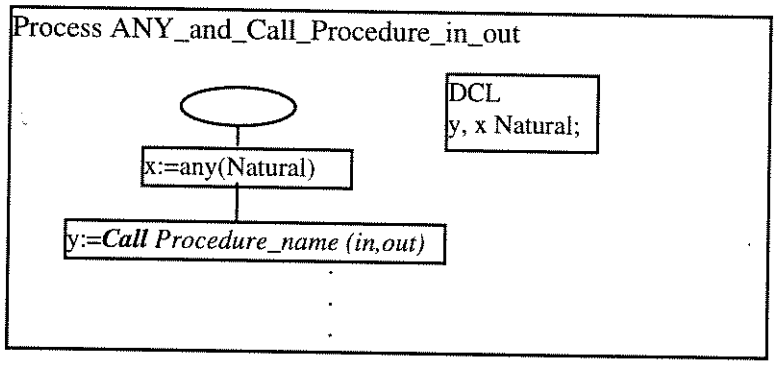

Figura A. 3 - Exemplo do uso do any e de Procedimento que retorna valor

Além dessas características novas apresentadas, ainda existem diversas vantagens adicionadas a especificação do SDL'92, essas novas características foram introduzidas para facilitar a especificação de sistemas e eliminar as limitações existentes na antiga versão.

A adição de recursos de orientação a objeto para especificações com esse recurso, cria-se a equivalência de certos conceitos de orientação a objeto na linguagem SDL'92, como apresentado no *methodology* da ferramenta SDT [58], e descrito na tabela A.1.

OOA	SDL'92
Class (no outer frame around the object)	Process type (alternatively ADT).
Class and object	Process Type (Alternatively ADT) and instances of this class in the system.
Attributes	Process variables.
Services	Signals. **Note:** The SDL'92 signal concept does not cover the whole service concept: most services returns a value; a service can even contain states!
External Input/Output	Signals to the enviroment.
Criate	Formal Parameters and start transitions in processes.
Connect (an implicit service)	Signalling between processes to be introduced to themselves.
Accesss (an implicit service)	Can be implemented with Export/Import - View/Revealed for the reading of variables in the other objects, or ordinary assignments if they are internal variables.
Groups	Block Types/block set.
Scenarios	MSC diagrams
Generalization/Specification	Inheritance using specialization.
Whole/part of structure	Block Types/ Block sets to some extent. Cardinality can be shown both for blocks and processes.
Object Relation	Cardinality can be shown both for blocks and processes.
Services Charts	Informal Process descriptions to describe a service.

Tabela A. 1 - Tabela de equivalência de conceitos orientados a objeto e SDL'92 [58]

A.4 - Notação SDL

Na versão SDL'88 existem vários termos utilizados na modelagem . Para utilização de ***Timer***, são utilizados os comandos ***Set/Reset*** para iniciar o valor e limpar o valor do temporizador utilizado na especificação.

A necessidade de se utilizar endereços para localizar em qual instância enviar o sinal, exige o emprego de *Pid*. Para saber o endereço de cada instância, é utilizado os seguintes comandos.

- ***Offspring*** - Retorna o endereço do processo criado.
- ***Sender*** - Retorna o endereço do Processo que enviou um determinado sinal.
- ***Self*** - Retorna o valor do processo.
- ***Parent*** - Tendo o *Pid* de um processo, esse comando retorna o valor do processo que o criou .

A linguagem SDL'92 apresenta diversos termos que facilitam a modelagem de sistema. Esses termos auxiliam na especificação e acrescenta característica orientado a objeto. Alguns desses termos introduzidos para modelagem são:

- ***Any*** - Gera aleatoriamente um valor correspondente ao tipo solicitado.
- ***Block Type*** - Tipo de um bloco utilizado para instanciação.
- ***Call*** - Chamada de Procedimento que retorna valor.
- ***Finalized*** - Tipo de Redefinição que não permite que as classes que o herdem alterem esta estrutura.
- ***Inherits*** - Indica a herança de determinada parte em um sistema.
- ***Package*** - Biblioteca de classes que poderão ser herdadas.
- ***Process Type*** - Tipo de Processo que pode ser instanciado.
- ***Redefined*** - Indica que determinada parte da especificação foi modificada.

- ***System Type*** - Instanciação de sistema.

- ***Use*** - Aponta o Package que contém as classes que serão herdadas.

- ***Via*** - Aponta o *gate* em que a mensagem será encaminhada.

- ***Virtual*** - Aponta as partes que podem ser alteradas.

Os símbolos mais adotados para especificação de sistemas utilizando a linguagem SDL podem ser vistos na Figura A.4 [46].

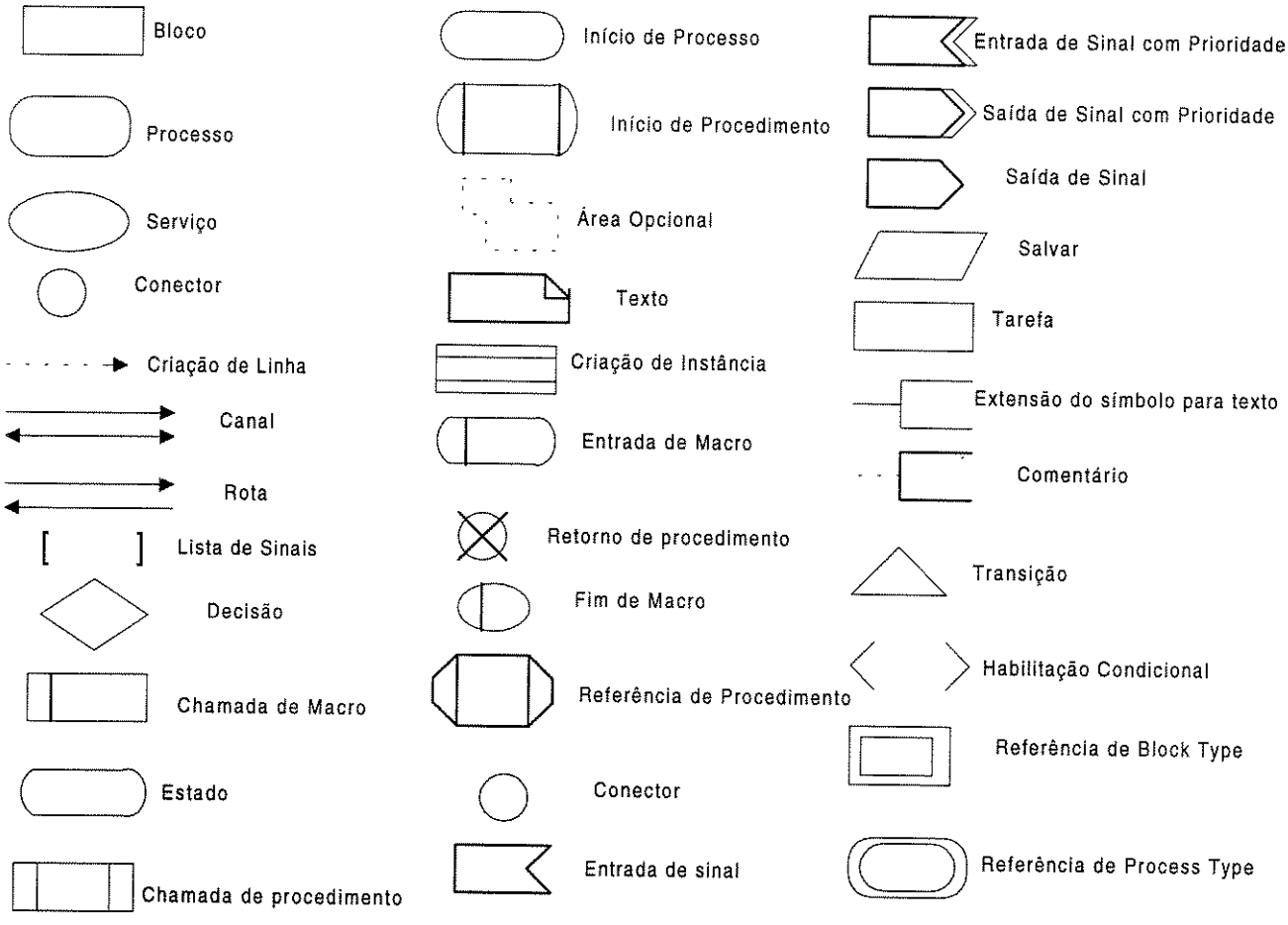

Figura A. 4 - Símbolos do SDL [46]

A partir da utilização desses símbolos, ferramentas CASE, como o SDT [56] [57] [58] [59], são utilizadas para facilitar o modelamento de sistemas, analisar a corretude da especificação, validar e simular o sistema especificado, trazendo grandes vantagens antes de se

realizar a implementação. Uma breve descrição sobre os recursos da ferramenta SDT utilizada esta descrita no Apêndice B.

Apêndice B

SDT (SDL Desing Tool)

O SDT (SDL Design Tool) [56] [57] [58] [59] é uma ferramenta CASE (Computer Aided Software Engineering) que utiliza a linguagem SDL para especificar, validar e simular um determinado sistema descrito. A versão utilizada, SDT[4] 3.02, possui os seguintes componentes:

1. Editor SDL, edita o comportamento do sistema na linguagem SDL;

2. Analisador e Make, faz a análise sintática e semântica para verificação de erros e gera o código para validação e simulação;

3. Simulador, simula as características do sistema de acordo com a determinação do sinal a ser tratado;

4. Validador, percorre todos os caminhos possíveis do sistema. Esse componente auxilia no encontro de erros e tratamento de erros não especificados.

5. Editor de MSC (Message Sequence Chart), esse componente auxilia a validação e simulação para acompanhar os passos que estão sendo realizados nesses dispositivos. O MSC possui uma iteração gráfica que facilita o desenvolvedor, acompanhar e analisar as tarefas que estão sendo executadas.

Essa ferramenta apresenta o uso de recursos gráficos para especificação de sistema, o que facilita o desenvolvimento aos programadores, além de disponibilizar recursos que tornam a especificação mais correta. Como é o caso do *Validator* que percorre todos os casos possíveis, encontrando erros de especificação, como não tratamento de erros não previstos, perdas de sinais, entre outros. O uso do MSC [33] torna possível visualizar passo-a-passo as tarefas que estão sendo executadas pelo sistema.

[4] Versão utilizada para simular, validar e especificar os sistemas apresentados neste trabalho.

Além de utilizar a linguagem SDL para realizar as especificações, essa ferramenta possibilita fazer a interface com outras linguagens como TTCN (Tree and Tabular Combined Notation) e ASN.1 (Abstract Syntax Notation One). Uma visão geral do funcionamento e seus aplicativos, pode ser visto na figura abaixo:

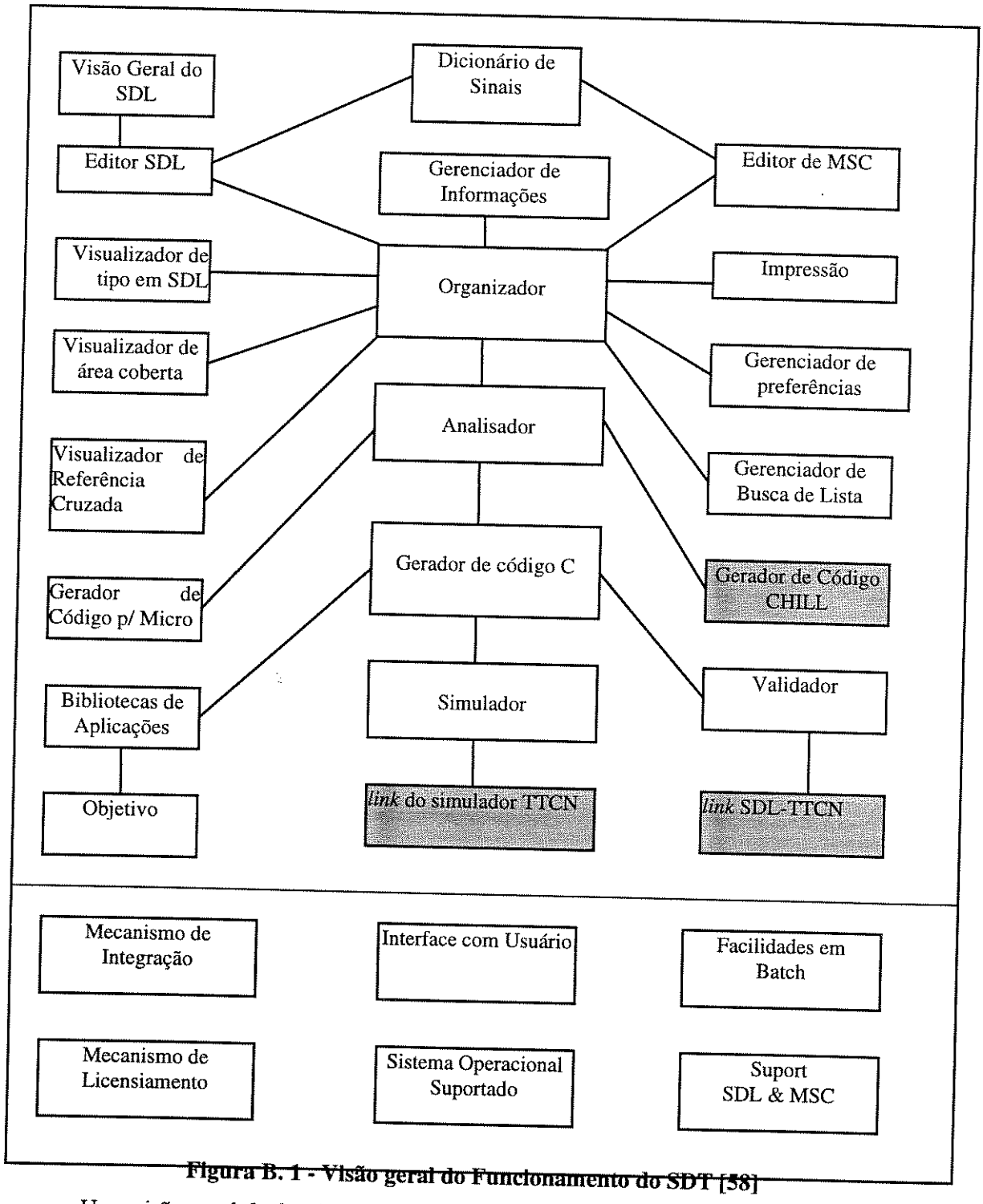

Figura B. 1 - Visão geral do Funcionamento do SDT [58]

Uma visão geral do funcionamento da ferramenta SDT, pode ser vista na Figura C.1, que descreve com maiores detalhes os aplicativos existentes na ferramenta e dos recursos que ela

disponibiliza para realizar suas tarefas. As características que estão presentes nas caixas escuras, são as que não são apresentadas no pacote utilizado para a realização deste trabalho.

A nova versão do SDT 3.1[5] , possui várias vantagens além das utilizadas para a realização desse trabalho. Essa versão emprega um editor de OMT utilizando conceitos como *Paste as* e *Implink*, para integrar o editor SDL e o editor OMT. Essa versão também possibilita gerar códigos IDL (Interface Definition Language) para realizar a interface com CORBA (Common Object Request Broker Architecture), mapeando um modelo de objetos para IDL ou IDL para SDL.

Para a utilização desses recursos de OMT e orientação-objeto da linguagem SDL, existe um trabalho proposto que emprega a combinação OMT e SDL'92. Esse modelo é proposto no trabalho desenvolvido pela *Telelogic*[6] [56] [57] [58] [59] . Esse grupo de pesquisa propõe a metodologia SOMT (SDL-oriented Object Modelling Technique) [54] [55] que combina uma análise orientada-objeto e SDL para construir uma análise de forma coerente. Esse método aplica a análise orientada objeto com os recursos oferecidos pela versão SDT[7] 3.1 [57] [59] que possui novas características como editor de OMT, para auxiliar a documentação durante a especificação em SDL. Contudo, essa metodologia só é limitada as características da versão da 3.1 da ferramenta SDT, por apresentar características que só poderão ser realizadas com o *implink* e o *Paste as* fornecidas por essa versão.

A descrição do modelo criado, o SOMT, pode ser analisado pela figura B.1.

[5] Versão mais recente da ferramenta SDT, instalada na fase final do desenvolvimento.

[6] Telelogic Combitech Group - Fabricante do pacote SDT.

[7] SDT 3.1 - Nova versão da ferramenta SDT que diponibiliza recursos para construir o modelo de objetos para especificação SDL.

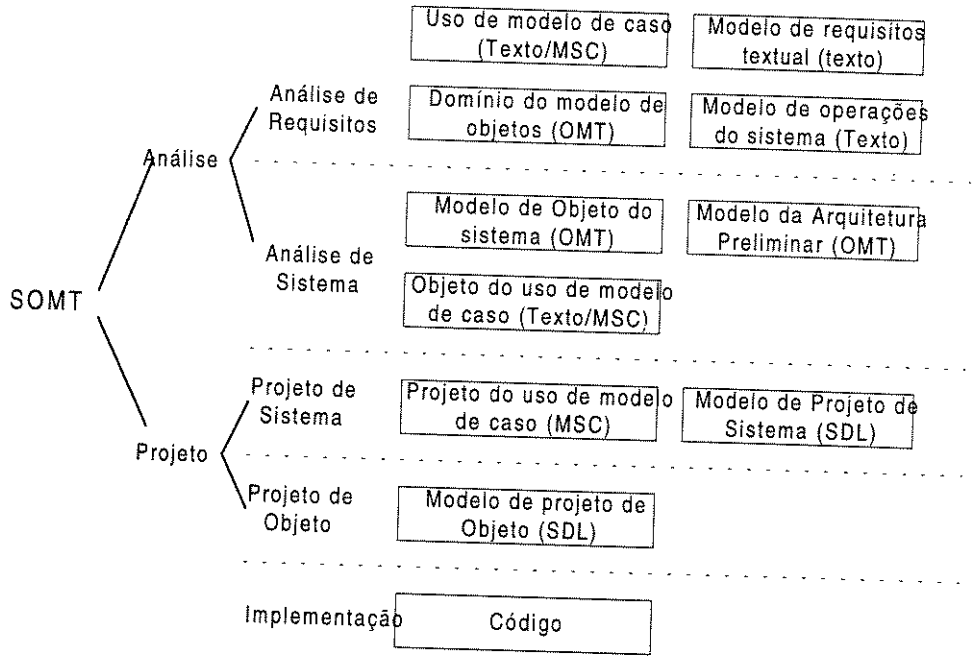

Figura B. 2 - Metodologia SOMT [54]

O modelo SOMT proposto, visto na figura B.1, consiste de 5 fases:

1. Análise de Requisitos, analisa o domínio do problema e os requisitos do sistema a ser construído.

2. Análise de Sistema, identifica os principais objetos que devem ser representados no sistema para alcançar a funcionalidade requerida e para ter uma idéia geral da arquitetura do sistema.

3. Projeto de Sistema, define a arquitetura do sistema e decide quais estratégias tomar para obter certas características como reuso e decomposição pacotes de trabalho para diferentes grupos de desenvolvedores.

4. Projeto de objetos, define em detalhes a funcionalidade, incluindo o comportamento dos objetos.

5. Implementação, cria a aplicação executável que implementa os requisitos.

A existência do modelo SOMT foi constatado mais tarde, após o começo desse trabalho. Assim, não sendo adotado para o planejamento dos sistemas. Além disso, muitas das

características propostas pelo modelo SOMT, estão ligadas diretamento a ferramenta SDT 3.1, por apresentar as características de *implink* e *paste as*, assim, limitando o modelo a ser adotado ao OMT e a ferramenta, sendo que a metodologia utilizada neste trabalho, não limita a metodologia adotada, nem uma ferramenta específica para a especificação formal de protocolos de comunicação.

Apêndice C

Especificações em SDL

C.1 - Sistema Cloud

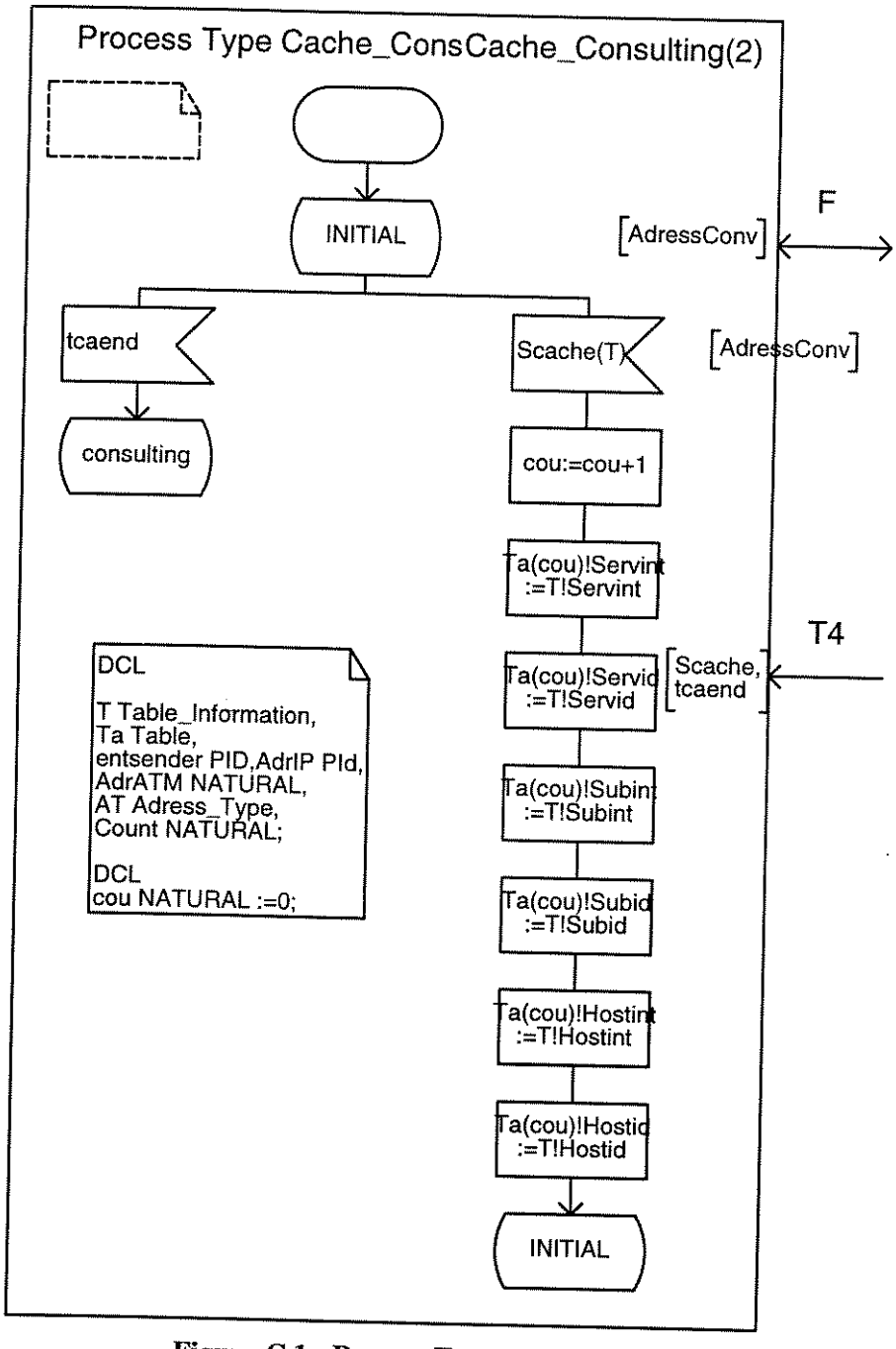

Figura C.1 - Process Type Cache_Consulting (1)

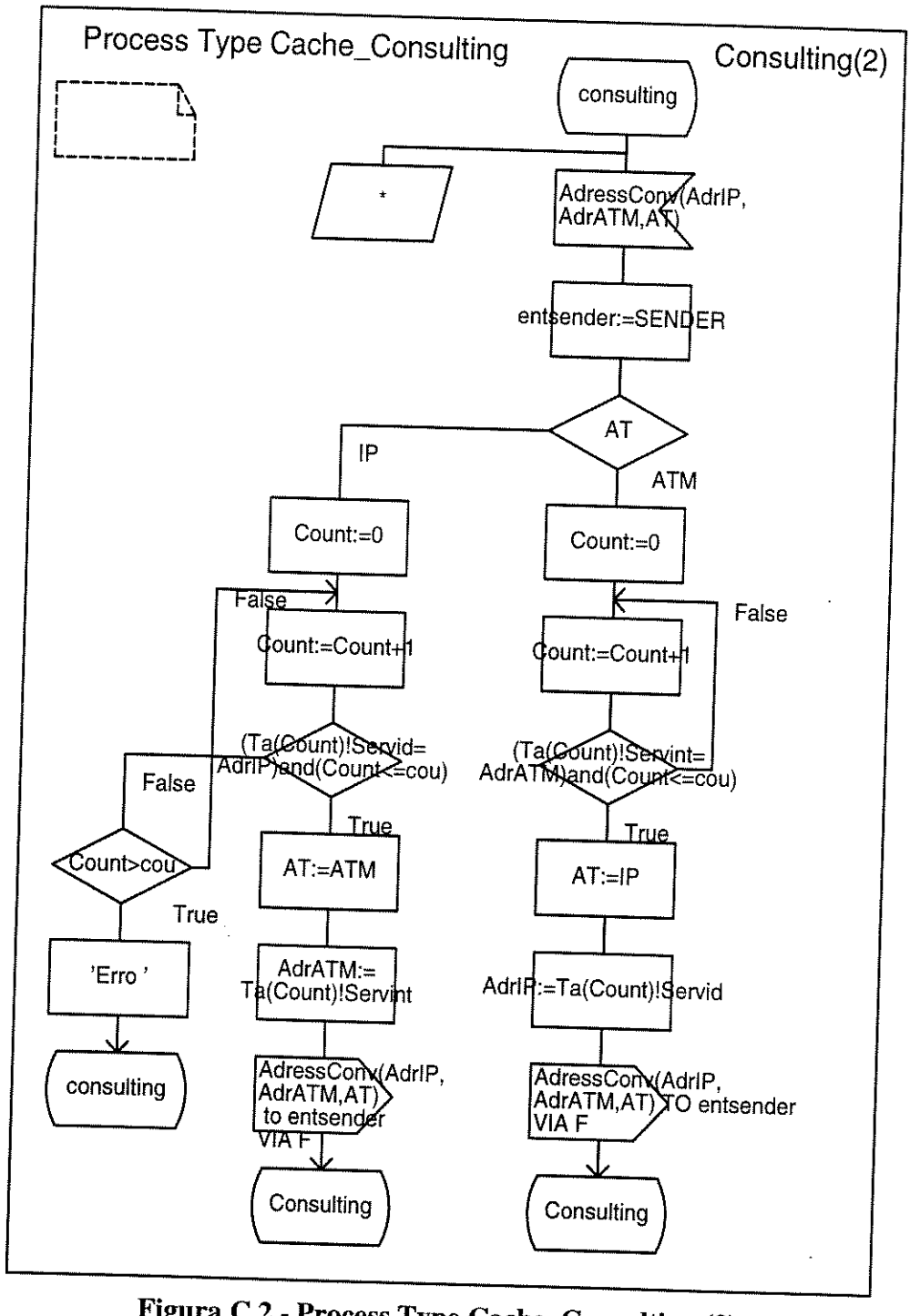

Figura C.2 - Process Type Cache_Consulting (2)

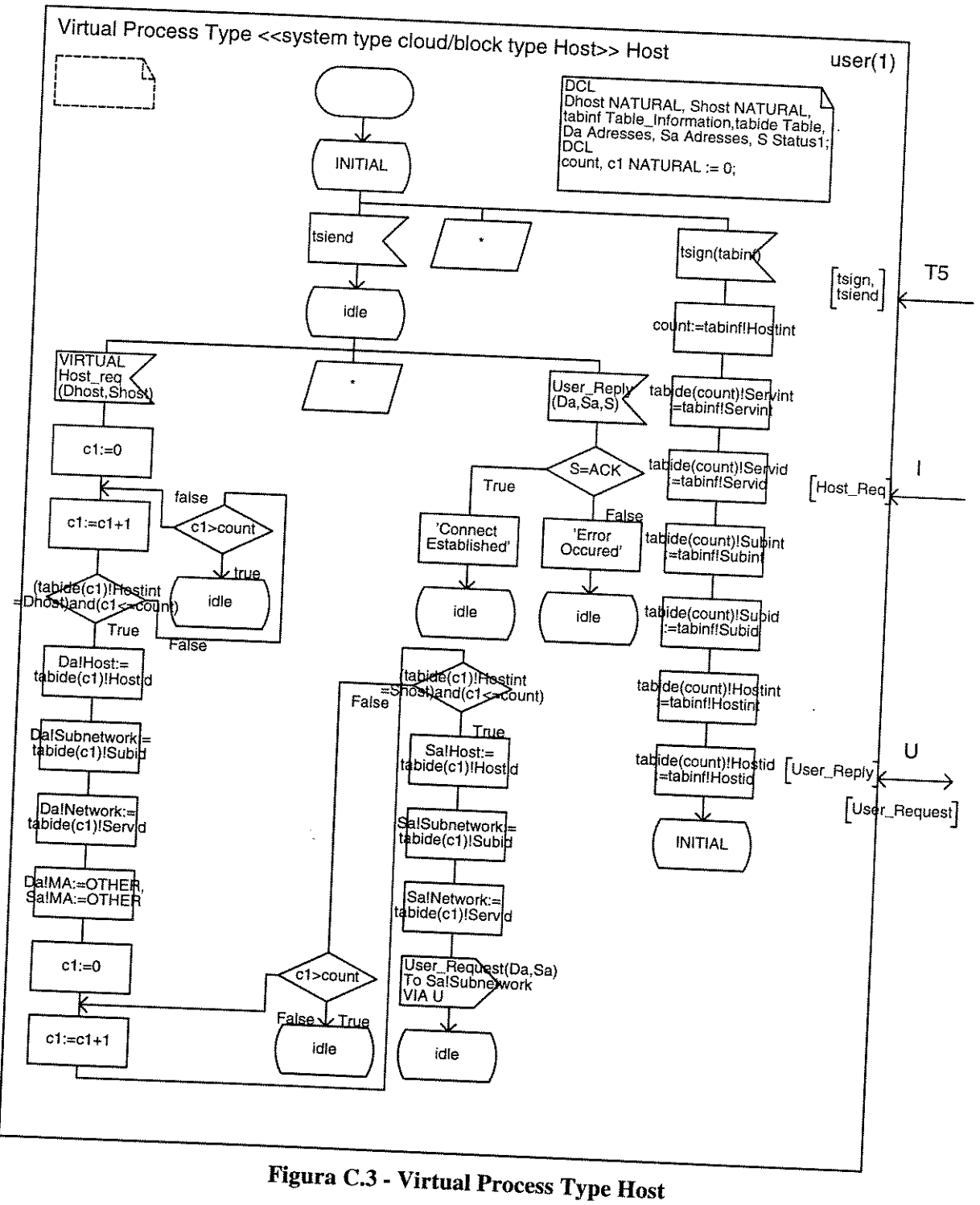

Figura C.3 - Virtual Process Type Host

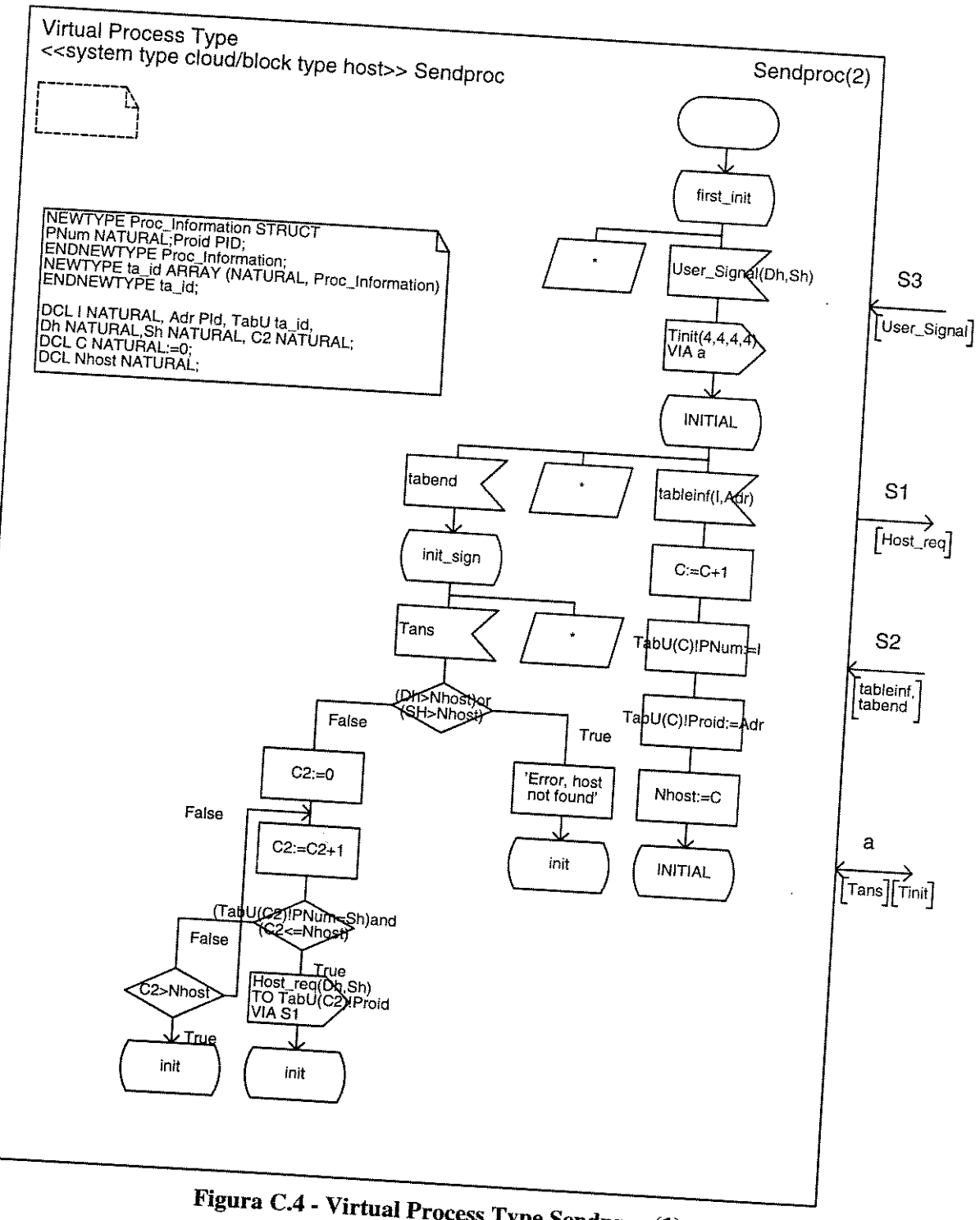

Figura C.4 - Virtual Process Type Sendproc (1)

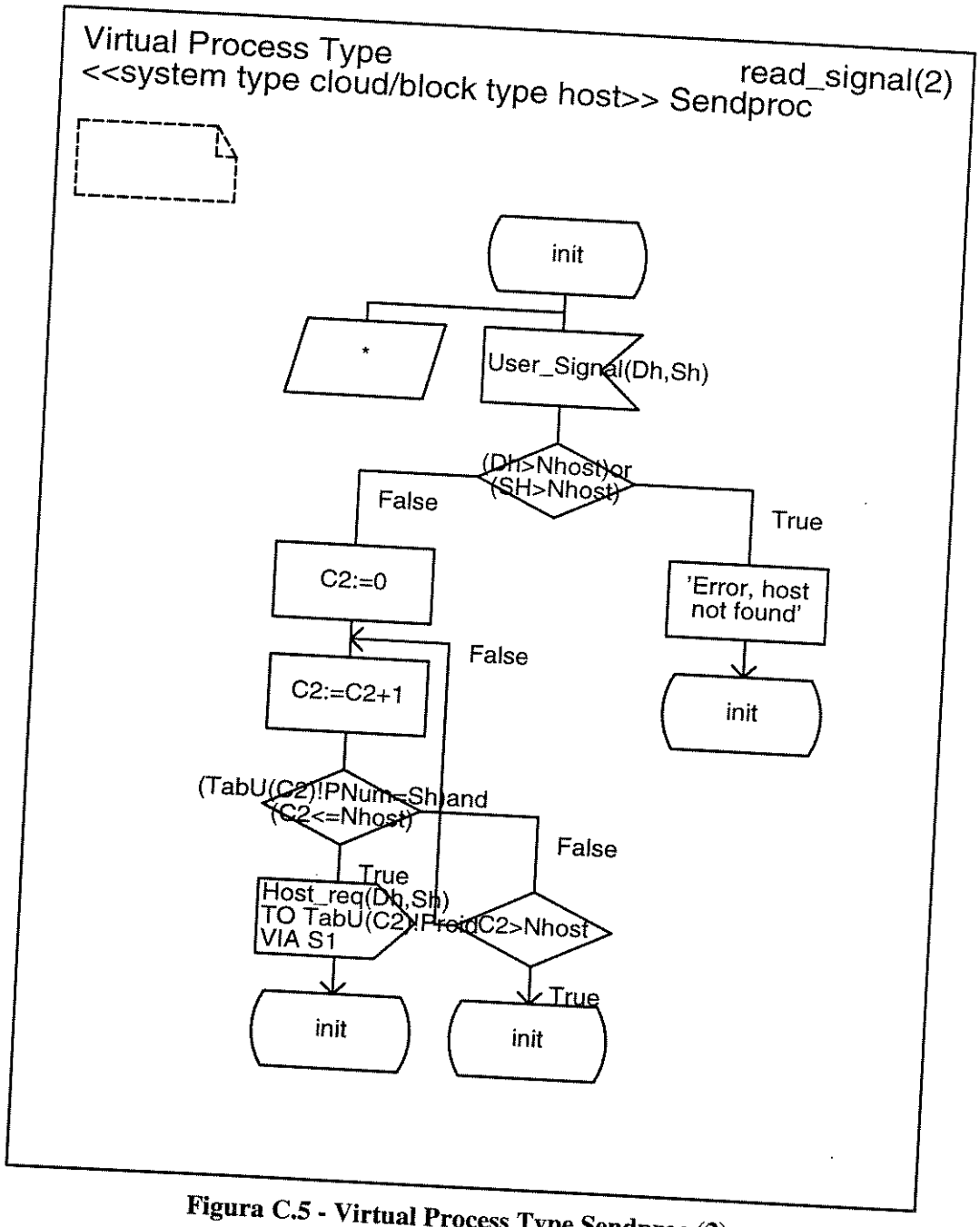

Figura C.5 - Virtual Process Type Sendproc (2)

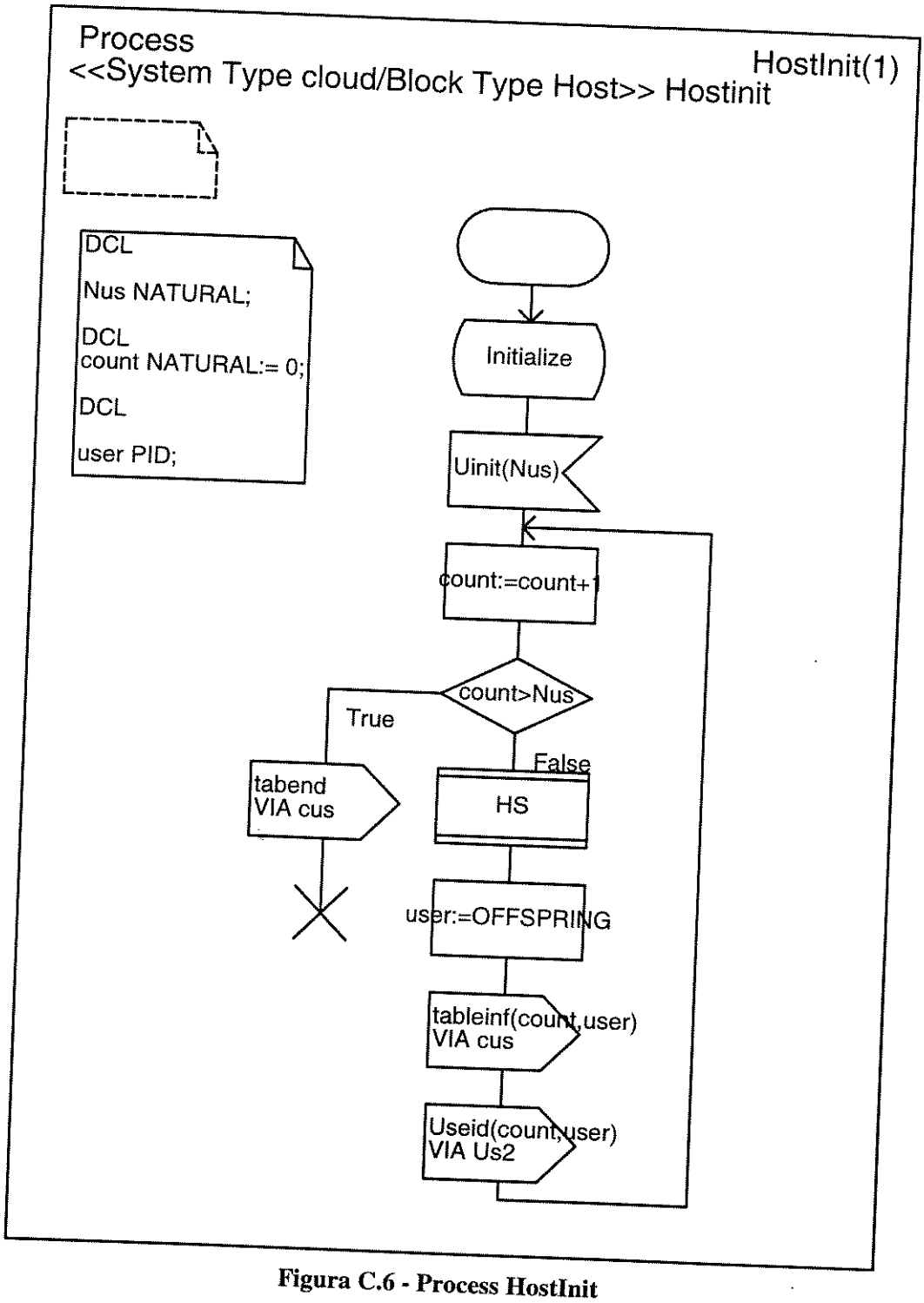

Figura C.6 - Process HostInit

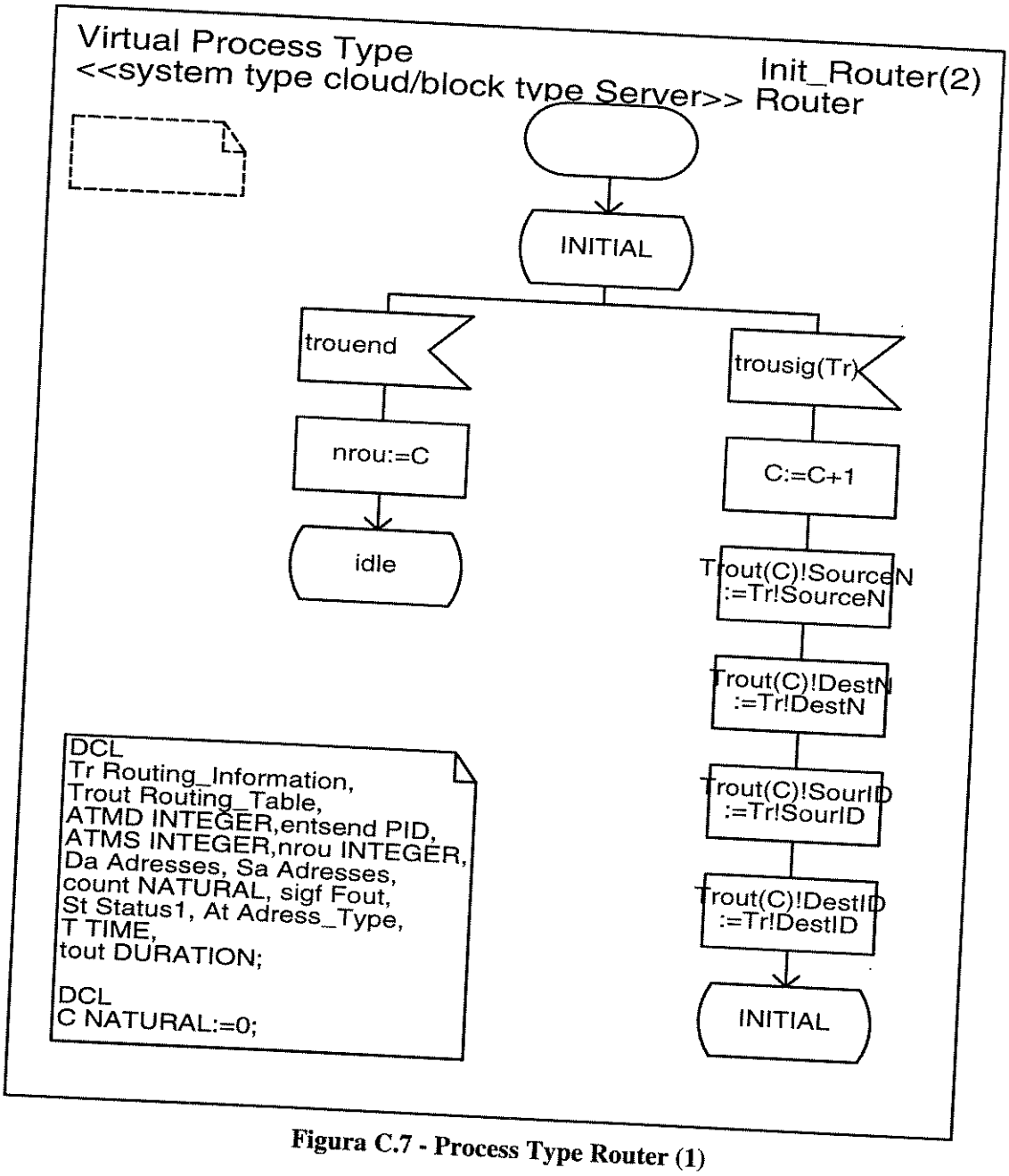

Figura C.7 - Process Type Router (1)

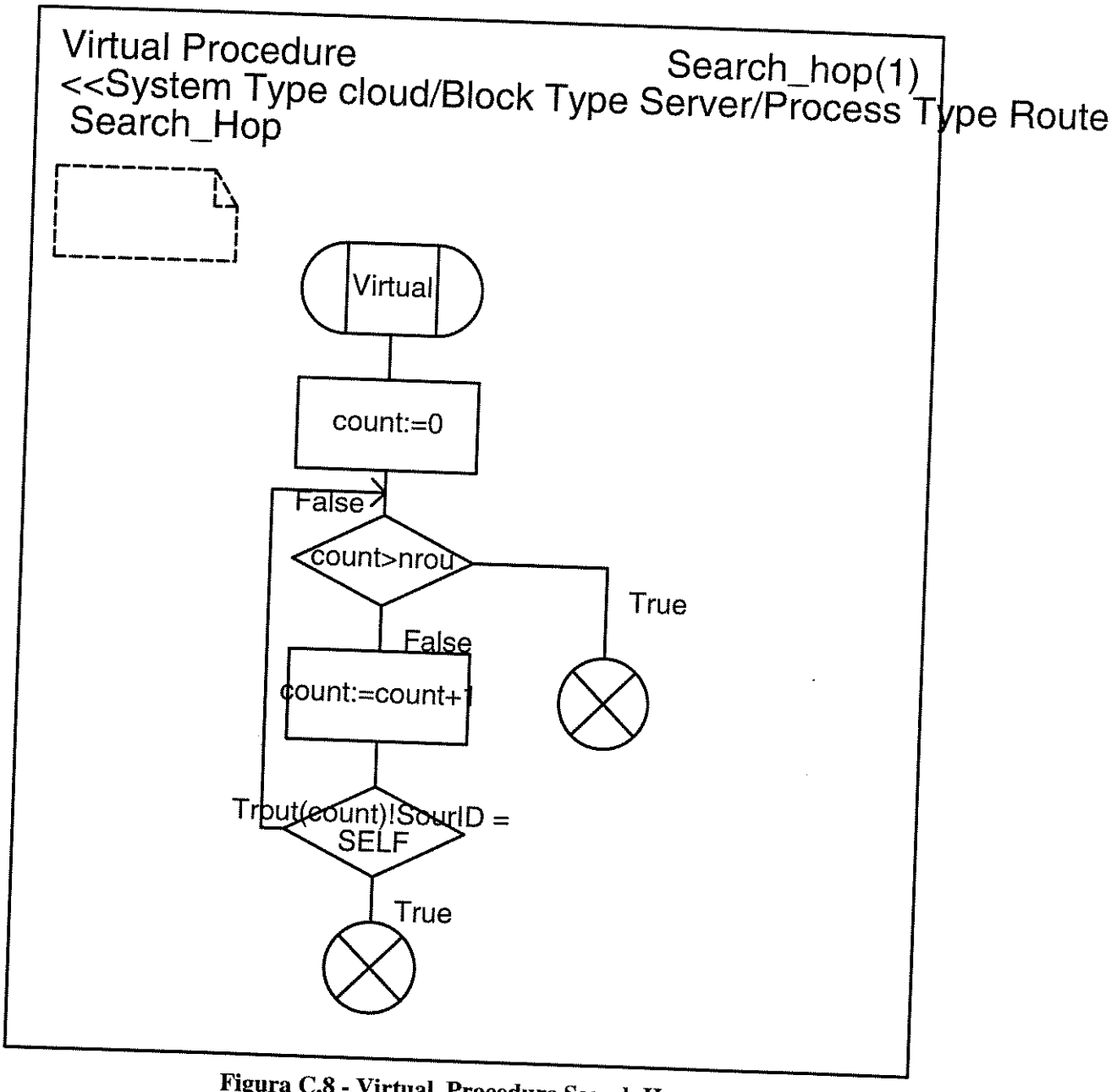

Figura C.8 - Virtual Procedure Search Hop

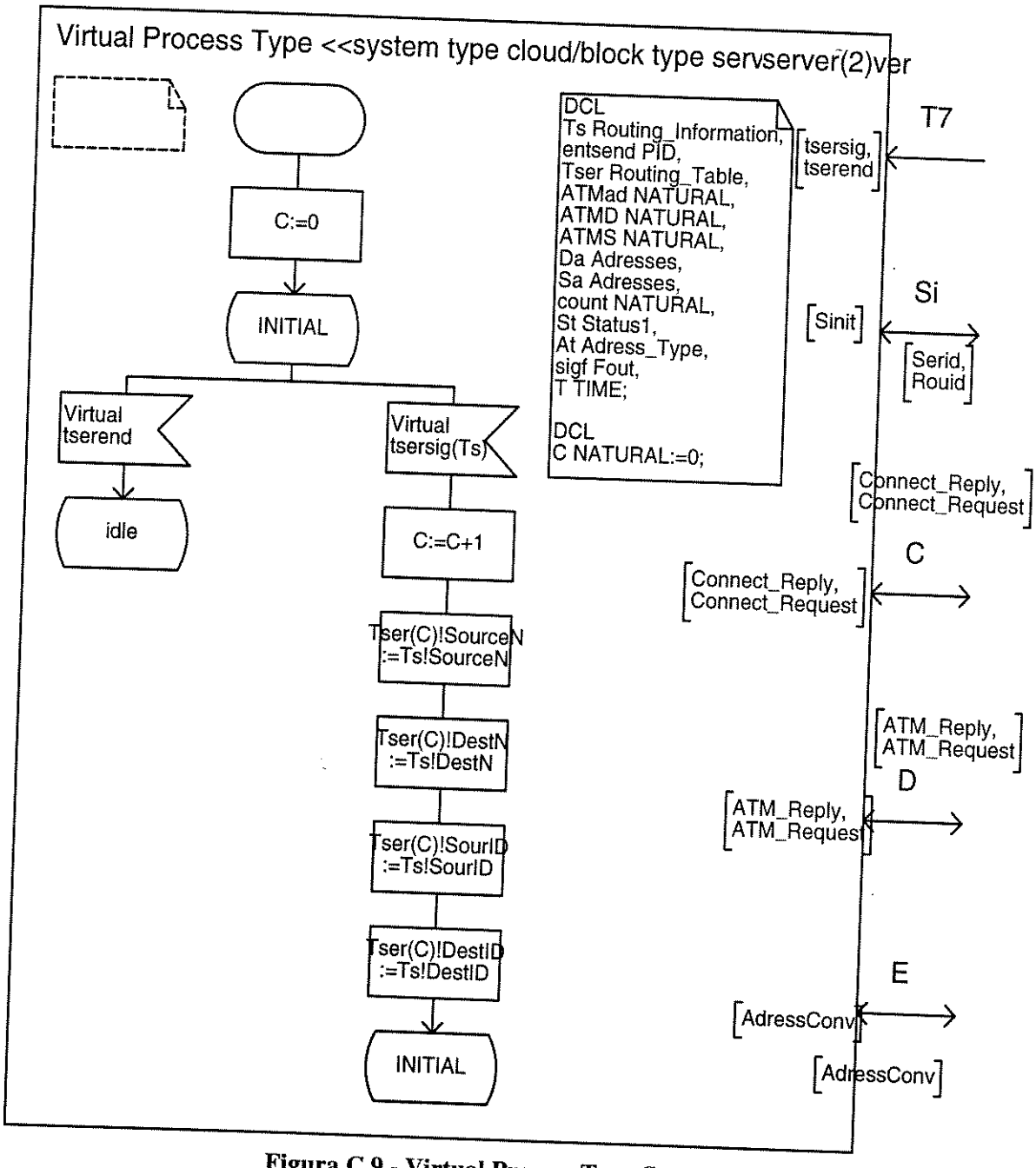

Figura C.9 - Virtual Process Type Server (1)

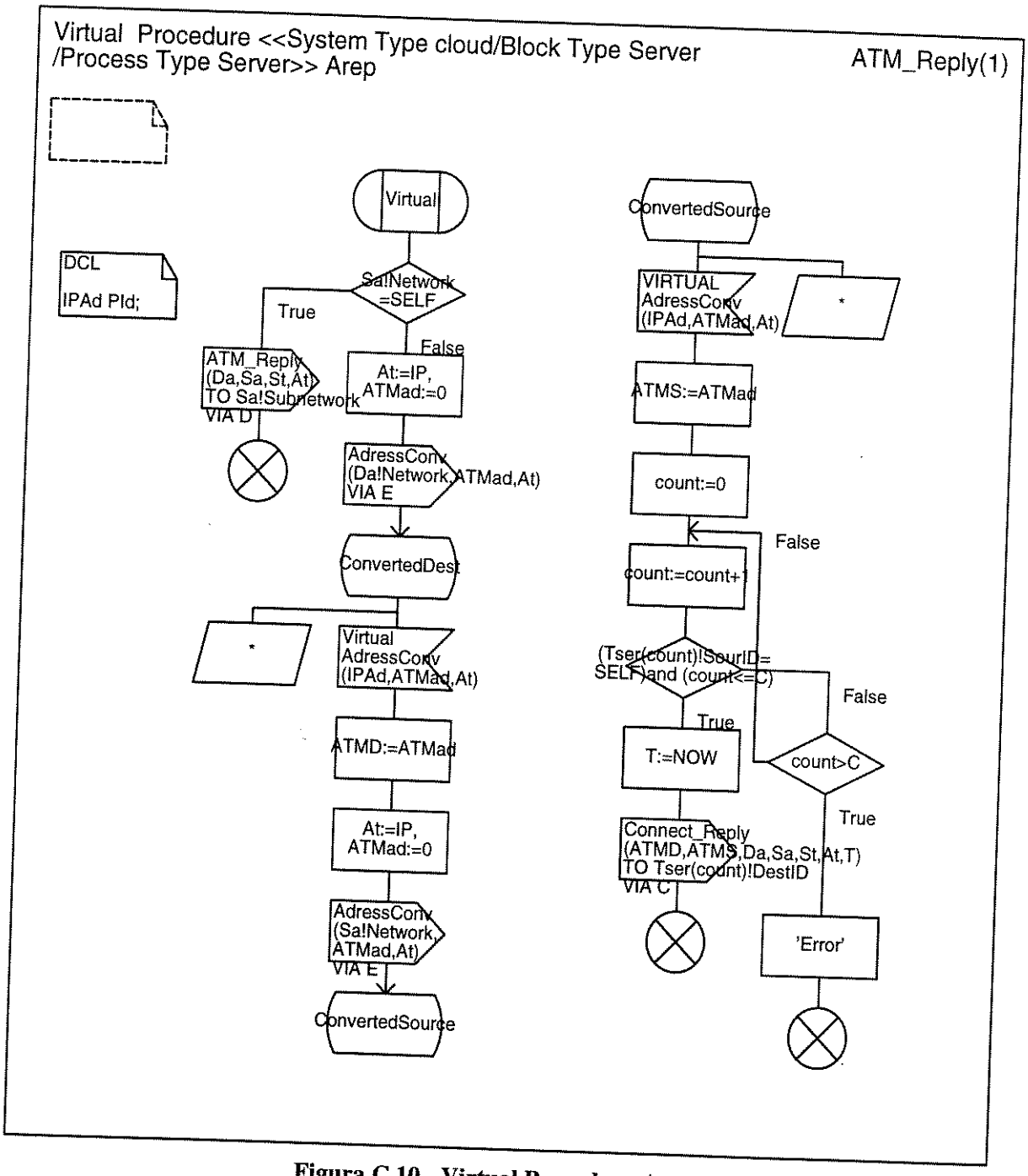

Figura C.10 - Virtual Procedure Arep

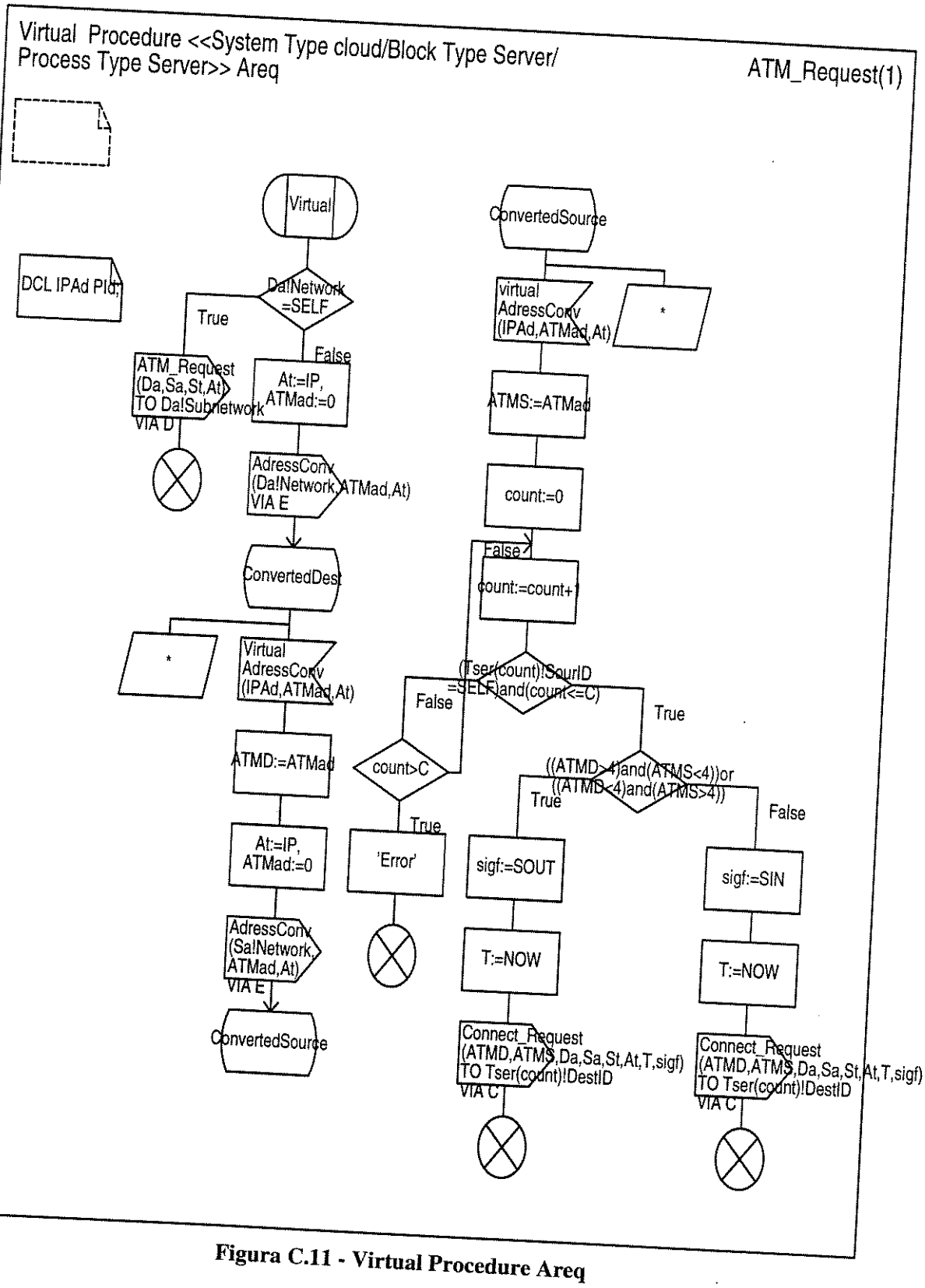

Figura C.11 - Virtual Procedure Areq

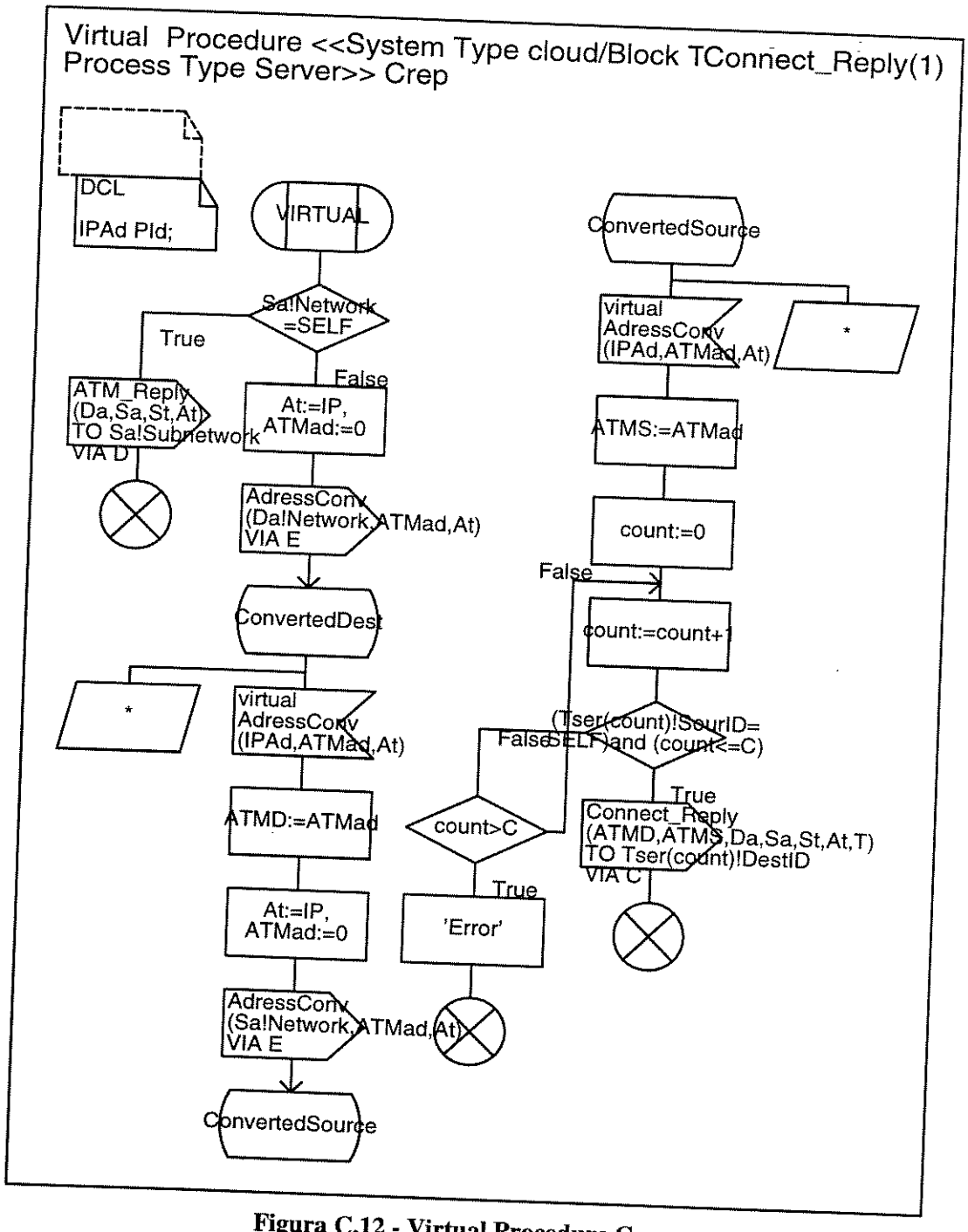

Figura C.12 - Virtual Procedure Crep

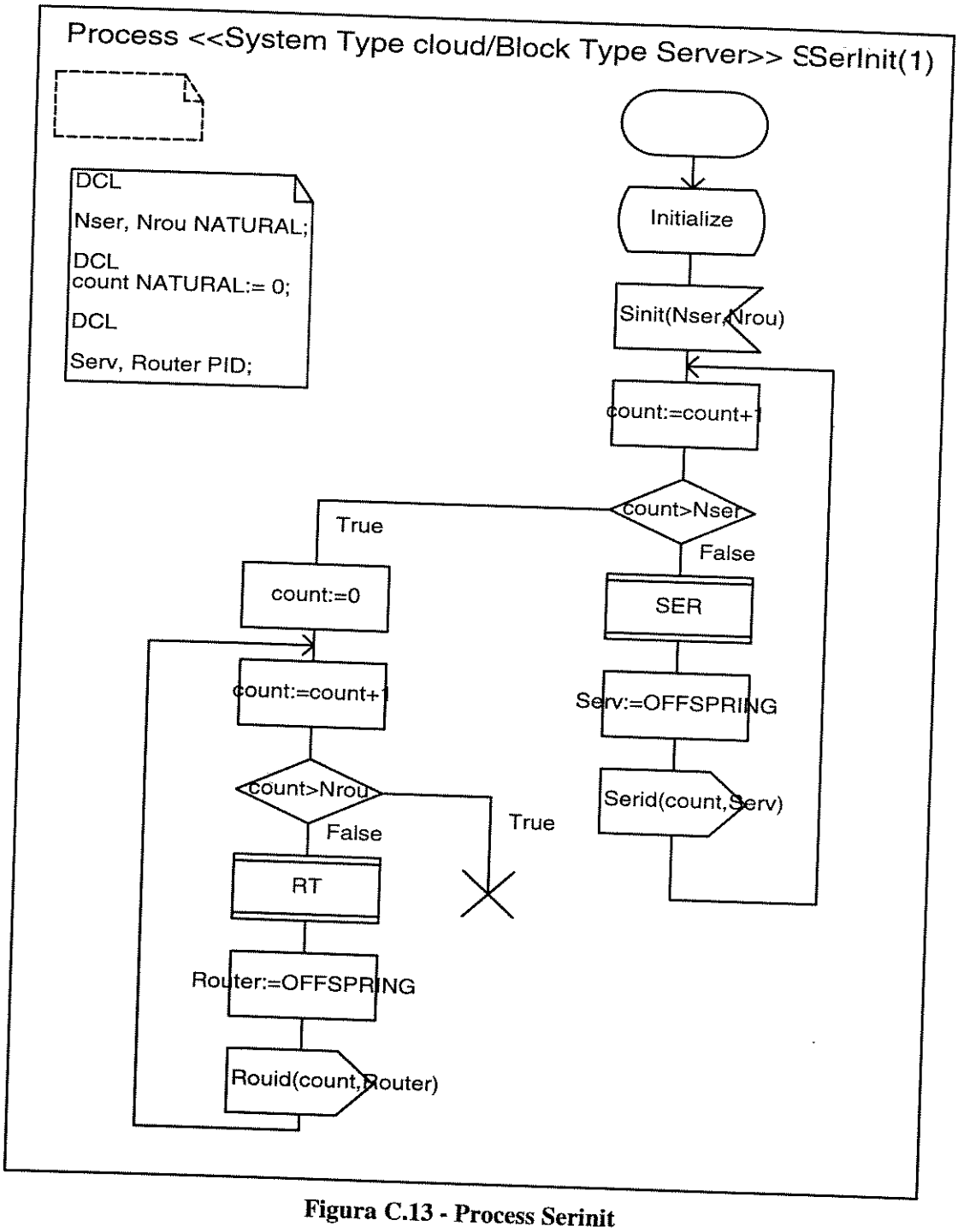

Figura C.13 - Process Serinit

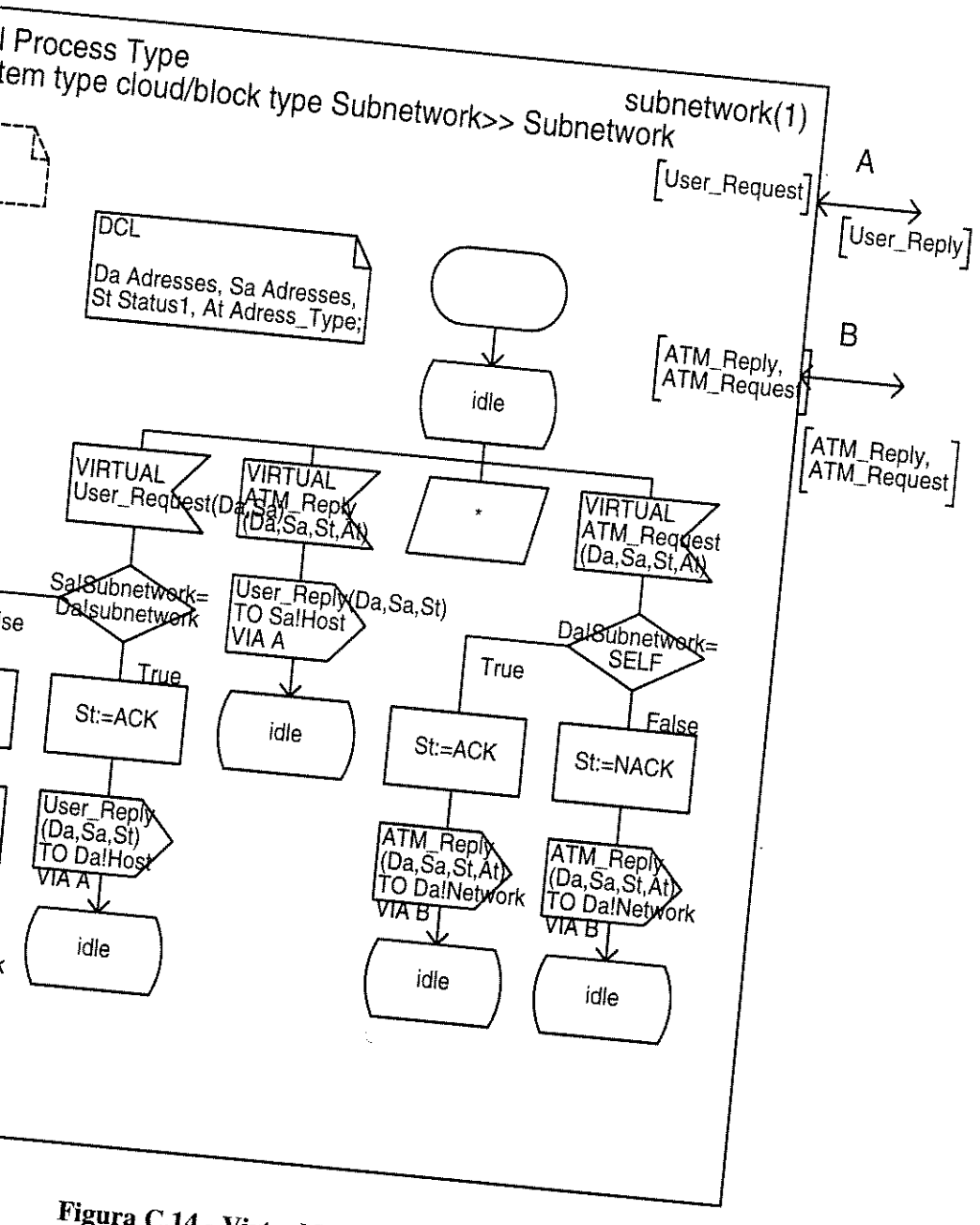

Figura C.14 - Virtual Process Tyep Subnetwork

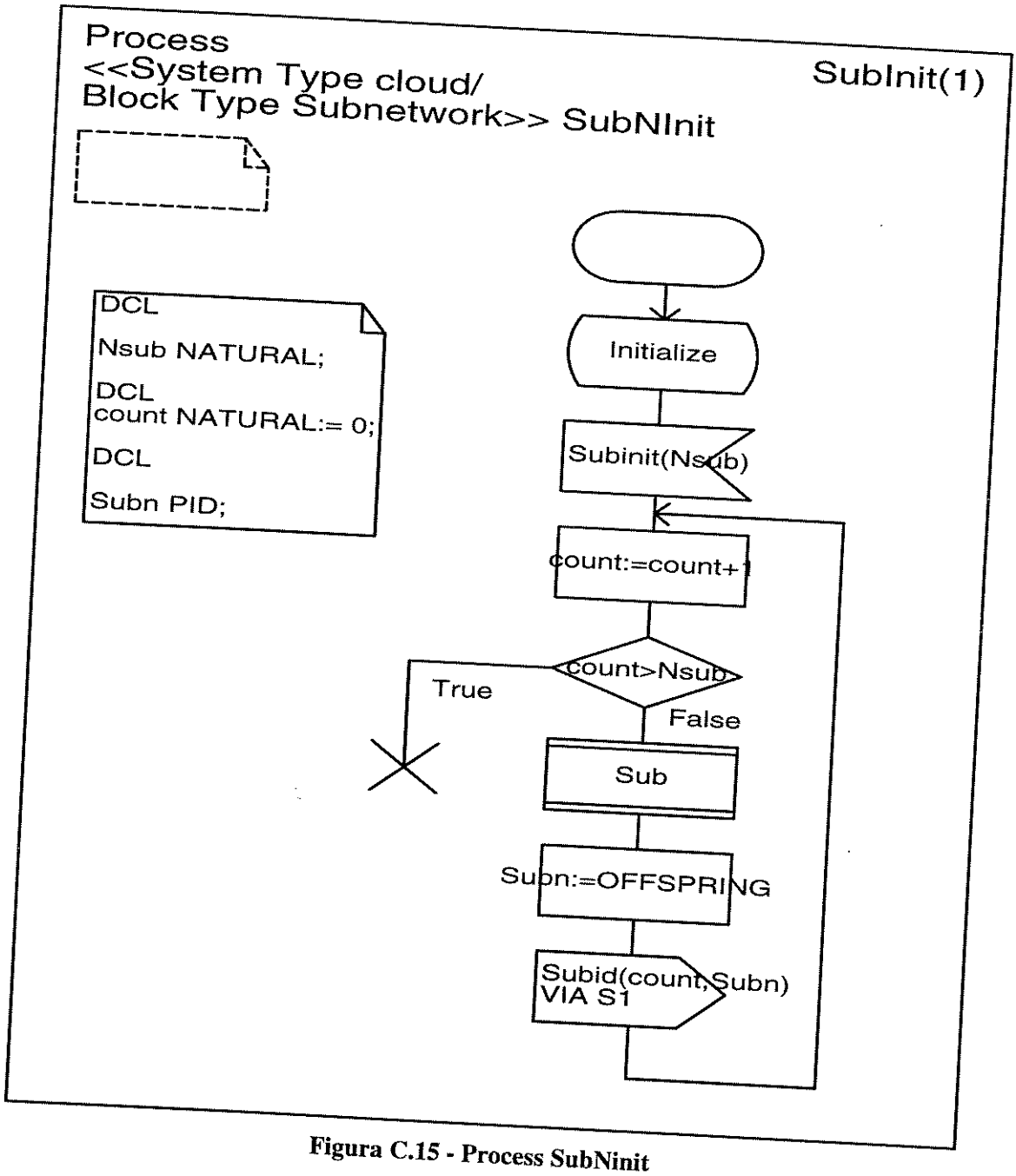

Figura C.15 - Process SubNinit

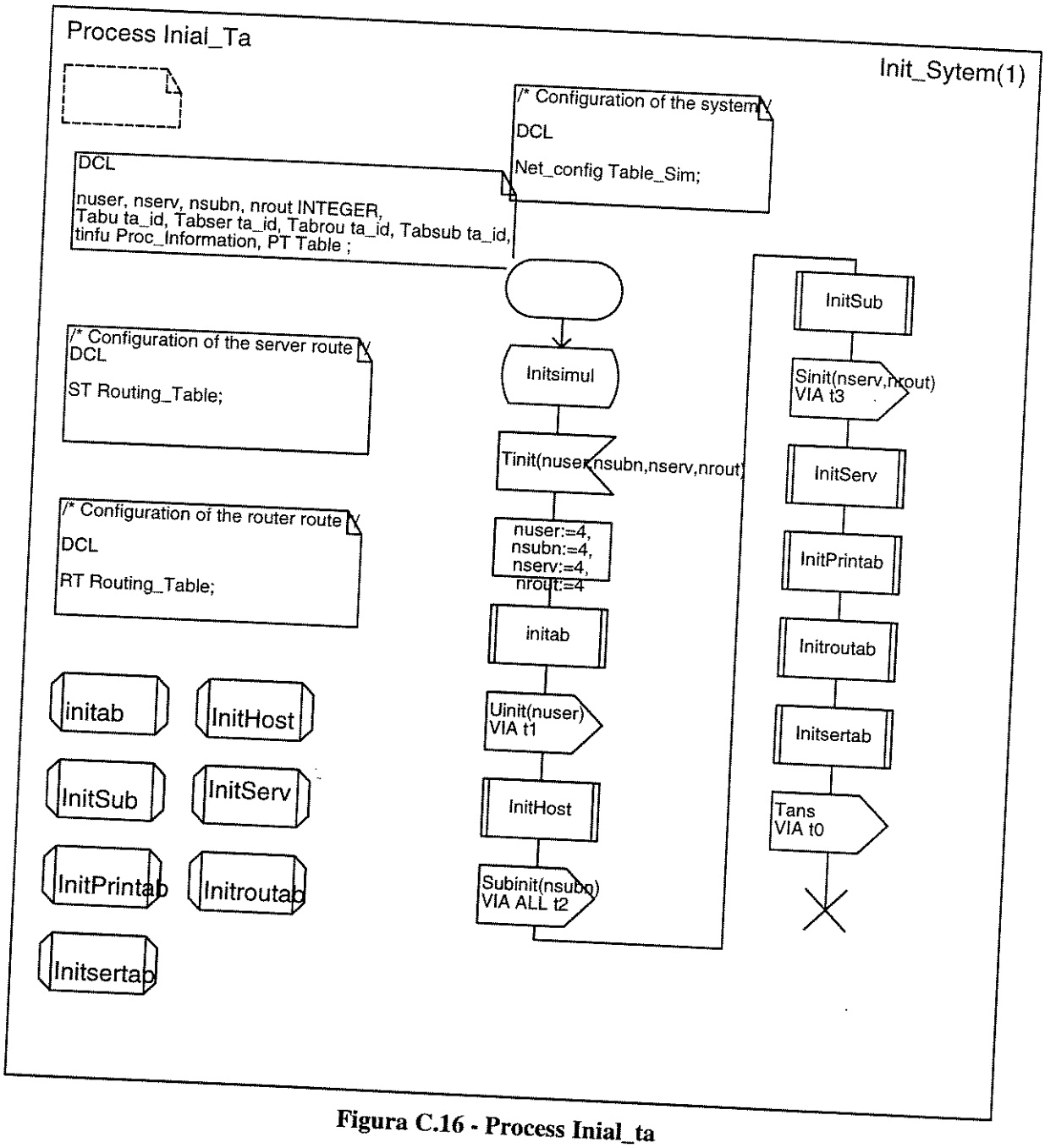

Figura C.16 - Process Inial_ta

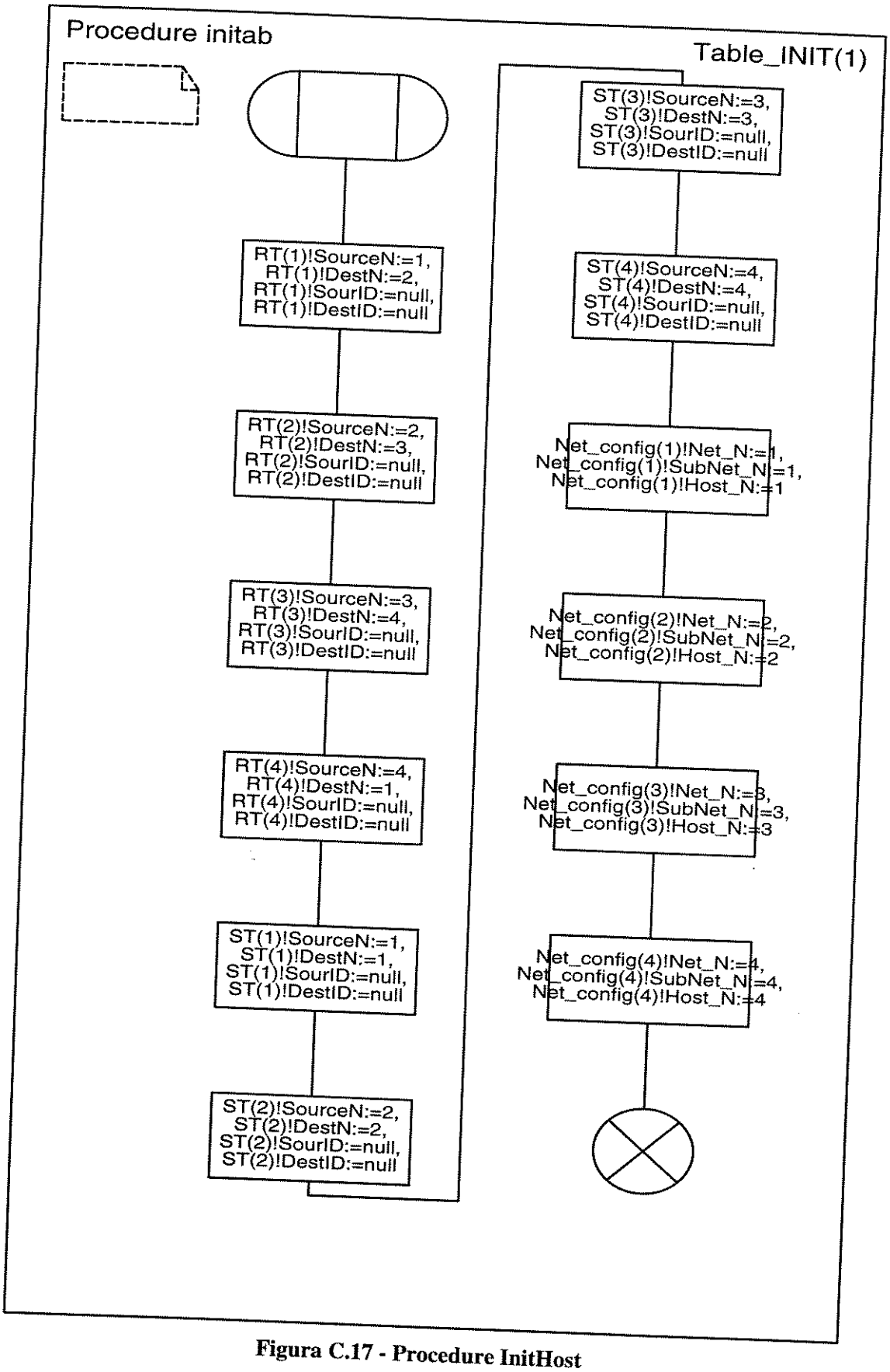

Figura C.17 - Procedure InitHost

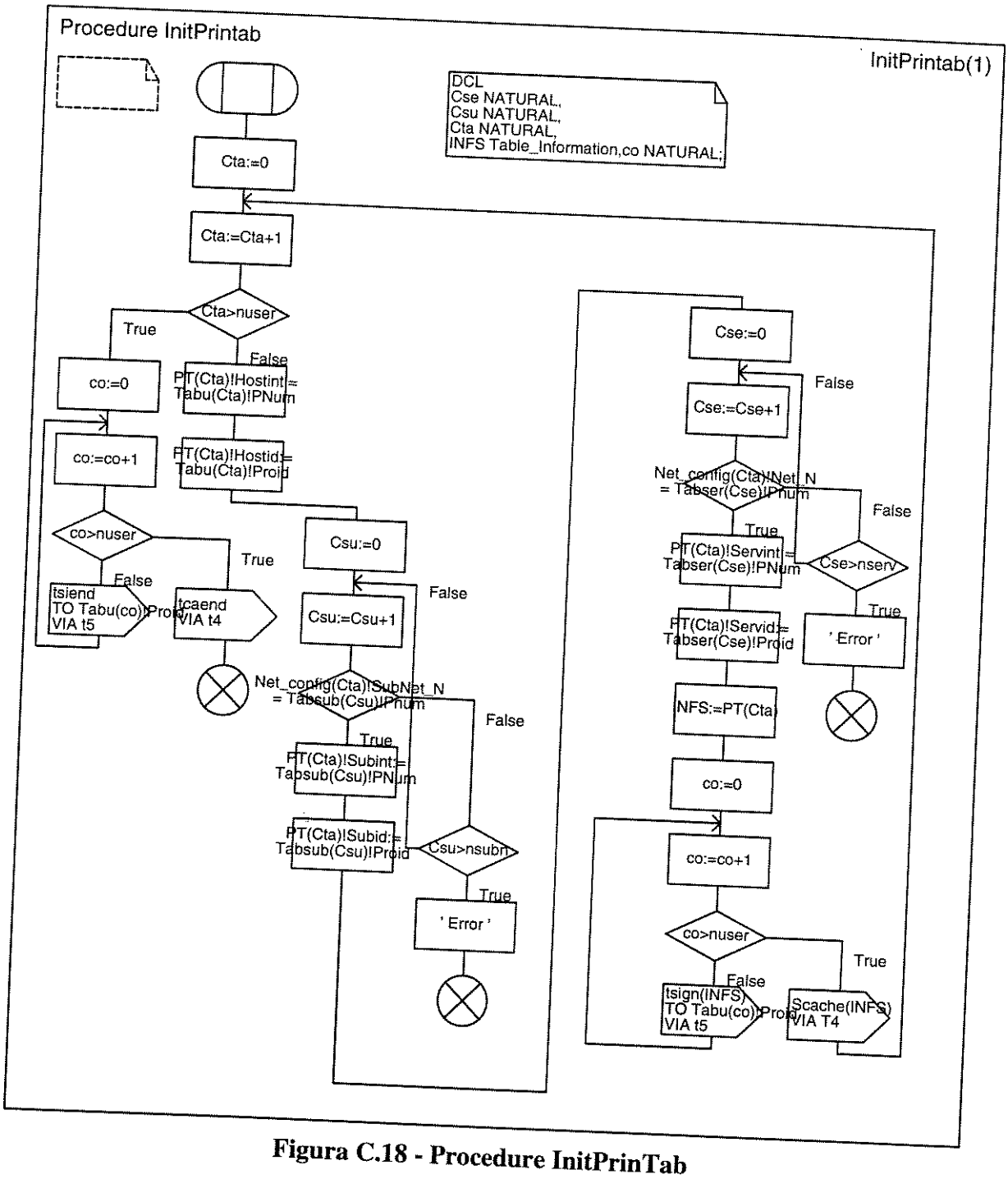

Figura C.18 - Procedure InitPrinTab

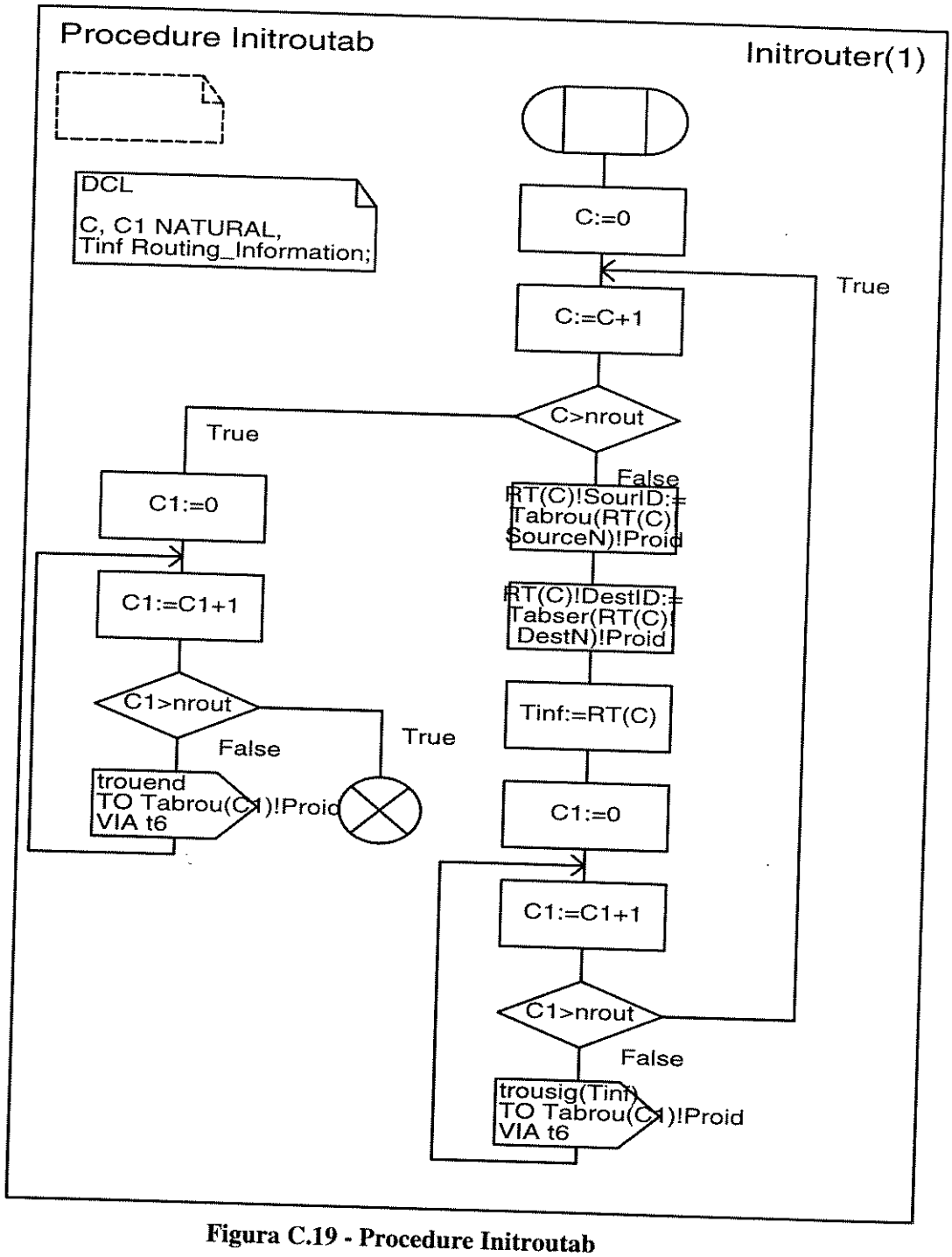

Figura C.19 - Procedure Initroutab

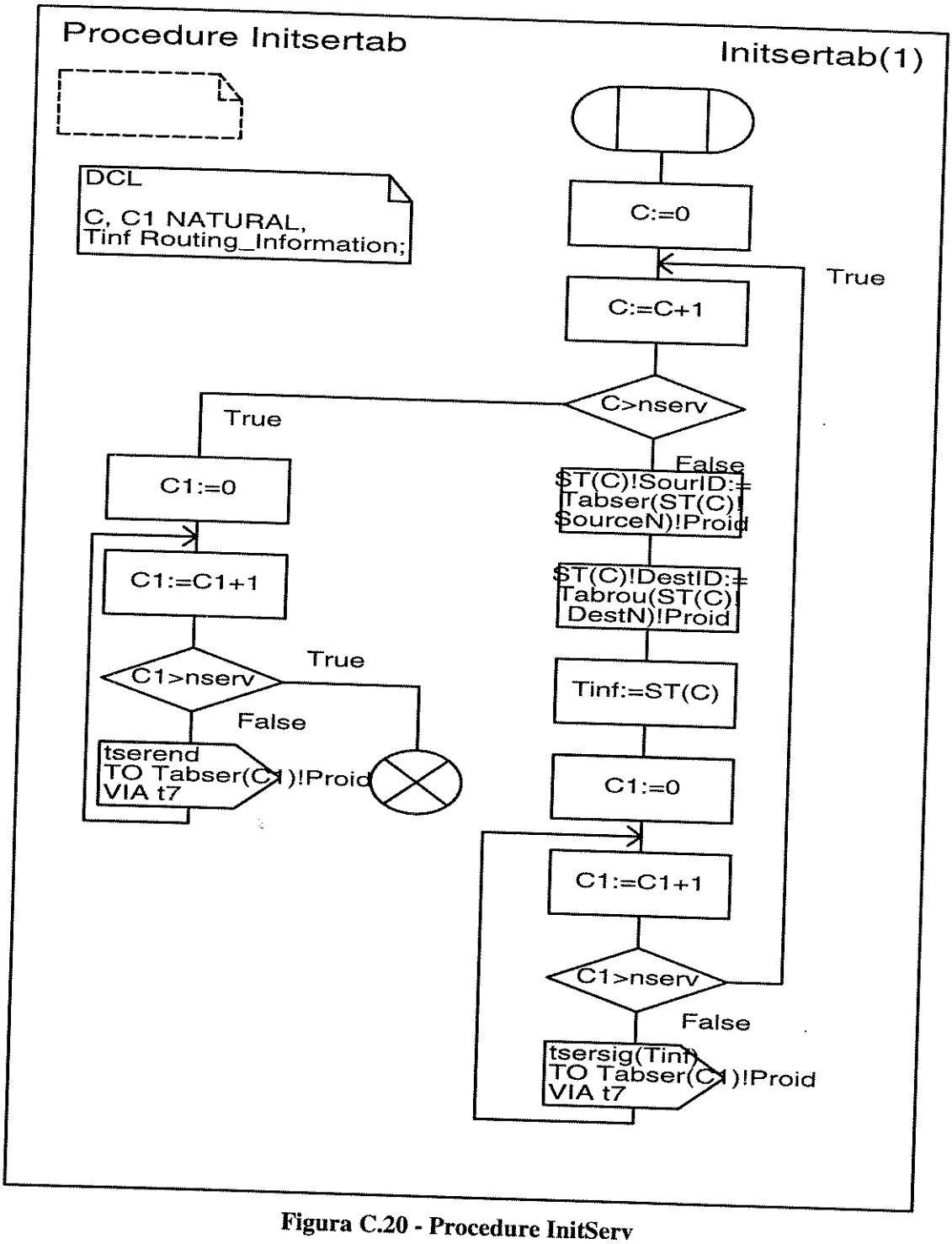

Figura C.20 - Procedure InitServ

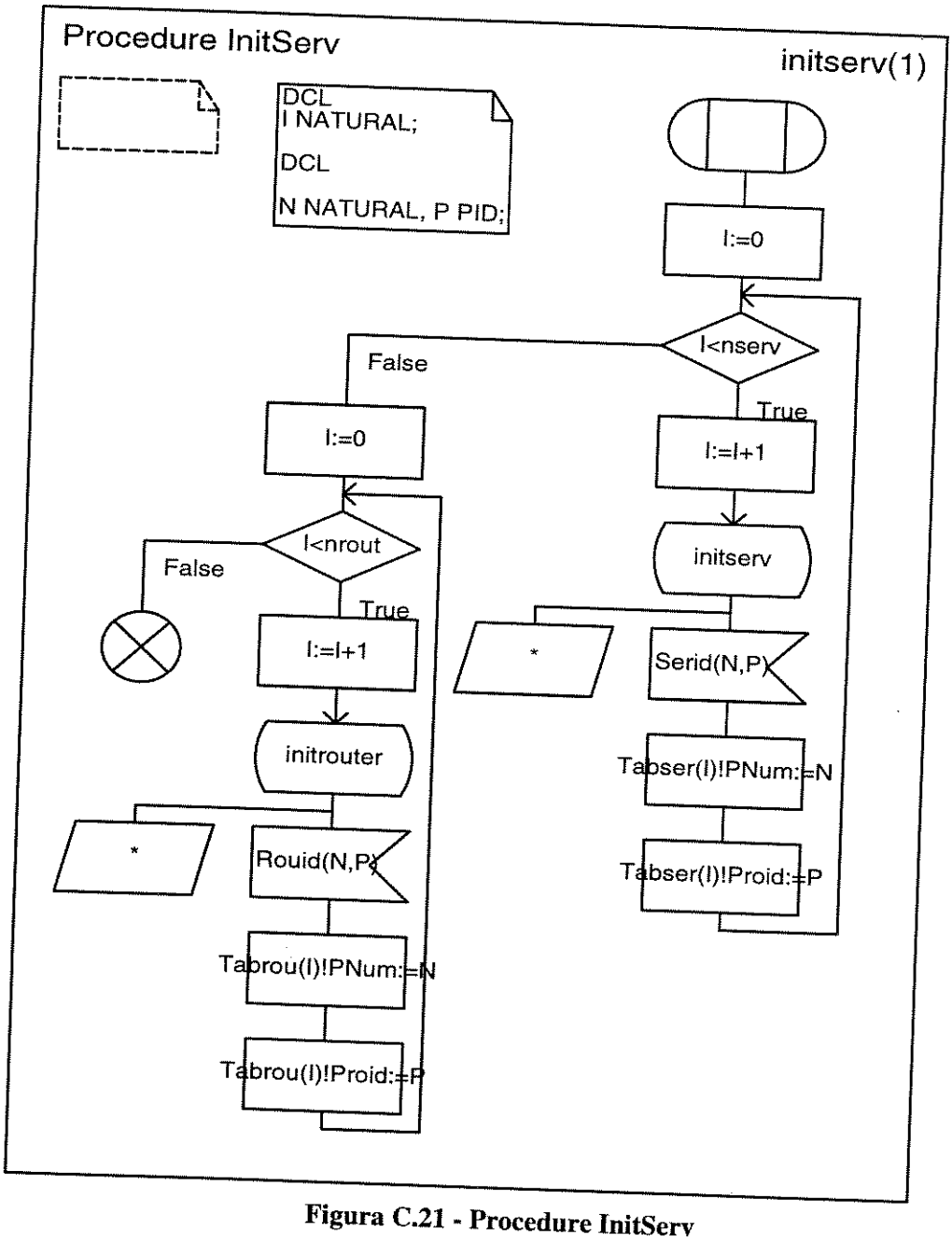

Figura C.21 - Procedure InitServ

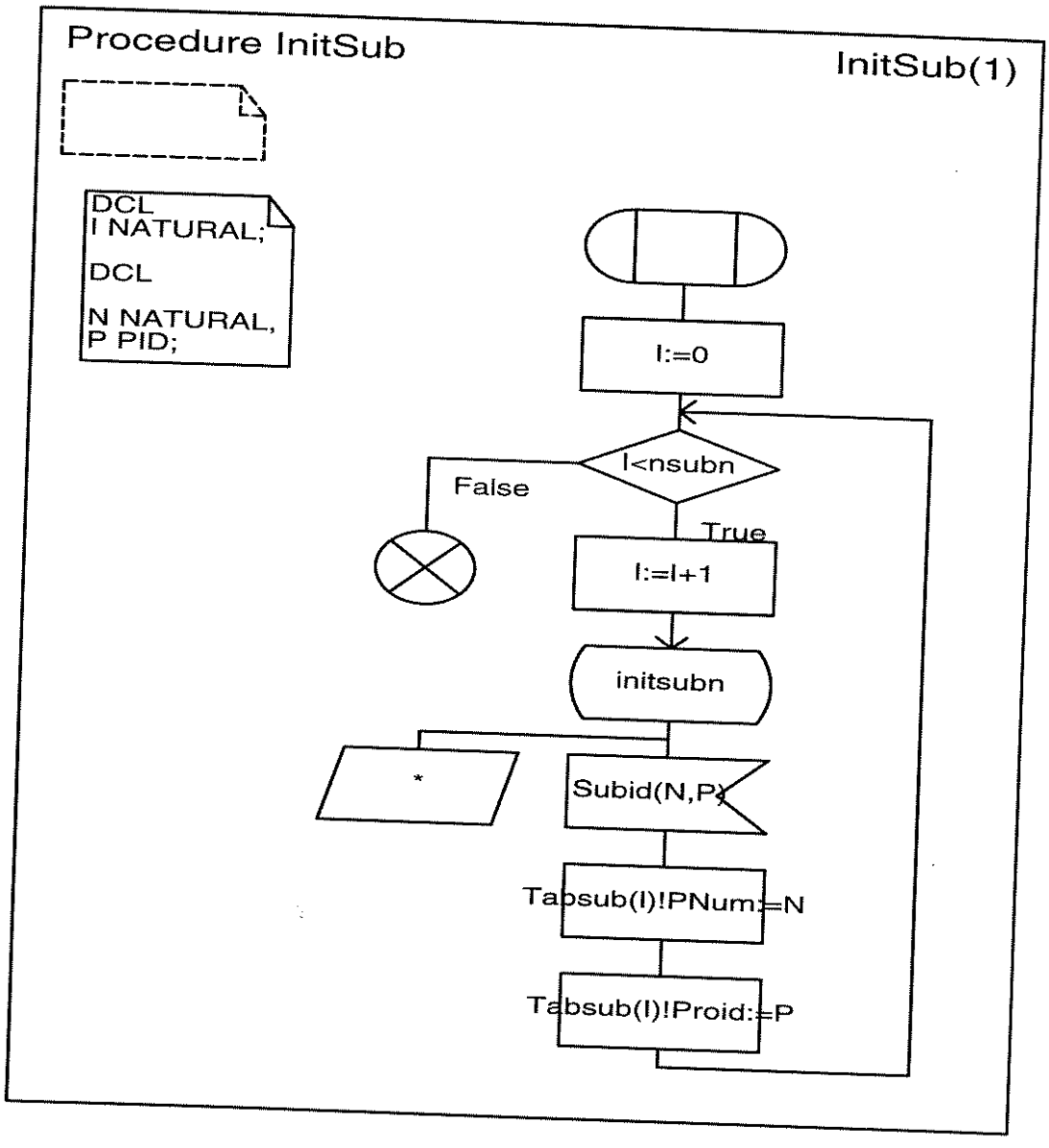

Figura C.22 - Procedure Init Sub

C.2 - Sistema IP Clássico

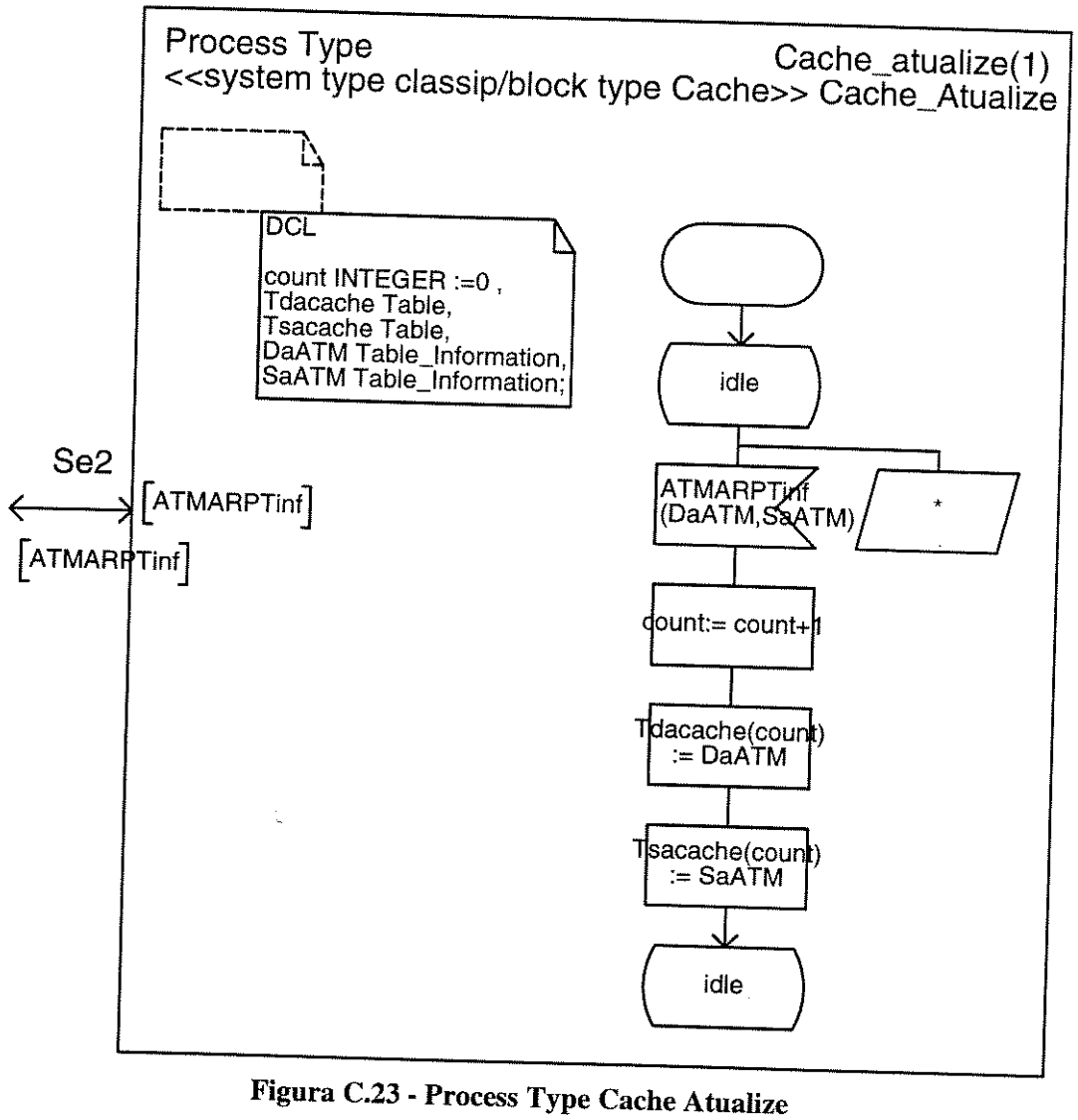

Figura C.23 - Process Type Cache Atualize

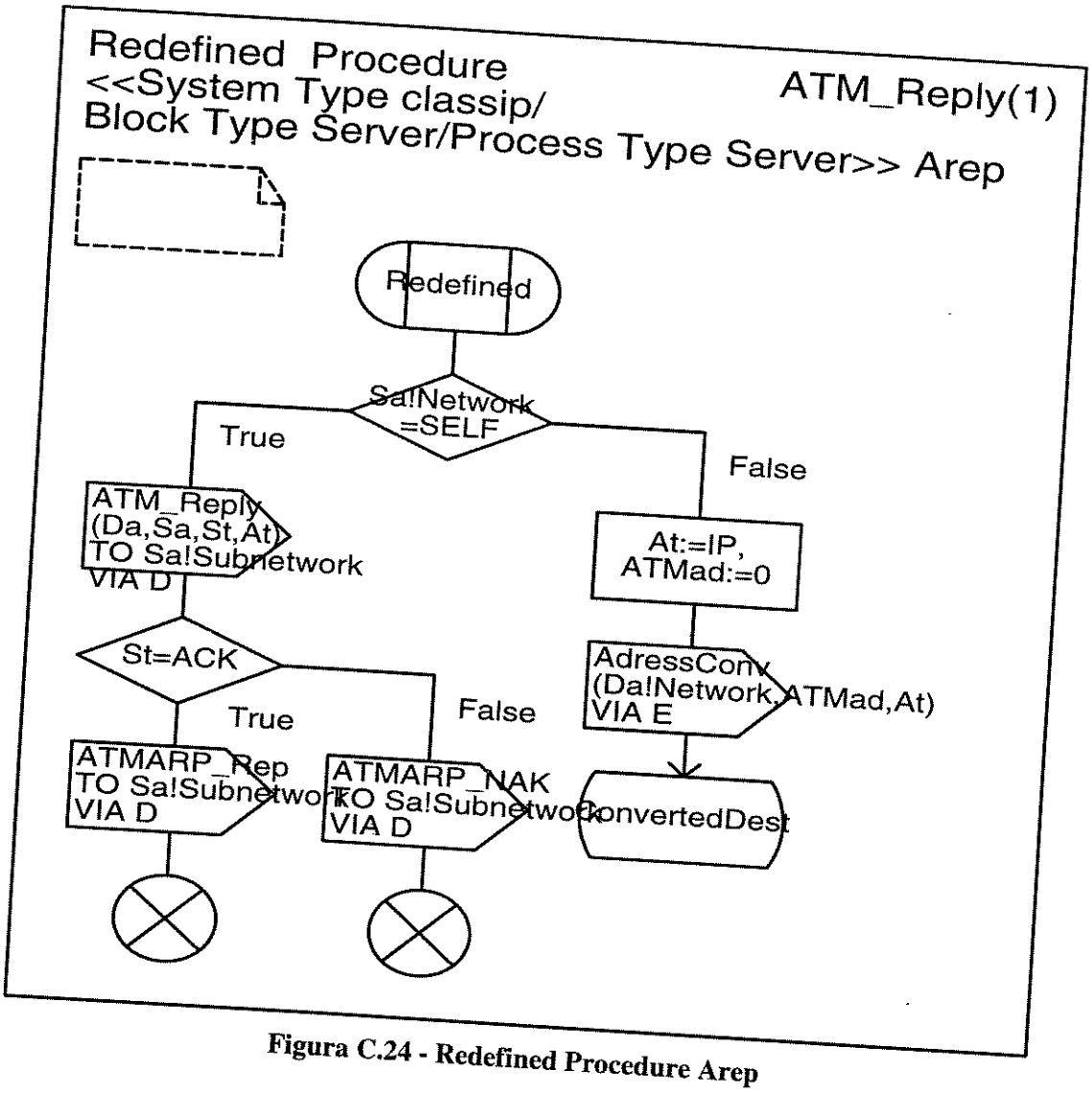

Figura C.24 - Redefined Procedure Arep

C.3 - Sistema NHRP

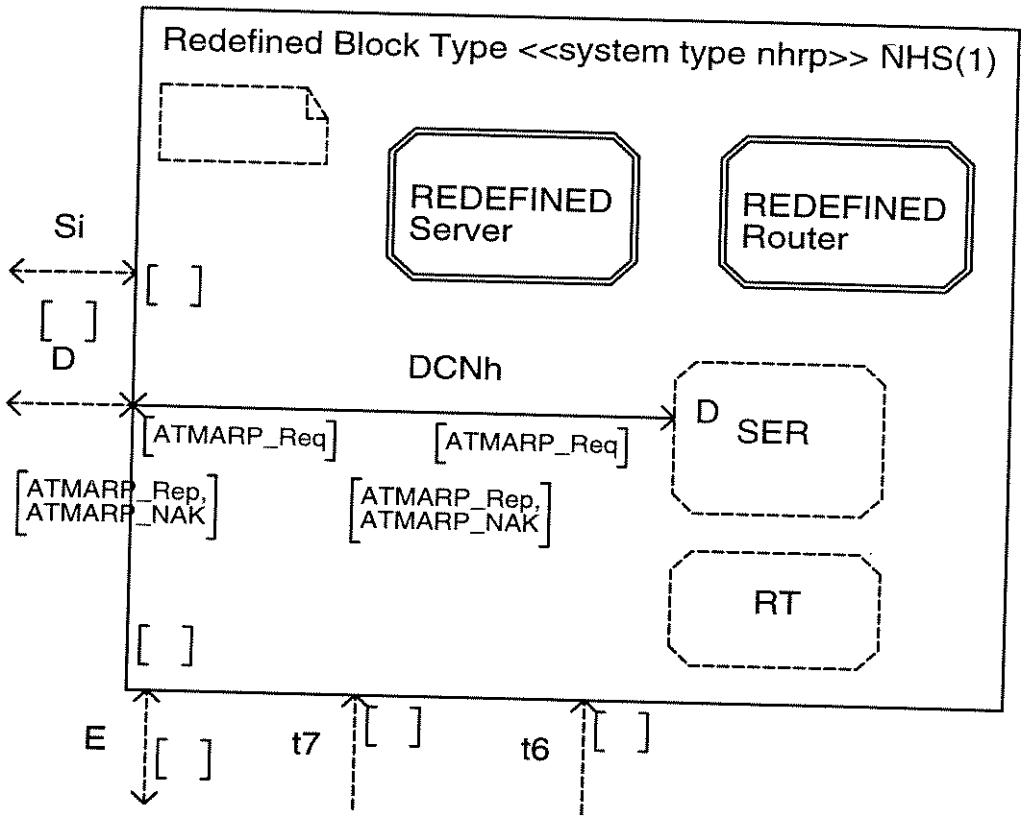

Figura C.25 - Redefined Process Type Router

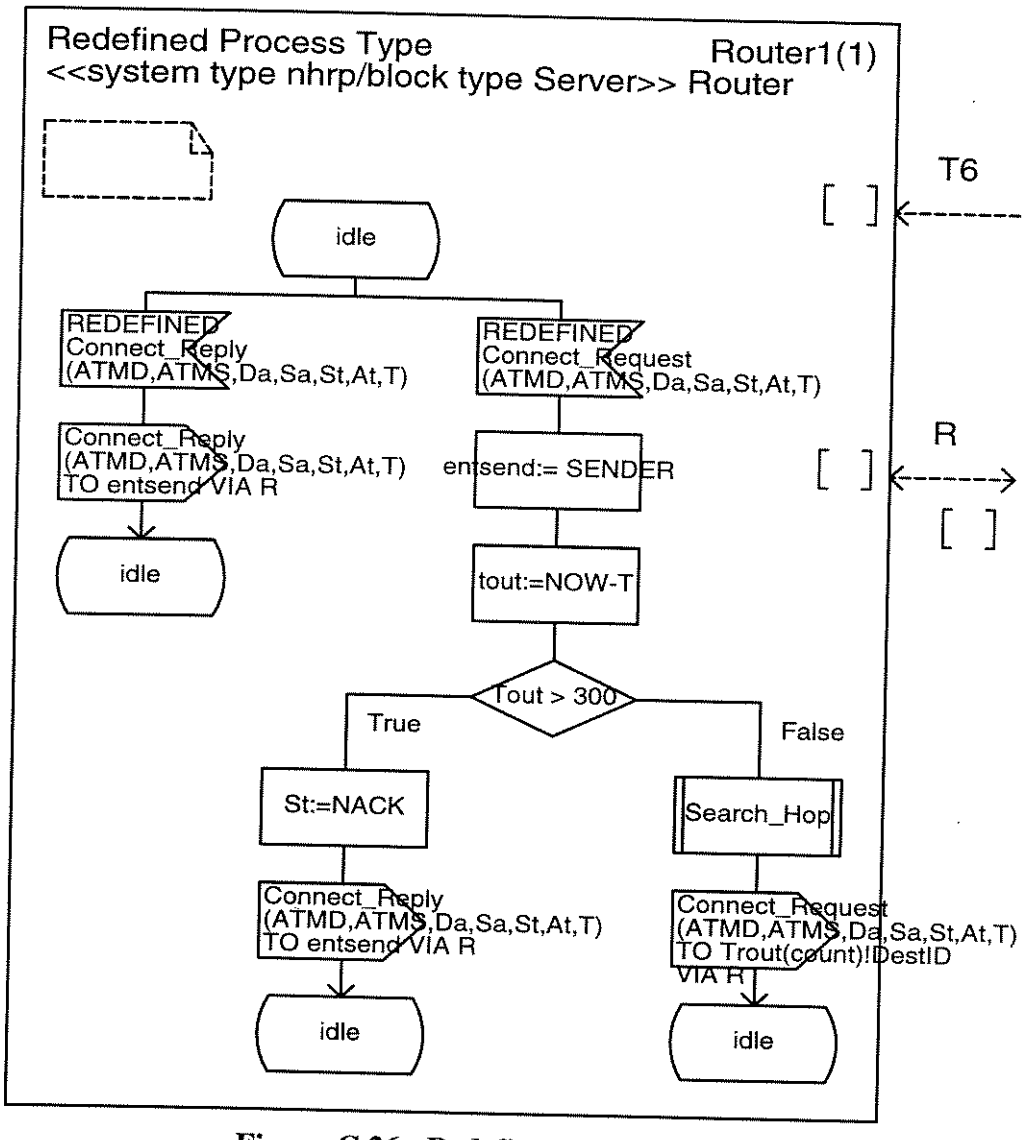

Figura C.26 - Redefined Process Type Server

C.4 - Sistema NHRP que se comunica através do protocolo Proxy-ARP

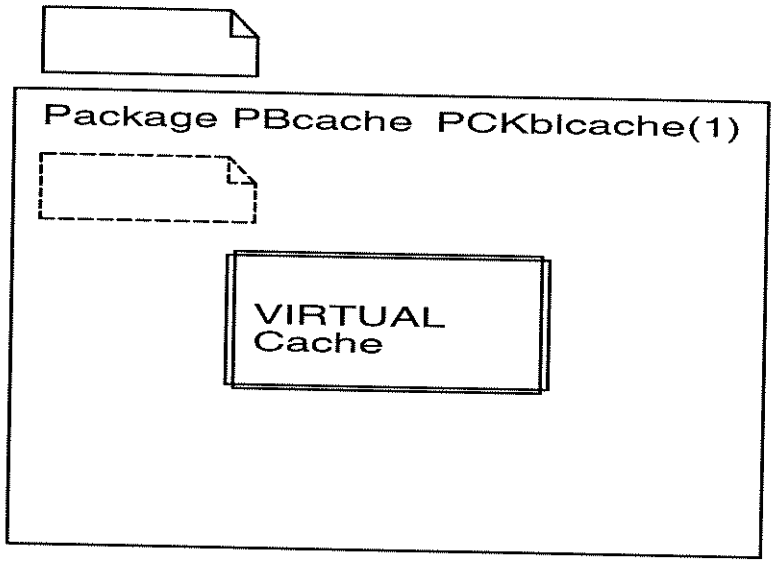

Figura C.27 - Package do Bloco Cache

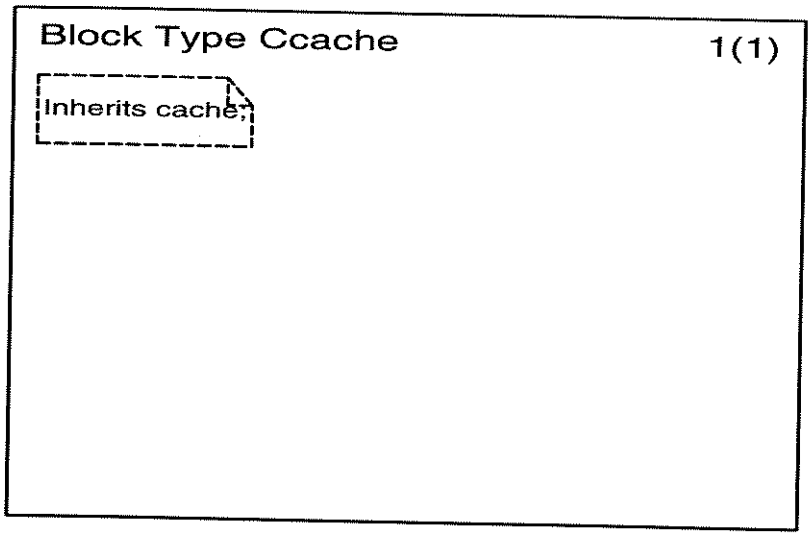

Figura C.28 - Block Type Ccache

C.5 - Sistema IP Clássico migrado para NHRP

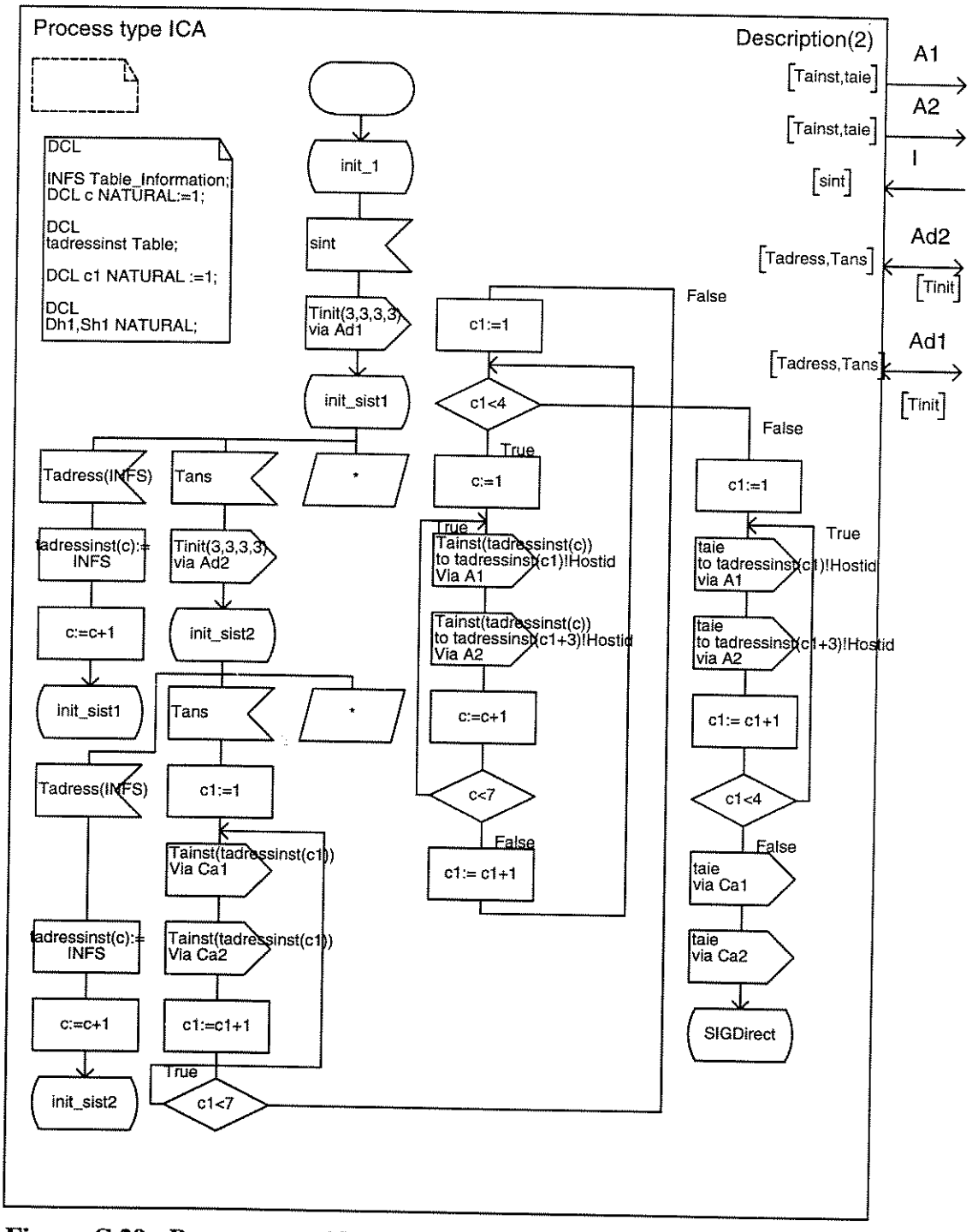

Figura C.29 - Processo servidor que herda o package do Processo Servidor do NHRP

C.6 - Sistema IP Clássico migrado que se comunica a uma nuvem NHRP

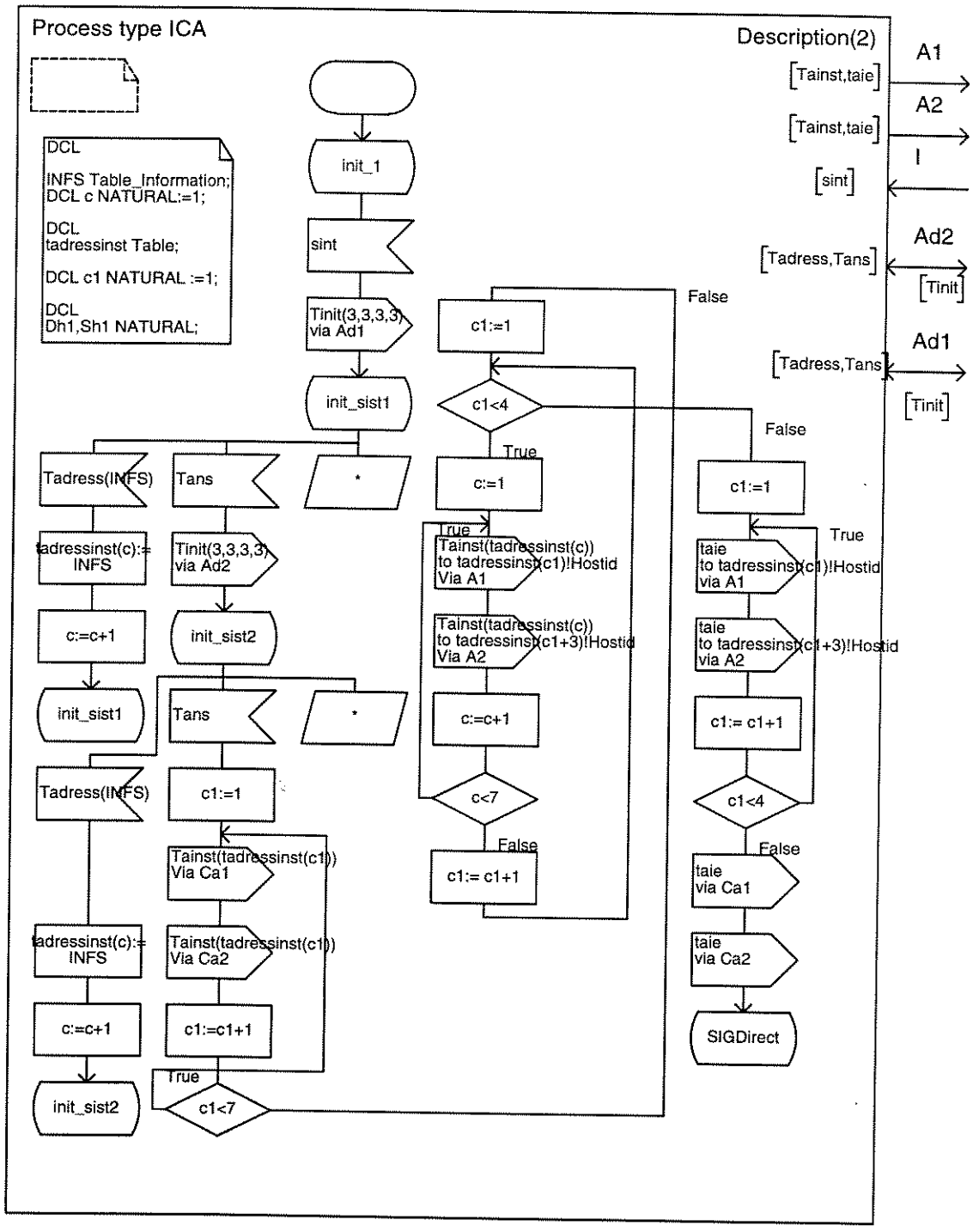

Figura C.30 - Process ICA (1)

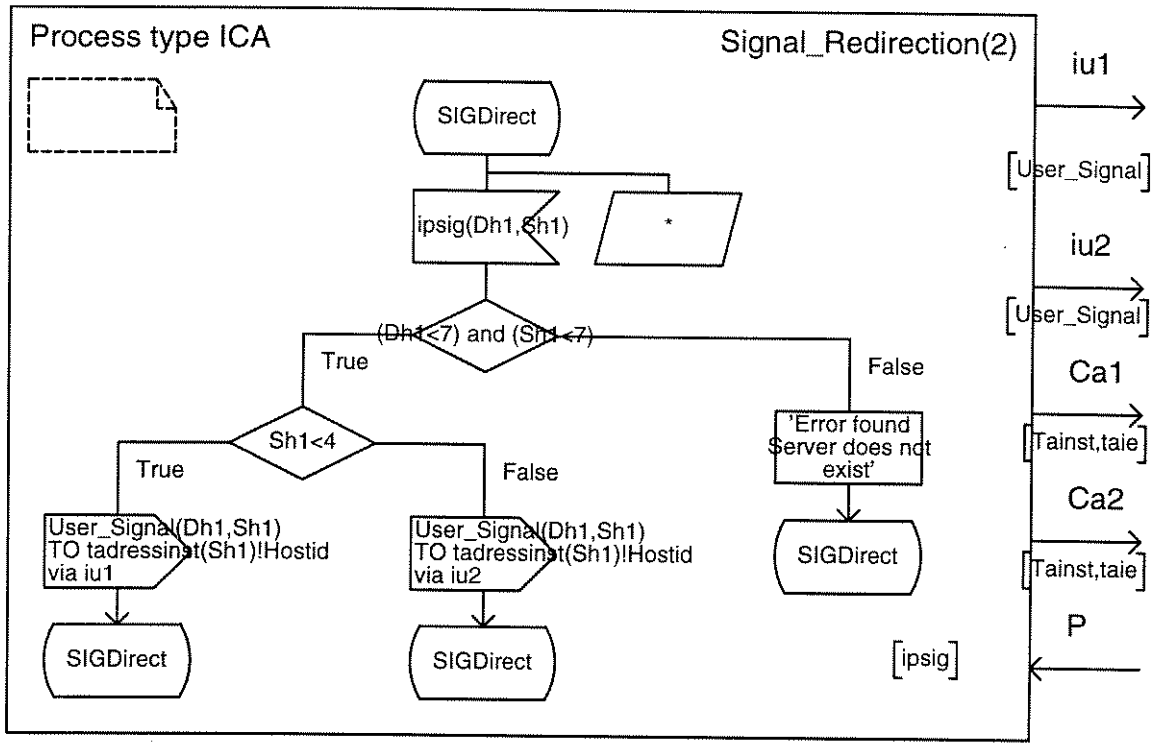

Figura C.31 - Process ICA (2)