FEEC
Faculdade de Engenharia Elétrica e de Computação

Universidade Estadual de Campinas
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE TELEMÁTICA

Especificação e Simulação na Linguagem Formal SDL das Sinalizações de Registro e Conexão de uma Rede ATM Sem Fio

Matheus de Carvalho Barreto
Graduado em Engenharia Elétrica pela Universidade Federal do Ceará

Banca Examinadora :
Dr. Walter da Cunha Borelli (**Orientador**) - DT/FEEC/UNICAMP
Dr. Antônio Fischer de Toledo - Telesp Celular
Dr. Michel Daoud Yacoub - DECOM/FEEC/UNICAMP
Dr. Shusaburo Motoyama - DT/FEEC/UNICAMP

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica.

Este exemplar corresponde a redação final da tese defendida por Matheus de Carvalho Barreto e aprovada pela Comissão Julgada em 30 / 04 / 1999
Orientador

30 de abril de 1999

UNICAMP
BIBLIOTECA CENTRAL

8418

N.º CHAMADA:
T/UNICAMP
B275e
V. Ex.
TOMBO BC/ 39012
PROC. 229/99
C [ ] D [x]
PREÇO R$ 11,00
DATA 14/10/99
Nº CPD

CM-00136234-6

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

B275e

Barreto, Matheus de Carvalho
Especificação e simulação na linguagem formas SDL das sinalizações de registro e conexão de uma rede ATM sem fio. / Matheus de Carvalho Barreto.--Campinas, SP: [s.n.], 1999.

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

1. SDL (Linguagem de programação de computador). 2. Redes de computação – Protocolos. 3. Simulação (Computadores). 4. Localização de falhas (Engenharia). I. Borelli, Walter da Cunha. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Aos meus pais, irmãos e minha avó

Agradecimentos

Esse trabalho é fruto de muita perseverança de minha parte, entretanto é injusto atribuir os méritos de sua conclusão somente a mim. Muitos foram os amigos que, de alguma forma, me auxiliaram durante o periodo do mestrado, contudo, antes de agradeçer a qualquer amigo agradeço a **Deus** por ter me dado forças para sair da casa de meus pais em Fortaleza e vir para a Unicamp, por não ter permitido eu desistir quando pensei em abandonar tudo e por sempre ter me guiado pelos desafios da vida.

Aos meus amigos, quase irmãos, Carlos Eduardo (Cacá) e Jaudelice (Jau), com quem dividi apartamento por quase dois anos, muito obrigado pela paciência que tiveram comigo, e por terem me ensinado muito, não somente no âmbito acadêmico, mas também na vida.

Aos colegas da Unicamp e da NEC do Brasil, que por serem muitos (graças a Deus!) não tenho espaço para citar os nomes de todos, e a todos os amigos do Ceará que vinheram comigo para Campinas, pois muito me ajudaram a enfrentar a vida longe de casa.

Aos administradores do laboratório do Departamento de Telemática, Jesus, Eduardo Trettel e Gustavo, pela paciência de sempre me ouvirem trazer problemas.

Aos professores da Universidade Federal do Ceará (UFC), em especial ao Prof. Dr. Josué Mendes Filho, que me orientou por três anos e meio em projetos de Iniciação Científica.

Ao meu amigo Everado Maia, enegeiro da Telemar, por seus conselhos e incentivos.

A Karla Érika, por sua ajuda revisando parte do texto escrito, e por sua paciência para escutar meus problemas, sempre me incentivando a resolvê-los.

Ao Prof. Dr. Walter da Cunha Borelli, meu orientador, por ter me dado a oportunidade de realizar o sonho de fazer meu curso de mestrado na Unicamp.

A banca, composta pelo Prof. Dr. Michel Yacoub e pelo Dr. Antônio Fischer, por terem ajudado a corrigir esse trabalho, deixando-o mais legível para o leitor.

A minha família, em especial meu pai (Cleomon), minha mãe (Amélia), meus irmãos (Fellype e André), minha avó (Cleonice) que tanto rezou, e reza, por mim e a Eugênia e a Raquel que de alguma forma fazem parte da minha família. Por durante todo esse tempo terem me ajudado com conselhos e orações, palavras e ações, a vocês meu MUITO OBRIGADO !!!

Matheus Barreto

Resumo

Nessa tese são especificadas formalmente em SDL (*Specification and Description Language*) e simuladas através do SDT[1] (*SDL Design Tool*), as sinalizações de Registro (*Registration*) e Conexão (*Call Setup*) de um terminal em uma rede ATM sem fio - WATM. Na especificação da sinalização de Registro, é proposta a inclusão de um sinal, *Updated*, cuja função é atualizar os dados de registro do terminal, quando ocorrer dele ser registrado inicialmente como visitante, e ter seu registro modificado para seu próprio domínio. Para a sinalização de Conexão é proposta uma modificação na seqüência onde é verificada a aceitação da qualidade de serviço (QoS) da conexão. Essa modificação visa diminuir o fluxo de mensagens entre os componentes da rede quando o pedido de conexão é recusado devido ao QoS. As especificações propostas foram simuladas com o objetivo de validação. Foram simuladas várias configurações para a rede WATM. Os resultados das simulações são apresentados em gráficos MSC (*Message Sequence Chart*) onde são mostradas as mensagens trocadas entre os elementos da rede, para cada uma das sinalizações especificadas. São também mostrados exemplos de falhas de especificação detectadas durante as simulações e suas correções.

Palavras-Chave: SDL - (*Specification and Description Language*), SDT - (*SDL Design Tool*), Especificação formal, Simulação, MSC - (*Message Sequence Chart*), Rede ATM sem fio - WATM (*Wireless ATM*), Sinalizações de registro e conexão de um terminal móvel, Detecção de falhas de especificação.

Publicação:

M. C. Barreto, W. C. Borelli, "Especificação em SDL'92 das Sinalizações de *Registration* e *Call Setup* de uma Rede Local ATM Sem Fio", Anais do XXIII CLEI (Conferencia Lationoamericana de informática), Valparaíso, Chile, 337-346, 1997.

[1]Pacote SDT (*SDL Design Tool; Telelogic, Sweden*), versão 3.3 (*SDT Base, MSC Editor and Simulator*): adquirido pelo DT/FEEC/UNICAMP através de um Projeto Temático FAPESP (Proc. 91/3660-0).

Abstract

In this thesis, the Registration and Call Setup signallings of a terminal in a wireless ATM network - WATM are formally specified in SDL (Specification and Description Language) and simulated using the SDT[2] (SDL Design Tool). During the Registration signalling specification, the insertion of a signal, *Updated*, is proposed, which has the function of updating the terminal registration data, whenever the terminal is initially registered as a visitor, and has its register data modified to its own domain. To the Call Setup signalling a modification on the sequence is proposed, where the acceptance of the quality of service (QoS) of the connection is verified. This modification aims at decreasing the messages flow among the components of the network when the connection request is denied due to the QoS. For the purpose of validation, the specifications proposed were simulated. Many configurations of the WATM network were simulated. The simulations results were presented in MSC (Message Sequence Chart) graphics, where the messages swapped among the network devices, for each specified signalling, are shown. Some examples of specifications faults detected during the simulations as well as their corrections were also shown.

Keywords: SDL - (*Specification and Description Language*), SDT - (*SDL Design Tool*), Formal specification, Simulation, MSC - (*Message Sequence Chart*), WATM (*Wireless ATM*), Registration and Call Setup Signalling of a mobile terminal, Specification fault detection.

Publications:

M. C. Barreto, W. C. Borelli, "Especificação em SDL'92 das Sinalizações de *Registration* e *Call Setup* de uma Rede Local ATM Sem Fio", Anais do XXIII CLEI (Conferencia Lationoamericana de informática), Valparaíso, Chile, 337-346, 1997.

[2]SDT Package (*SDL Design Tool; Telelogic, Sweden*), version 3.3 (*SDT Base, MSC Editor and Simulator*): purchased by DT/FEEC/UNICAMP under the Thematic Project FAPESP (Proc. 91/3660-0).

Glossário

- ABR - *Available Bit Rate*
- ACTS - *Advanced Communications Technologies and Services*
- AMPS - *Advanced Mobile Phone Service*
- ATM - *Asynchronous Transfer Mode*
- BAHAMA - *Broadband Adaptive Home ATM Architecture*
- CAC - *Connection Admission Control*
- CASE - *Computer Aided Software Engineering*
- CCITT - *Comité Consultatif International Téléphonique et Télégraphique*
- CBR - *Constant Bit Rate*
- CCC - Central de Comutação e Controle
- CDMA - *Code Division Multiple Access*
- CSC - Controladora da ERB Central
- DLC - *Data Link Control*
- DLS - *Domain Location Service*
- ERB - Estação de Rádio Base
- ETSI - *European Telecommunications Standards Institute*
- FIFO - *First In First Out*
- GCR - Grupo de Canal de Rádio
- GSM - *Global System for Mobile*
- HIPERLAN - *High-Performance Radio LAN*
- HR - *Home Register*
- IEEE - *Institute of Electrical and Electronics Engineers*
- IMT - *International Mobile Telecommunications*

- IP - *Internet Protocol*
- ISDN - *Integrated Services Digital Network*
- ITU-T - *Intertional Telecommunications Union - Telecommunication Sector* (antigo CCITT)
- LAN - *Local Network*
- MAC - *Medium Access Control*
- MR - *Mobile Representative*
- MSC - *Message Sequence Chart*
- MSP - *Mobile Switching Point*
- NIU - *Network Interface Unit*
- NMT - *Nordic Mobile Telephone*
- NNI - *Network-Network Interface*
- PDA - *Personal Digital Assistants*
- PBS - *Portable Base Station*
- PCMCIA - *Personal Computer Memory Card International Association*
- PHY - *physical-layer*
- PVC - *Permanent Virtual Connection*
- QoS - *Quality of Service*
- RDSI - Redes Digitais de Serviços Integrados
- RCR - *Research and Developmente Center for Radio System*
- RISC - *Reduced Instruction Set Computing*
- SDL - *Specification and Description Language*
- SDT - *SDL Design Tool*
- SVC - *Switched Virtual Connection*
- TACS - *Total Access Control System*
- TDMA - *Time Division Multiple Access*

- UBR - *Unspecified Bit Rate*
- UId - *User Identify*
- UNI - *User-Network Interface*
- VBR - *Variable Bit Rate*
- VCI - *Virtual Channel Identifier*
- VPC - *Virtual Path Connection*
- VPI - *Virtual Path Identifiers*
- WATM - *Wireless ATM*
- WLAN - *Wireless LAN*
- WLL - *Wireless Local Loop*

Índice

Resumo i

Abstract ii

Glossário iii

1 Introdução 1
- 1.1 Motivação e Objetivos . . . 1
- 1.2 Estruturação do Trabalho . . . 3

2 Conceitos Básicos de Redes 4
- 2.1 Introdução . . . 4
- 2.2 Tecnologia ATM - Modo de Transferência Assíncrono . . . 4
- 2.3 Topologia das Redes Sem Fio . . . 7
  - 2.3.1 Topologia Centralizada . . . 8
  - 2.3.2 Topologia Distribuída . . . 9
  - 2.3.3 Topologia Multihop . . . 10
- 2.4 Sistema Celular . . . 10
  - 2.4.1 Composição de um Sistema Celular Básico . . . 12
  - 2.4.2 Operações de um Sistema Celular . . . 13
    - 2.4.2.1 Inicialização da Unidade Móvel . . . 14
    - 2.4.2.2 Chamada Originada pela Estação Móvel . . . 15
    - 2.4.2.3 Terminação de Chamada para a Estação Móvel . . . 15
    - 2.4.2.4 Finalização da Chamada . . . 15
    - 2.4.2.5 Procedimento de *Handoff* . . . 15
- 2.5 Configurações das Redes WATM . . . 16
  - 2.5.1 Rede BAHAMA . . . 16
  - 2.5.2 *Hierarchical Wireless ATM LAN* - Rede Centralizada . . . 17
  - 2.5.3 Procedimento de Registro para as Redes Centralizadas . . . 18
  - 2.5.4 Procedimento de Conexão para as Redes Centralizadas . . . 19
- 2.6 Conclusão . . . 20

3 Rede ATM Sem Fio: Arquitetura e Sinalizações Propostas **21**
3.1 Introdução . . . 21
3.2 Arquitetura da Rede WATM . . . 21
3.3 Considerações sobre a rede WATM proposta . . . 23
3.4 Sinalizações propostas . . . 24
3.4.1 Sinalização de Registro - (*Registration*) . . . 25
3.4.2 Sinalização de Conexão - (*Call Setup*) . . . 27
3.5 Conclusão . . . 29

4 Especificação em SDL'92 da Rede WATM **30**
4.1 Introdução . . . 30
4.2 Especificação da rede WATM . . . 31
4.3 Bloco Mobile . . . 32
4.3.1 Processo Mobile . . . 34
4.3.2 Especificação da sinalização de Registro para o terminal móvel . . . 34
4.3.2.1 Procedimento RegisterMobile . . . 35
4.3.2.2 Procedimento Register_complete . . . 37
4.3.3 Especificação da sinalização de Conexão para o terminal móvel . . . 37
4.3.3.1 Procedimento Callup . . . 38
4.4 Bloco BaseStation . . . 40
4.4.1 Processo BaseStation . . . 41
4.4.2 Especificação da sinalização de Registro para a estação base . . . 42
4.4.2.1 Procedimento Register_request . . . 42
4.4.3 Especificação da sinalização de Conexão para a estação base . . . 43
4.4.3.1 Procedimento Connection_request . . . 44
4.4.3.2 Procedimento Connection_completeI . . . 45
4.4.3.3 Procedimento Connection_SetupI . . . 45
4.4.3.4 Procedimento Verify_blocking . . . 47
4.5 Bloco MobileSwitchingPoint . . . 47
4.5.1 Processo MobileSwitchingPoint . . . 47
4.5.2 Especificação da sinalização de Registro para o MSP . . . 48
4.5.2.1 Procedimento Updated . . . 49
4.5.3 Especificação da sinalização de Conexão para o MSP . . . 49
4.5.3.1 Procedimento Connection_located . . . 50
4.5.3.2 Procedimento Connection_Setup . . . 52
4.5.3.3 Procedimento Connection_refused . . . 52
4.5.3.4 Procedimento Connection_complete . . . 54
4.6 Bloco DomainLocationService . . . 55
4.6.1 Processo DomainLocationService . . . 55
4.6.2 Processo Home_Register . . . 56
4.6.3 Processo Visiting_Register . . . 56
4.7 Conclusão . . . 57

5 Simulação das Sinalizações de Registro e Conexão para a Rede WATM 59
5.1 Introdução . . . 59
5.2 Simulação do Registro de um Terminal Móvel . . . 60
5.2.1 Exemplo do Uso da Mensagem *Updated* . . . 64
5.3 Simulação de uma Conexão do Terminal Móvel . . . 67
5.3.1 Exemplo de Conexão onde o móvel não é encontrado . . . 71
5.4 Exemplos de falhas de especificação detectadas em simulações . . . 72
5.4.1 Exemplo com a sinalização de Registro . . . 73
5.4.2 Exemplo com a sinalização de Conexão . . . 75
5.5 Conclusão . . . 76

6 Conclusão 78
6.1 Considerações Gerais . . . 78
6.2 Sugestões para trabalhos futuros em redes WATM . . . 79

A Notação e Conceitos em SDL usados na Especificação da rede WATM 80
A.1 Introdução . . . 80
A.2 Estruturas . . . 82
A.2.1 Estrutura *cell* . . . 82
A.2.2 Estrutura *Home_struct* . . . 83
A.2.3 Estrutura *PIds_struct* . . . 83
A.2.4 Estrutura *Table_Struct* . . . 83
A.3 Mensagens . . . 84
A.3.1 Mensagens do System WATM . . . 84
A.3.2 Mensagens do Block Type Mobile . . . 86
A.3.3 Mensagens do Block Type BaseStation . . . 86
A.3.4 Mensagens do Block Type DomainLocationService . . . 87
A.4 Definição de Novos Tipo . . . 88
A.5 Matrizes . . . 88
A.5.1 Matriz *Table* . . . 88
A.5.2 Matriz *Table1* . . . 89
A.5.3 Matriz *Table2* . . . 89
A.5.4 Matriz *Table3* . . . 89
A.6 Variáveis . . . 89
A.6.1 Variáveis do **Process Signal_Manager** . . . 89
A.6.2 Variáveis do **Process Mobile_Id_Table** . . . 89
A.6.3 Variáveis do **Process Type Mobile** . . . 90
A.6.3.1 Procedure Callup . . . 90
A.6.3.2 Procedure RegisterMobile . . . 91
A.6.4 Process Base_Id_Table . . . 91
A.6.4.1 Procedure Which_MSP . . . 91
A.6.4.2 Procedure MSP_Begin . . . 91
A.6.4.3 Procedure Ask_Bases . . . 91

A.6.5 Process Type BaseStation 92
A.6.5.1 Procedure Register_request 92
A.6.5.2 Procedure Register_request 92
A.6.5.3 Procedure Connection_request 92
A.6.6 Process Type MobileSwitchingPoint 92
A.6.6.1 Procedure Updated 93
A.6.6.2 Procedure InitializeSystemI 93
A.6.7 Process Type Home_Register 93
A.6.7.1 Procedure write_data 93
A.6.8 Process Type Visiting_Register 93
A.6.9 Process Type DomainLocationService 94
A.6.9.1 Procedure data_inseted 95
A.6.9.2 Procedure Register_Base 95

B Processos e Procedimentos Auxiliares da Especificação **96**

Bibliografia **119**

Capítulo 1

Introdução

1.1 Motivação e Objetivos

Há uma crescente demanda por aparelhos que possuam a comodidade de serem móveis, e assim possam ser usados em qualquer lugar e a qualquer hora. Já está se tornando bastante comum a utilização de laptops, palmtops e PADs tanto em empresas como no uso doméstico. Contudo sempre existe a necessidade de conectar esses aparelhos em rede para que eles compartilhem informações entre si, e possam utilizar recursos de outros computadores de maior porte.

A grande maioria das redes utilizadas para a conexão dessas máquinas usa cabos, quer sejam coaxiais, pares trançados ou fibras ópticas. Esse tipo de rede limita a mobilidade, que é a grande característica desse tipo de equipamento, ao comprimento do cabo que o liga à rede. Surge então a necessidade de uma rede que não restrinja a mobilidade que é desejada a esses equipamentos.

As redes sem fio (*Wireless Network*) aparecem como uma alternativa a esse problema. Entretanto, a tecnologia a ser usada pela rede, Ethernet, Token Bus, Token Ring, ou mesmo rede local (LAN) ATM (*Asynchronous Transfer Mode*), está relacionada com o tipo de tráfego a ser transmitido pela rede.

Com o desenvolvimento da tecnologia digital, criou-se a necessidade de transmitir diferentes tipos de informações, como texto, áudio, vídeo e voz, dando origem aos sistemas multimídia, necessitando dessa forma, desenvolver arquiteturas de redes de alta velocidade. Como os sistemas de telecomunicações foram desenvolvidos para o transporte de tipos de informações específicas, surgiu a necessidade do desenvolvimento de um sistema integrado que transmita e processe todos os tipos de dados. Para prover essa necessidade foram propostas as Redes Digitais de Serviços Integrados RDSI - (*ISDN*[1]).

Apesar de as redes *ISDN* ainda não terem atingido o desenvolvimento esperado, elas já estão na segunda geração. A segunda geração, referenciada como *ISDN* de faixa larga (*B-ISDN*[2]), suporta altas taxas de dados, mais de 100 Mbps [28], e utilizam comutação orientada por pacote.

[1] *Integrated Services Digital Network.*
[2] *Broadband ISDN.*

A tecnologia ATM foi escolhida como solução para as redes *B-ISDN* pelo ITU-T (*International Telecommunications Union - Telecommunication Sector*) [21], por apresentar diversas vantagens como escalabilidade, potencial para criar LANs (*Local Area Network*) virtuais e principalmente pela habilidade de suportar interconexão de redes com qualidade de serviço (QoS[3]).

Da união dessas duas crescentes áreas de telecomunicações, surgem as redes locais sem fio que utilizam tecnologia ATM. O estudo nessa área de LANs ATM sem fio é razoavelmente recente, sendo que o conceito *Wireless ATM* - (WATM) foi proposto pela primeira vez em 1992, em um artigo de D. Raychaudhuri et al [22, 23].

A WATM tem o objetivo de prover suporte para as aplicações de redes multimídia, baseadas em QoS, que sejam executadas tanto em dispositivos fixos quanto em dispositivos móveis. Nesse ambiente, uma variedade de computadores, desde estações de trabalho a PCs móveis, comunicam-se cada um com o outro e/ou acessam informações de servidores remotos.

Para maximizar a usabilidade e performance é indicado que se tenha uma rede transparente e uma arquitetura de software que incorpore a parte com e sem fio do sistema. Uma solução mais abrangente deveria também prover flexível integração com o QoS, controlando tudo por intermédio dos protocolos de rede. As redes com tecnologia ATM fornecem uma base útil para esse sistema, dado que elas foram projetadas para integração de serviços com suporte para QoS.

Algumas padronizações para WLAN[4] estão sendo desenvolvidas em instituições como o IEEE (padronização IEEE 802.11), ETSI (European Telecommunications Standards Institute) - RES.10, e RCR (Research and Development Center for Radio System) do Japão [31]. Em particular, a padronização IEEE 802.11 foi concluída no início do segundo semestre de 1997, e desenvolveu um padrão para a tecnologia de rede sem fio, estabelecendo especificações sobre componentes MAC (controle de acesso ao meio) e PHY (especificação de camada física) de dispositivos sem fio, assim como sobre os dispositivos *spread spectrum* e infravermelho que operam na freqüência de 2,4 GHz e à taxa de transmissão de 2 Mbps. O padrão deve possibilitar a interoperabilidade entre os componentes incorporados ou adaptados em dispositivos móveis.

O objetivo desse trabalho é fazer a especificação formal das sinalizações de Registro (*Registration*) e Conexão (*Call Setup*) de um terminal que utilize uma rede local ATM sem fio (WATM). Na sinalização de Registro foi proposta a inclusão do sinal *Updated* que avisa ao terminal móvel a alteração dos seus dados de registro; na seção 5.2.1, capítulo 5, é apresentado um exemplo do uso desse sinal. Na sinalização de Conexão foi modificada a seqüência, proposta por [14], da verificação da qualidade de serviço da conexão. Com esta alteração, para o caso da conexão ser rejeitada pela estação base, que serve o terminal móvel que pediu a chamada, existirão menos trocas de mensagens do que com a sinalização proposta por [14].

[3] *Quality of Service.*

[4] As redes WLAN (*Wireless LAN*) que estão sendo padronizadas não utilizam tecnologia ATM.

Para realizar a especificação, foi utilizada ferramenta a CASE[5] SDT[6] como interface para o uso da linguagem SDL [8] (*Specification and Description Language*) em sua versão de 1992, SDL'92.

Esta ferramenta possibilita a simulação, com objetivo de validação, dessas especificações. Os resultados das simulações são apresentados sob a forma de gráficos temporais MSC [9] (*Message Sequence Chart*).

A análise dos gráficos MSC gerados a partir de simulações das especificações das sinalizações de Registro e Conexão, permitiu que fossem realizadas propostas, apresentadas nos capítulos 3 e 4, para as duas sinalizações especificadas.

1.2 Estruturação do Trabalho

Esse trabalho é dividido em seis capítulos, inclusive este, e dois apêndices.

O segundo capítulo apresenta os conceitos de redes de comunicação necessários para o entendimento desse trabalho. Esse capítulo aborda conceitos básicos de: redes ATM; topologias de redes sem fio; telefonia celular; e redes WATM.

O terceiro capítulo aborda a arquitetura e sinalizações propostas, nesse trabalho, para uma rede WATM. Nele é explicado como procede o registro de um terminal da rede WATM utilizando a sinalização de Registro. A apresentação da sinalização de Conexão para a chamada entre dois terminais da rede WATM também é feita no terceiro capítulo. Durante a explicação das duas sinalizações, Registro e Conexão, é realizado um estudo comparativo com as propostas existentes.

O quarto capítulo apresenta as propostas de especificação formal em SDL, das sinalizações de Registro e Conexão de um terminal, para cada dispositivo que compõe a rede WATM.

O quinto capítulo desse trabalho traz as simulações das sinalizações especificadas. Essas simulações têm o objetivo de validar a especificação. Para cada uma das sinalizações, Registro e Conexão, são apresentados dois casos, um mais comum e outro particular. No final desse capítulo são mostrados dois exemplos mais comuns de falhas de especificação detectadas durante as simulações, assim como as soluções adotadas para suas correções.

O sexto capítulo apresenta a conclusão do trabalho, e traz sugestões para eventuais trabalhos futuros.

O apêndice A apresenta a função e o tipo de todos os sinais trocados entre os processos em SDL. A apresentação de todas as variáveis, tabelas e estruturas usadas nos processos e procedimentos da especificação das sinalizações de Registro e Conexão, também é realizada nesse apêndice. Essa parte do trabalho é bastante útil para o entendimento do quarto e quinto capítulos.

O apêndice B contém as figuras da especificação gráfica em SDL das sinalizações de Registro e Conexão que não foram apresentadas no capítulo 4.

---

[5] *Computer Aided Software Engineering.*

[6] Pacote SDT (*SDL Design Tool; Telelogic, Sweden*), versão 3.3 (*SDT Base, MSC Editor and Simulator*): adquirido pelo DT/FEEC/UNICAMP através de um Projeto Temático FAPESP (Proc. 91/3660-0).

Capítulo 2

Conceitos Básicos de Redes

2.1 Introdução

O desenvolvimento das telecomunicações modernas está se procedendo ao longo de duas trajetórias distintas: na contínua evolução da infra-estrutura mundial da telefonia convencional, telefonia fixa, para aumentar o suporte para os serviços de multimídia de banda larga, e o grande aumento do uso da telefonia celular para os serviços de voz e dados.

Recentemente, considerável interesse tem surgido para o estudo do tópico "Wireless ATM". Entretanto, ainda não existe uma padronização definida nem pelo ATM Forum, nem pelo ITU-T [11].

Esse capítulo apresenta, na seção 2.2, os conceitos básicos da tecnologia ATM. São apresentados os tipos de conexões virtuais e as classes de qualidade de serviços (QoS).

A seção 2.3 apresenta as três principais topologias das redes sem fio. O objetivo dessa seção é o de conhecer as topologias utilizadas pelas redes WATM.

A seção 2.4 aborda o sistema de telefonia celular. São apresentados um breve histórico desse sistema, os componentes básicos do sistema celular, e por último as operações realizadas pelas unidades móveis do sistema de telefonia celular. Alguns conceitos apresentados nessa seção são usados nas redes WATM, como por exemplo o conceito de célula.

Na seção 2.5 é abordada a configuração das redes WATM. São dados exemplos de protótipos que estão sendo desenvolvidos para essas redes. São apresentados exemplos de redes WATM que utilizam as topologias apresentadas na seção 2.3, e explicado como ocorre o registro e conexão de um terminal nas redes de topologia centralizadas. A explanação ocorre apenas para essa topologia, pois esta foi a selecionada para esse trabalho.

2.2 Tecnologia ATM - Modo de Transferência Assíncrono

A necessidade por comunicação em faixa larga nas redes fixas tem levado ao desenvolvimento de uma nova tecnologia de comutação chamada Modo de Transferência Assíncrono (ATM) [2, 6, 21, 28]. A tecnologia ATM oferecerá maior flexibilidade e performance que as tecnologias já existentes, sendo capaz de integrar todos os tipos de dados em uma

única rede, usualmente chamada de Rede Digital de Serviço Integrados de Faixa Larga (*B-ISDN*).

As redes de telecomunicações tradicionais são especializadas havendo uma clara relação entre o serviço ofertado e a rede utilizada. Muitos dos serviços de telecomunicações têm suas próprias redes, e essas redes, normalmente, não são muito convenientes para suportar bem outros serviços que não sejam aqueles para os quais elas foram previamente projetadas. Como um exemplo, a rede de telefonia pública oferece conexão de voz de dois caminhos (full-duplex), enquanto que a transmissão de programas de áudio de alta qualidade é feito usando redes de rádio especializadas.

A intenção da *B-ISDN* é suportar todos os tipos de serviços em uma única rede. Devido a isso, a rede *B-ISDN* necessita de uma tecnologia de comutação extremamente flexível. A tecnologia ATM tem sido desenvolvida para preencher as necessidades da rede *B-ISDN*. Enquanto que o ATM pode ser considerado como um modo de transferência para a transmissão física que permite altas taxas de transmissão de dados, *B-ISDN* é uma especificação de rede usando a tecnologia ATM [21].

Uma rede ATM consiste de um conjunto de comutadores ATM interconectados por conexões ATM ponto-a-ponto ou interface. O comutador ATM suporta dois tipos de interfaces: interface usuário-rede, (UNI[1]), e interface rede-rede, (NNI[2]).

A interface UNI conecta os roteadores e servidores ao comutador ATM. A interface NNI proporciona uma conexão entre dois comutadores ATM quaisquer que se comuniquem usando o protocolo NNI [2]. Para conexão entre um comutador ATM de uma rede privada e um comutador ATM da rede pública é usada a interface NNI-P. O protocolo NNI é usado dentro das redes ATM para rotear pedidos de sinalização ATM entre os comutadores ATM.

As redes ATM são fundamentalmente orientadas a conexão. Devido a isso, existe a necessidade de ser estabelecido um circuito virtual entre a rede ATM antes de qualquer transferência de dados. Os circuitos virtuais são de dois tipos: Caminhos Virtuais, determinados por Identificadores de Caminho Virtual (VPI[3]) e Canais Virtuais, determinados pela combinação de um VPI e um Identificador de Canais Virtuais (VCI[4]). Na figura 2.1 é mostrado como ocorre a comutação dos VPI's e VCI's.

A operação de comutação do ATM estabelece uma tradução das tabelas de prioridade para a transmissão de dados. A maneira pela qual essas tabelas são estabelecidas determina dois tipos de conexões ATM: Conexões Virtuais Permanentes (PVC[5]) e Conexões Virtuais Comutadas (SVC[6]).

Uma PVC é uma conexão estabelecida por algum mecanismo externo, tipicamente um gerente de rede, na qual um conjunto de comutadores existentes entre o comutador ATM fonte e o destino, é programado com os valores de VPI/VCI apropriados. A sinalização ATM facilita o estabelecimento de PVC's, entretanto os PVC's sempre requerem alguma

[1] *User-Network Interface.*
[2] *Network-Network Interface.*
[3] *Virtual Path Identifier.*
[4] *Virtual Channel Identifier.*
[5] *Permanent Virtual Connection.*
[6] *Switched Virtual Connection.*

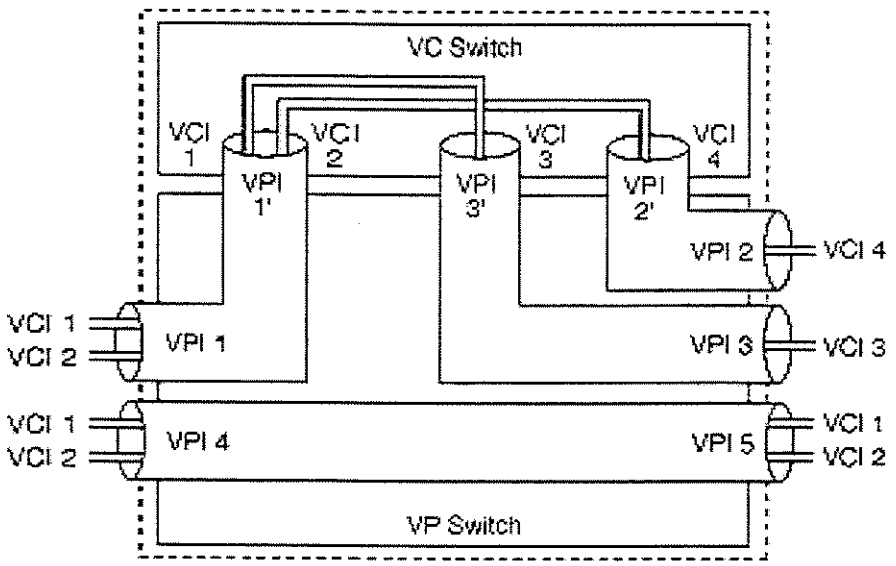

Figura 2.1: Comutação usando caminhos virtuais e circuitos virtuais [2]

configuração manual.

Uma SVC é uma conexão estabelecida automaticamente durante um protocolo de sinalização. SVC's não necessitam da interação manual como ocorre com as PVC's e portanto é esperado que sejam mais utilizados.

Uma das grandes vantagens do ATM é o suporte para garantir a qualidade de serviço (QoS) nas conexões. Quando uma conexão é pedida por um nó, pode ser requerida uma determinada QoS para a rede, e pode ser considerado que a rede irá prover essa QoS durante toda a conexão. As conexões são classificadas de acordo com as Classes de QoS das conexões ATM, tabela 2.1.

Classe QoS	Classe de Serviço	Descrição
1	A	performance de linha digital privada(emulação de circuito ou CBR - (*Constant Bit Rate*)
2	B	empacotamento de áudio/vídeo para teleconferência e multimídia VBR - (*Variable Bit Rate*)
3	C	protocolos orientados a conexão tal com frame relay VBR
4	D	protocolos orientados a conexão tal como IP e LAN emulation ABR - (*Available Bit Rate*)
5	não especificado	melhores esforços como definidos pelo provedor de serviços UBR - (*Unspecified Bit Rate*)

Tabela 2.1: Classe de Serviço [7]

Para prover uma garantia para a QoS, os comutadores ATM implementam a função

de Controle de Admissão de Conexão (CAC[7]). Qualquer instante em que um pedido de conexão é recebido pelo comutador, o comutador executa a função CAC. Baseado nos parâmetros de tráfego e no QoS pedido para a conexão, o comutador determina se a aceitação da conexão viola as garantias de QoS para o estabelecimento da conexão. O comutador aceita a conexão apenas se as garantias de QoS não forem violadas.

2.3 Topologia das Redes Sem Fio

As redes de computadores com fio podem ter seus nós conectados de várias maneiras; do mesmo modo os terminais em uma rede sem fio podem ser interconectados usando uma variedade de topologias. As configurações das redes de comunicação com fio, tal como uma rede de telefonia pública, redes de computadores *long-haul* e LAN's com fio, possuem algumas similaridades com as configurações das redes sem fio, mas existem importantes diferenças entre as redes com fio e sem fio.

As diferenças entre as redes sem fio e com fio ocorrem devido ao fato de as comunicações sem fio usarem fundamentalmente o modo de transmissão *broadcast*. Uma conseqüência dessa característica é que, como os terminais não são conectados uns aos outros utilizando fios, uma mensagem transmitida pode, a princípio, ser recebida por um número arbitrário de usuários. Além disso, devido às característica de rádio propagação, não pode ser sempre garantido a existência de um enlace do terminal que transmite para o terminal que recebe a chamada.

Nas comunicações sem fio, a potência com que o terminal transmite suas sinalizações é um parâmetro importante. Quanto maior a potência usada para transmitir a sinalização dos terminais, mais confiável é a comunicação entre os dispositivos que estão trocando informações. Entretanto, deve-se tomar cuidado para evitar uma excessiva interferência em outras redes que operem com a mesma freqüência.

Essas várias considerações têm influenciado o uso de diferentes tipos de topologia para as redes sem fio. Como em qualquer outra área de projeto de sistema, cada tipo de configuração tem suas vantagens e desvantagens. O objetivo final do projeto de uma rede sem fio é permitir que o usuário se comunique a taxas comparáveis àquelas que ele teria acesso com uma rede convencional, em uma mesma categoria de aplicação, para que o meio de transmissão venha a ser transparente ao usuário. Isso implicaria em taxas de transmissão para aplicações móveis, próximas às obtidas com os modem de alta performance, e no caso das redes WLANs, a taxas de transmissão aproximadas às obtidas pelas redes LANs convencionais. Esses objetivos das redes sem fio mostram a necessidade de que eficientes métodos de acesso ao canal sejam providos, e que diferentes métodos de acesso sejam apropriados a diferentes topologias de redes. As principais topologias empregadas em redes locais sem fio são três: centralizada, distribuída e configuração multihop.

[7] *Connection Admission Control.*

2.3.1 Topologia Centralizada

Na topologia centralizada, figura 2.2, uma estação serve como *hub* da rede, e as demais estações a utilizam para se comunicarem. A estação que serve de *hub* controla as estações usuárias e monitora o que cada estação está transmitindo. A estação hub é a responsável pelo gerenciamento do acesso das estações usuárias e pela alocação de largura de faixa.

Nessa topologia não existe possibilidade de comunicação direta par-a-par, isto é, quando um terminal conecta-se com outro, o *hub* deve intermediar a comunicação. Ela é comum nas redes telefônicas, onde todos os assinantes de uma determinada área estão conectados a uma central de comutação. Outro exemplo desse tipo de configuração é a topologia estrela utilizada nas redes LAN's convencionais. Os sistemas de telefonia móvel celular usam uma configuração de rede centralizada para servir aos terminais móveis que estão sob a cobertura de uma determinada célula.

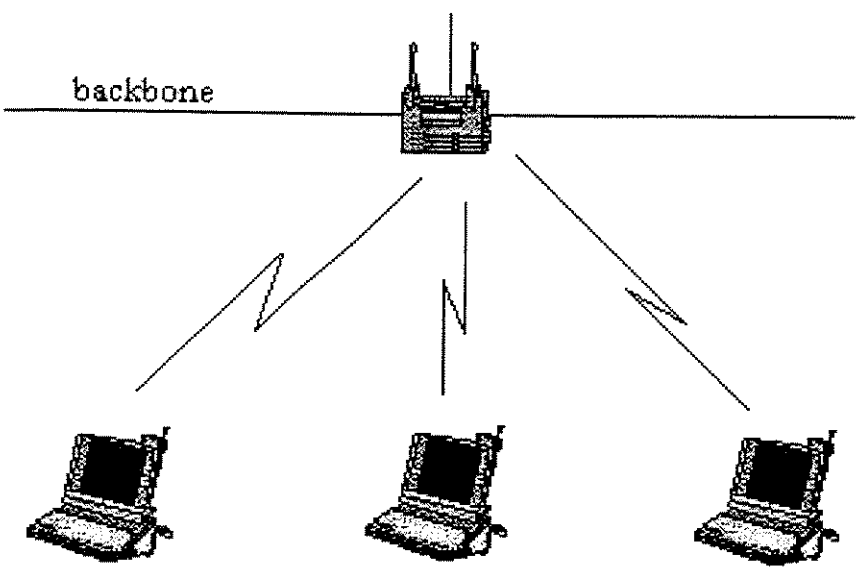

Figura 2.2: Topologia Centralizada

Uma importante vantagem da topologia centralizada é que a rede pode ser projetada para operar com um uso eficiente da potência de transmissão. Comparando uma rede que utilize topologia centralizada com uma que permita a conexão entre dois terminais sem o uso do *hub*, ou seja redes par-a-par, é observado que os terminais da rede centralizada conseguem comunicar-se com outros terminais com uma potência menor que a utilizada por terminais de uma rede par-a-par [17]. Como, na rede centralizada, os terminais comunicam-se com a estação *hub*, esta última deve ser instalada em uma localização que seja otimizada em relação ao conjunto de usuários dessa estação, para que os terminais usem a menor potência de transmissão possível sem denegrir, contudo, a qualidade de conexão.

Na configuração centralizada, a estação *hub* é o ponto natural ao qual é efetuada a ligação da rede ao *backbone*. Por essa última razão muitas redes sem fio de computadores

adotam a configuração centralizada.

2.3.2 Topologia Distribuída

Na topologia distribuída, apresentada na figura 2.3, todos os terminais podem comunicar-se entre si diretamente. Essa topologia é a solução ideal para as aplicações de rede *ad hoc*, onde vários terminais portáteis, tal como laptops, desejem formar uma rede de rápida organização. A capacidade de comunicação par-a-par dessa topologia, permite uma conexão instantânea sem qualquer necessidade de um controlador central. Entretanto, nas redes sem fio de computadores existentes, a configuração distribuída não é tão largamente usadas quanto a centralizada, pois os transmissores, necessitando operar a altos níveis de potência (conexão par-a-par), podem interferir acidentalmente em receptores que estejam próximos ao terminal transmissor [17].

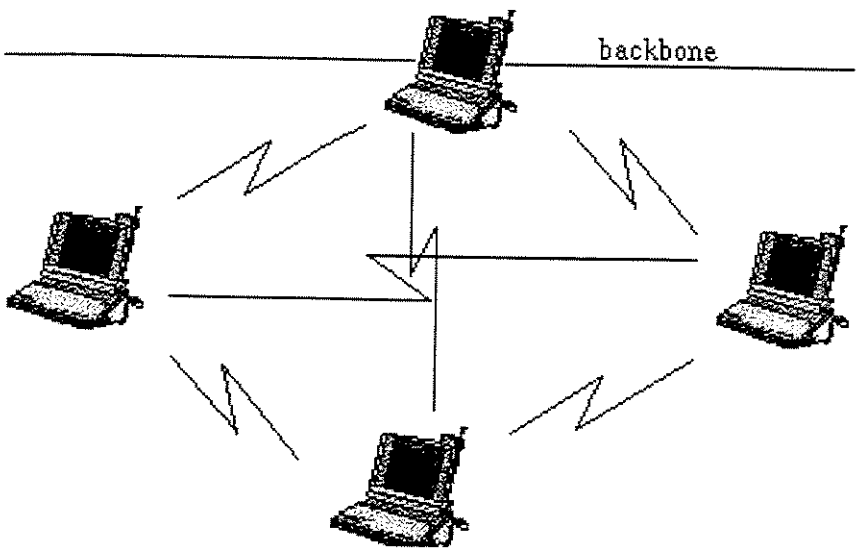

Figura 2.3: Topologia Distribuída

Nessa topologia, a conexão com o *backbone* é realizada por um nó servidor que atua como ponte ou *gateway*. Uma importante vantagem dessa topologia de conexão é que a falha de um ponto não prejudica toda a rede. Outra vantagem é que as mensagens par-a-par não sofrem com o atraso de armazenamento e reenvio da mensagem, atraso interente à topologia centralizada. Dado que nenhuma função de roteamento necessita ser implementada em qualquer das estações, a complexidade do equipamento em seu projeto pode ser minimizada. Uma desvantagem dessa topologia é o fato de que transmissores operam a um alto nível de potência; isso pode ocasionar interferência indesejável.

2.3.3 Topologia Multihop

Nessa topologia, mostrada no figura 2.4, qualquer usuário pode comunicar-se com um outro via transmissão que passa por múltiplos enlaces na rede. Essa topologia provê a melhor conexão usuário-a-usuário dentre todas as topologias. A maior vantagem dessa topologia é a eficiência de potência, a qual deriva do fato de que a transmissão de mensagens entre usuários distantes é obtida com um conjunto de pequenas conexões.

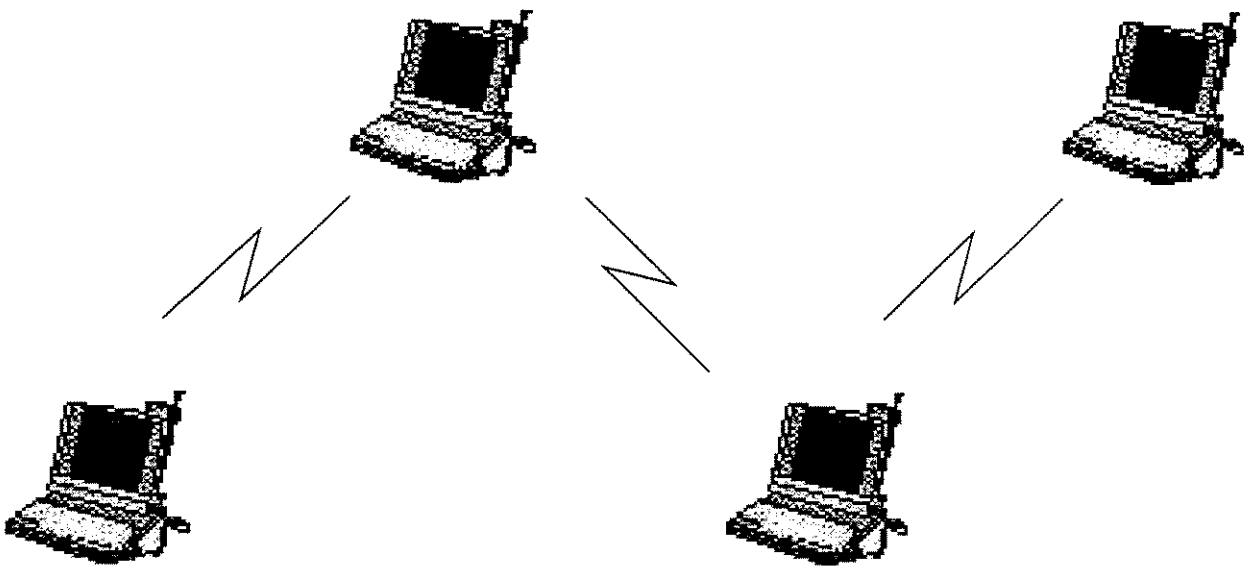

Figura 2.4: Topologia Multihop

Redes de multihop, como na topologia distribuída, possuem sua conexão ao *backbone*, ou a outras redes, provendo a um ou mais nós da rede a capacidade para essa conexão.

Uma desvantagem dessa topologia é a complexidade de projeto dos terminais, os quais devem possuir um eficiente algoritmo de roteamento. Além disso, devido à grande quantidade de enlaces que devem ser feitos para permitir a troca de mensagens entre dois usuários distantes, há atraso ocasionado pelo grande armazenamento e reenvio das mensagens. Comercialmente, essa topologia não vem sendo adotada pelas indústrias de redes sem fio para computadores [17].

2.4 Sistema Celular

Os primeiros sistemas móveis consistiam de uma estação base cuja transmissão e recepção eram efetuadas no pico de uma colina. A área de cobertura era escolhida para ser a maior possível, de maneira similar à transmissão dos sistemas de televisão e rádio. O transmissor geralmente operava com potência da ordem de 200 watts, possibilitando uma área de cobertura com aproximadamente 40Km de raio [34]. Esses sistemas são ditos não celulares, porque os conceitos de reuso de freqüência e *handoff* não eram conhecidos.

A primeira geração dos sistemas móveis celulares, baseados na tecnologia analógica, surgiu no início dos anos 80. Esses sistemas ofereciam baixa capacidade e poucos serviços. Os sistemas de segunda geração são baseados em tecnologia digital e oferecem um amplo conjunto de serviços incluindo serviços de mensagens e serviços de dados. A segunda geração possui maior capacidade que a anterior e possibilita um melhor suporte para o *roaming*. O serviço *roaming* é a facilidade de um usuário operar seu terminal em redes diferentes da qual ele foi habilitado. Os sistemas digitais também oferecem uma melhor segurança através do uso de criptografia e tecnologias de autenticação [13].

Atualmente os serviços de telecomunicação sem fio são oferecidos por sistemas de primeira e segunda geração especializados. Já é previsto que os sistemas de segunda geração serão incapazes de lidar com a grande demanda pela comunicação móvel no futuro, que requisitará altas taxas de dados para tráfego de multimídia. Mais chamadas por uma dada área, assim como uma grande gama de novos serviços sofisticados com grande largura de faixa devem ser suportados. O aumento da capacidade pode ser obtida através do uso de células de raio menor e a mudança para freqüências mais elevadas onde faixas mais largas estão disponíveis. O proposto sistema móvel de terceira geração seria integrado a uma nova rede fixa em desenvolvimento, especialmente a *B-ISDN*.

O IMT 2000 (*International Mobile Telecommunications*) é sempre referido como sistema móvel de terceira geração. O IMT 2000 pretende integrar todos os móveis, mas não prevê o suporte às conexões de faixa larga. A proposta de redes WATM é diferente da IMT 2000; elas propõem integrar as comunicações fixas e sem fio de faixa larga [6]. Na figura 2.5 há um resumo das gerações do sistema celular.

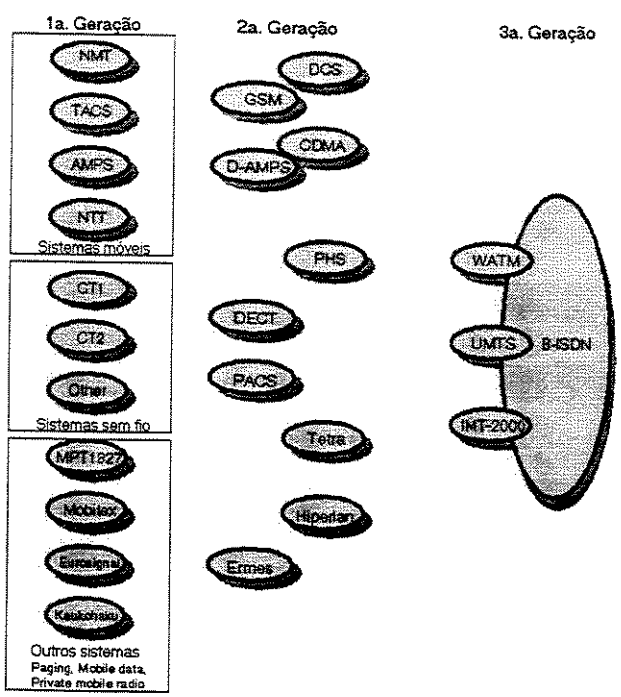

Figura 2.5: Gerações do Sistema Celular [6]

2.4.1 Composição de um Sistema Celular Básico

Um sistema celular básico, figura 2.6, consiste de três partes: a Unidade Móvel, Estações de Rádio Base (ERB's) e uma Central de Comutação e Controle (CCC) com conexões para ligar os três subsistemas.

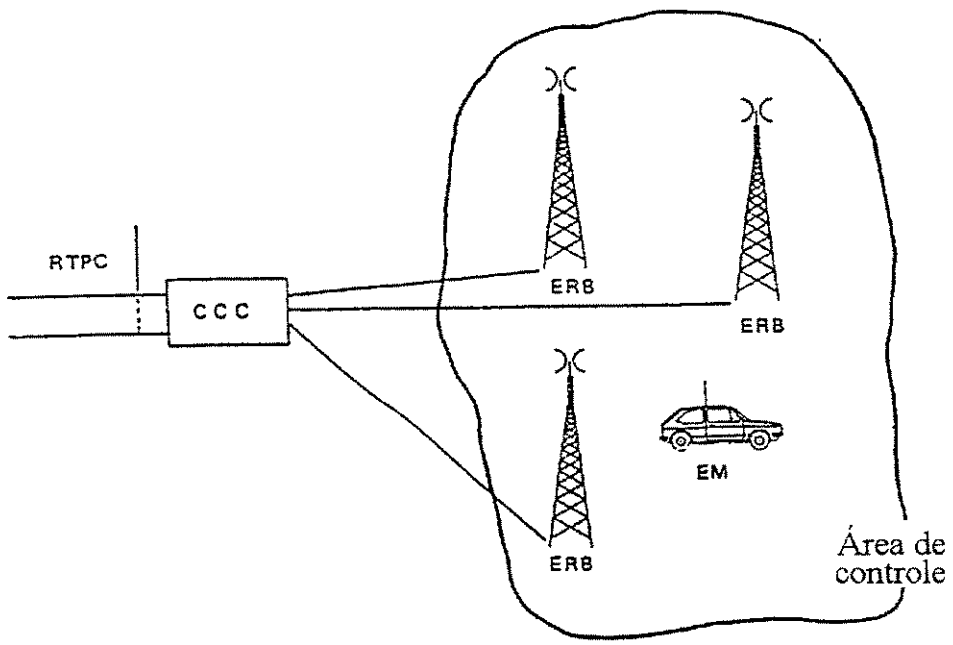

Figura 2.6: Configuração do sistema móvel celular [29]

A unidade móvel (telefone celular) contém uma unidade de controle, um transceiver e um sistema de antena. A ERB é responsável pela interface entre a unidade móvel e a CCC; a área de serviço de uma ERB é denominada célula. A ERB possui as seguintes unidades funcionais: Grupo de Canal de Rádio (GCR), Controladora da ERB Central (CSC), Combinador de Antenas, Antenas e Fonte de Alimentação. A composição da estação rádio base é mostrada na figura 2.7. O Grupo de Canal de Rádio compreende canais de voz e controle [34, 29].

A controladora da ERB (CSC) trabalha como um adaptador dos sinais entre a CCC e a ERB. Assim este equipamento recebe dados e voz das unidades de canais e envia estas informações para a CCC através do enlace dedicado ERB-CCC. Na direção oposta, o equipamento recebe dados e voz da CCC por meio do enlace de comunicação CCC-ERB e os envia para a unidade de canal ou controle correspondente. Dependendo do modo de transmissão (ERB-CCC), a velocidade do enlace para dados no canal de controle poderá ser de 2,4, 4,8, ou 9,6 kbits/s em enlaces analógicos, ou de 64 kbits/s se o enlace for digital [29].

A CCC coordena as ERB's de uma área de serviço; ela contém um processador celular e um comutador celular. A CCC faz a interface com a rede de telefonia fixa, com outras

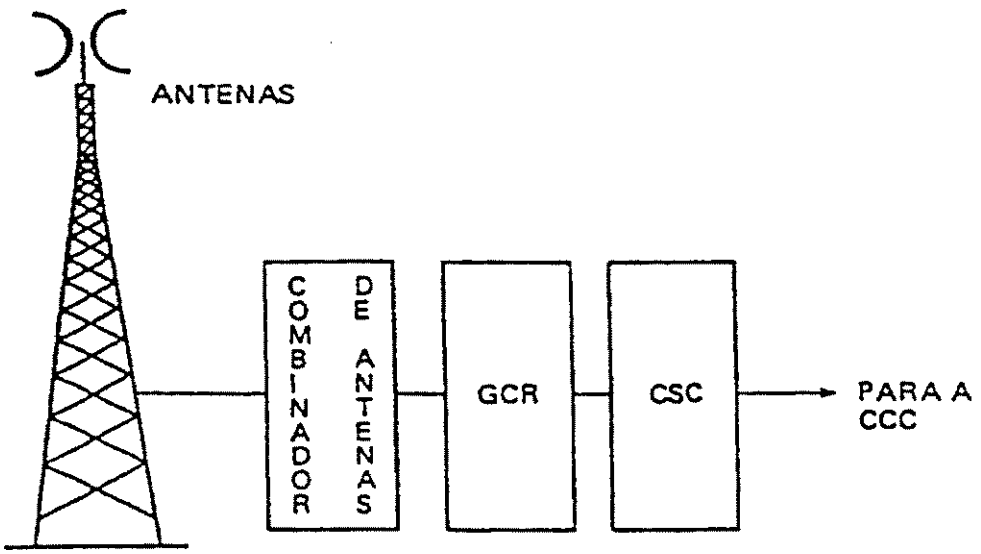

Figura 2.7: Composição da Estação Rádio Base [29]

redes celulares e com outras CCC's da mesma rede. Ela controla o processamento de chamadas, as atividades de bilhetagem, *handoffs*, e *roaming*. A CCC é o cérebro do sistema móvel celular. Seu processador é responsável pela coordenação da central e a administração do sistema celular.

O comutador celular, que tem que ser digital, comuta as chamadas para conectar usuários móveis a outros usuários móveis e a usuários da rede fixa. O entroncamento de voz utilizado é similar ao usado pela rede fixa para o entroncamento de voz entre centrais. Ele também contém ligações de dados para supervisionar as ligações entre o processador e o comutador, e entre as ERB's e o processador.

Na figura 2.8 é apresentado um diagrama básico da transmissão entre uma CCC e uma ERB.

2.4.2 Operações de um Sistema Celular

Durante a utilização de uma unidade móvel esta realiza diversas tarefas transparentes ao usuário e que são necessárias para possibilitar a mobilidade do dispositivo. Os detalhes de cada uma dessas atividades pode variar um pouco de acordo com a tecnologia utilizada para o acesso, contudo o conceito permanece o mesmo.

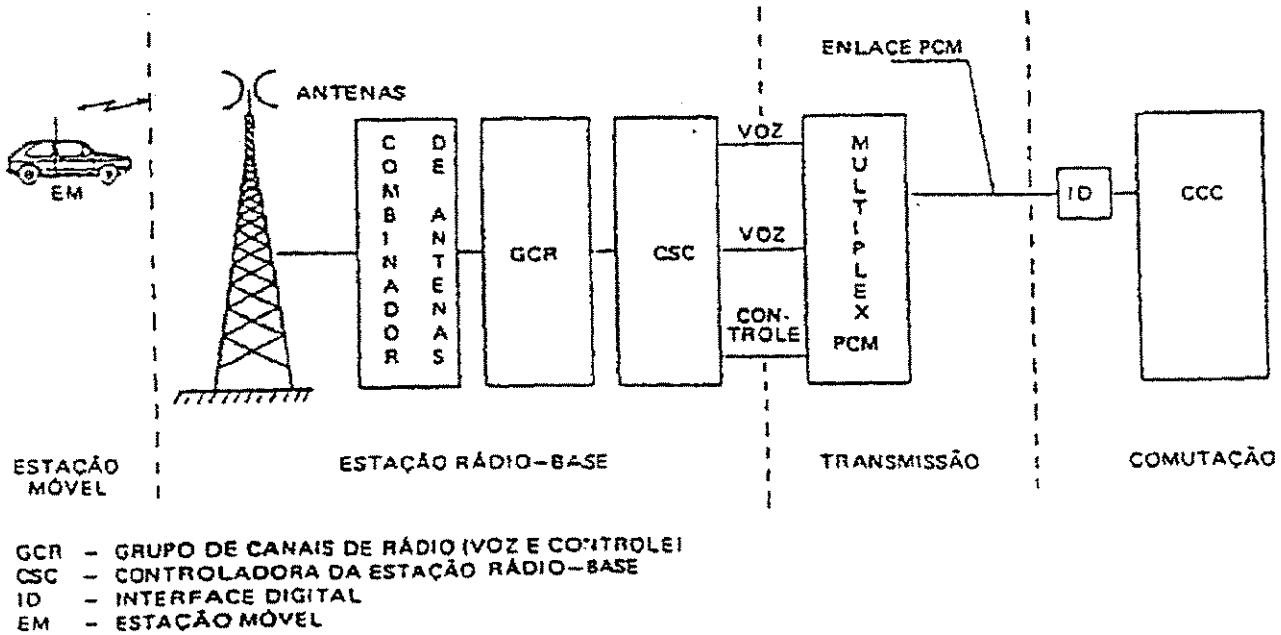

Figura 2.8: Interligação ERB-CCC [29]

2.4.2.1 Inicialização da Unidade Móvel

Quando o aparelho é ligado , o receptor procura um dos 21 canais estabelecidos de um total de 416, quando é utilizada a tecnologia AMPS - (*Advanced Mobile Phone Service*). A unidade então seleciona o melhor sinal e sintoniza nele por um período de tempo. Dado que cada célula é identificada por um canal diferente denominado canal de controle, então sintonizando-se no canal mais forte geralmente significa selecionar a célula mais próxima. Essa auto localização é usada quando a unidade móvel está em estado de espera, ou *idle*, e é independente do usuário.

A vantagem da auto localização é que ela elimina o trabalho do sistema que é necessário para localizar a unidade móvel. A desvantagem é que nenhuma informação de localização da unidade móvel é obtida enquanto ela estiver em estado *idle*. Devido a isso, quando há uma chamada para a unidade móvel, o processo de localização pode ser demorado. Uma vez que a grande maioria das chamadas é originada a partir dos móveis, o processo de auto localização é justificado. Após um determinado período de tempo, o procedimento de auto localização é repetido [13].

Quando em *idle*, a estação móvel envia uma mensagem denominada registro de localização. Esta mensagem é transferida até o processador da central de comutação e controle (CCC), para que a CCC possua uma informação aproximada da ERB onde se encontra a estação móvel (EM) e possa buscá-la, de maneira eficiente, quando a EM receber alguma ligação.

2.4.2.2 Chamada Originada pela Estação Móvel

Quando o usuário deseja efetuar uma ligação pela unidade móvel, esta envia para a ERB um pedido para esse serviço através de um canal selecionado durante auto localização. Ao receber a mensagem do móvel, a estação rádio base (ERB) a envia para a CCC usando uma conexão de dados de alta velocidade. A CCC determina um canal de voz apropriado para a chamada, e a ERB o usa para a conexão com a unidade móvel.

2.4.2.3 Terminação de Chamada para a Estação Móvel

Quando uma chamada é originada da rede fixa para uma unidade móvel, a central da rede fixa (PSTN[8]) identifica o número chamado como o de um móvel e direciona a chamada para a CCC. A CCC envia uma mensagem de localização, ou mensagem de *paging*, para as ERB's. Cada ERB ao receber a mensagem da CCC, a transmite por suas células por intermédio do canal de controle de cada uma. O móvel reconhece sua identificação no canal mais forte, sintoniza nele, e responde para a ERB. O móvel também segue a instrução de sintonizar o canal de voz determinado pela CCC e inicia o tom que informa ao usuário a chamada de uma ligação.

2.4.2.4 Finalização da Chamada

Quando a ligação é encerrada, um sinal particular é enviado para a ERB, e ambos os lados liberam o canal de voz. A unidade móvel reinicia o monitoramento das mensagens de configuração do sistema e busca através do canal de controle mais intenso.

2.4.2.5 Procedimento de *Handoff*

Durante uma chamada, se a unidade móvel direcionar-se para fora da área de cobertura de uma célula sobre a qual ela está, a recepção começará a ficar fraca. Será necessário um *handoff*. Existem dois tipos de *handoff* básicos nos atuais sistemas em operação: O *Mobile Assisted* utilizado nas tecnologias CDMA - (*Code Division Multiple Access*), TDMA - (*Time Division Multiple Access*) e GSM - (*Global System for Mobile*) e o *Base Station Directed* que é usado nos sistemas AMPS - (*Advanced Mobile Phone Service*), TACS - (*Total Access Control System*) e NMT - (*Nordic Mobile Telephone*). No *handoff Mobile Assisted* é a estação móvel que irá pedir o *handoff* quando detectar que o sinal tornou-se fraco. No *handoff Base Station Directed* é a ERB que toma as decisões, controlada pela CCC.

Quando for executado o *handoff*, o sistema comutará a chamada para um novo canal de freqüência, em uma nova célula, sem interromper a ligação ou informar ao usuário que a troca de canal será realizada. A ligação continuará enquanto houver células para efetuar o *handoff* na direção em que o móvel se direciona. Caso o móvel não consiga mais fazer *handoff*, a qualidade da ligação ficará ruim, pois aumentará o ruído e culminará com a queda da ligação. O procedimento de *handoff* foi primeiramente usado pelo sistema

[8] *Public Switched Telephone Network*

AMPS, para o sistema europeu, devido ao diferente significado entre o inglês britânico e o americano, foi renomeado para *handover*.

2.5 Configurações das Redes WATM

Os dispositivos que compõem uma rede WATM dependem basicamente da topologia escolhida. Existem principalmente dois tipos de topologia em estudo, uma baseada em topologia mista, conhecida como *Broadband Adaptive Home ATM Architecture* (BAHAMA) [33, 12], e outra em topologia centralizada, isto é, a *Hierarchical Wireless ATM LAN* [33]. As redes BAHAMA foram desenvolvidas, e estão sendo pesquisadas, pelo *Bell Laboratories/Lucent* [33, 12].

2.5.1 Rede BAHAMA

A rede BAHAMA (*Broadband Adaptive Home ATM Architecture*) [33, 12], figura 2.9, é composta por dois elementos: estações bases portáteis (PBS's) e terminais móveis. Cada PBS combina as funções de comutador ATM e interface de acesso aéreo. As estações bases são projetadas para serem portáteis, pois dessa forma podem ser convenientemente relocadas quando necessário. Os procedimentos da rede BAHAMA foram desenvolvidos para permitir que essas estações bases sejam conectadas usando qualquer tipo de topologia de rede de computadores conhecida, aumentando desse modo a flexibilidade para posicionar os PBS's onde quer que eles sejam necessários. Os terminais móveis podem ser assumidos como laptops portáteis equipados com interface ATM e capacidade para utilização do meio aéreo para comunicação.

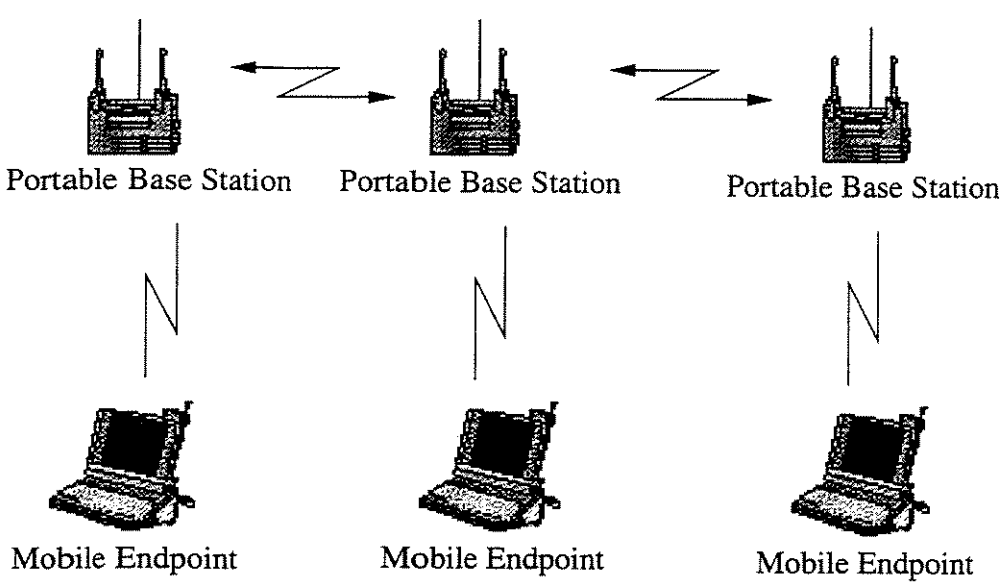

Figura 2.9: Rede BAHAMA

As redes LAN's BAHAMA usam topologia de comutação distribuída, onde cada PBS possui pequenas estruturas comutadoras. Devido a isso é muito provável que a conexão entre dois terminais deva passar por vários PBS's. Com o intuito de diminuir o tempo de estabelecimento de conexão para troca de dados, especialmente para aplicações tal como dados de protocolo internet (IP), cada par de PBS's é ligado com conexões preestabelecidas.

Tais conexões preestabelecidas são feitas conectando-se cada par de nós via um VPC (*Virtual Path Connection*). Desse modo não é necessário uma configuração de comutadores nó-a-nó para troca de dados. Entretanto, esse recurso adotado reduz a utilização da rede. Devido a isso, as redes BAHAMA em vez de usarem pares de VPC's utilizam Árvores Virtuais (*Virtual Tree*) [12].

O algoritmo de árvores virtuais permite a multiplexação de células de diferentes fontes para o mesmo destino dentro da mesma árvore virtual. Apesar de o algoritmo de árvores virtuais resultar em uma melhor utilização da rede quando comparada ao uso de pares de VPC's, ele ainda requer um estabelecimento prévio de recursos.

A localização de cada terminal na rede BAHAMA não utiliza o servidor de *home*. Em vez disso, cada PBS sabe qual terminal móvel está localizado sob sua cobertura. Existem duas vantagens quando esse tipo de localização é utilizada, se comparado ao modo tradicionalmente utilizado nas redes de telefonia celular que usam o registro de *home*.

A primeira vantagem é que, dado que cada PBS tem conhecimento da localização de cada móvel que está situado sob sua cobertura, não é necessária a utilização da mensagem de *page* na interface aérea; isso melhora a qualidade pois os recursos desse meio de transmissão utilizado são limitados. A outra vantagem é que essa solução evita a localização centralizada de banco de dados, e portanto diminui o custo de administração de tais bancos de dados [33].

2.5.2 *Hierarchical Wireless ATM LAN* - Rede Centralizada

Para esse tipo de topologia pode ser utilizado mais de um tipo de configuração para a rede sem fio. Essas diferenças variam de acordo com o fabricante do equipamento ou mesmo da tecnologia empregada para o acesso ao meio de transmissão. Contudo, a principal característica dessa configuração permanece, e todas as informações da rede, relativamente aos usuários, convergem para o mesmo ponto.

A estrutura proposta por J. Porter e A. Hopper em [19] consiste em um grande número de células de transmissão, denominadas pico-células, cada uma servindo a uma estação base. A estação base atua nessa estrutura como um roteador (*gateway*) ou uma ponte entre a parte sem fio e a com fio da rede.

As estações base estão ligadas a um Ponto de Comutação Móvel, *Mobile Switching Point - MSP*. O ponto de comutação é responsável por fazer as conexões entre móveis, ou destes para terminais fixos.

Esse sistema possui um banco de dados usado para armazenar a localização atual de cada terminal móvel; esse banco de dados é o *Home Register - HR*.

Cada terminal móvel desse sistema possui uma Representatividade Móvel, *Mobile Representative - MR*. Essa entidade é um software capaz de controlar e administrar as funções do terminal.

Para a manutenção das tabelas das várias entidades dinâmicas dentro de um domínio determinado, esse sistema possui o Serviço de Localização do Domínio, *Domain Location Service*. O DLS troca informações de controle com o HR e o MR.

Outro exemplo de estrutura é dado por D. Raychaudhuhri, um dos precursores do WATM, em [22]. A estrutura da rede, denominada de *WATMnet*, é composta por quatro elementos: Unidade de Interface de Rede (NIU), Estação Base (BS[9]), Terminal Pessoal, e Comutador ATM.

A NIU é um dispositivo que incorpora antena, rádio, modem, acelerador MAC/DLC, processador e módulos de interface de barramento. Esse dispositivo é utilizado pelo terminal portátil e pela estação base por intermédio da interface PCMCIA (*Personal Computer Memory Card International Association*) e representa os dois extremos da interface aérea.

A estação base é responsável por fazer uma ponte entre a conexão ATM sem fio e a rede ATM fixa, em termos de dados e pilhas de protocolos de sinalização e controle. Cada estação base possui um processador *i960*[10] o qual roda um sistema operacional que permite multiprocessamento com o acréscimo de outros *i960* caso seja necessário. Existem ainda um cartão de interface ATM e um cartão adaptador PCMCIA que conectam-se à NIU.

O terminal pessoal móvel é um laptop PC *NEC Versa - M/100* rodando o sistema operacional *LINUX*. A interface NIU está conectada a cada laptop que compõe a rede via um cartão adaptador PCMCIA.

O comutador ATM usado na *WATMnet* é um comutador ATM para áreas locais de 2,4Gb/s. O software de sinalização do comutador foi modificado para suprir algumas novas funções. Um dos comutadores ATM usados no trabalho de D. Raychaudhuhri foi um *Fore ASX-100* com processador de controle UNIX/Sparc. Cada elemento da *WATMnet* tem suas funções descritas com mais detalhes no trabalho [22].

Como estrutura alternativa as propostas em [20] e [22], existe a proposta por D. Cox e B. Akyol [1]. Essa topologia é composta por quatro elementos: rádios, controladores aos quais os rádios estão ligados, banco de dados e o equipamento de interface ATM. Esse último dispositivo tem conectado a ele os controladores de rádio e o banco de dados. É essa interface que conecta a rede sem fio à rede com fio. D. Cox denomina esse conjunto de equipamentos como uma zona, figura 2.10, podendo uma rede complexa ser constituída de várias zonas [1].

2.5.3 Procedimento de Registro para as Redes Centralizadas

A operação de registro é realizada para manter a informação da localização dos rerminais móveis. O processo inicia com o envio do número de identificação do usuário (UId) e a identificação da zona. O terminal móvel reconhece a mudança de zona comparando a

[9] *Base Station*.

[10] Processador *Intel* de alta performance que utiliza arquitetura RISC

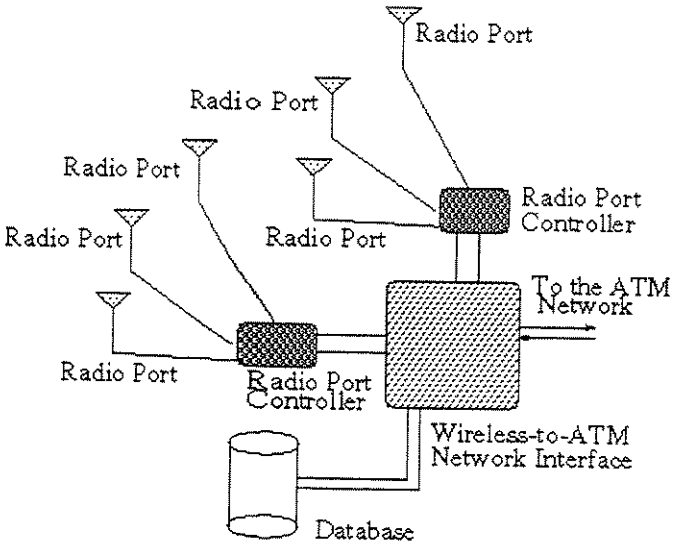

Figura 2.10: Estrutura proposta por D. Cox e B. Akyol [1]

identificação da zona atual com a última identificação de zona armazenada. Os rádios são responsáveis por transmitir a identificação de zona periodicamente para assistir os processos de registro. Após a troca de chaves de encriptação entre o terminal portátil e o rádio, a senha (*password*) do usuário é também transmitida por meio da interface aérea em uma mensagem criptografada.

Recebido o UId e a informação de autenticação, o comutador da nova zona estabelece uma conexão ATM para a zona que contém o perfil do usuário (zona *home*). A localização do perfil do usuário pode ser obtida questionando a última zona em que o móvel esteve.

Após a conexão ATM entre o comutador da nova zona e da zona *home* ser estabelecida, o comutador da zona atual do portátil requer a autenticação gravada do usuário. Com o recebimento dessa autenticação o usuário é autenticado.

Se a autenticação for bem sucedida, o perfil do usuário é autenticado com a informação da nova localização. O perfil atualizado é transferido para a zona corrente. O perfil do usuário na zona prévia é apagado.

2.5.4 Procedimento de Conexão para as Redes Centralizadas

Quando o móvel deseja estabelecer uma conexão, ele inicia da mesma forma que na rede fixa ATM, usando o protocolo Q.2931 [10] da mensagem de SETUP. Quando esse pedido é recebido pelo comutador, este identifica o móvel que originou a chamada, assim como o terminal chamado, e verifica a localização dessas duas unidades. Enquanto isso o comutador inicia uma máquina de estados para a chamada, cria uma gravação para mapear as conexões e realiza os procedimentos necessários para o roteamento. A aceitação de uma chamada inicial é baseada nos dados do perfil de serviço do usuário e nos requerimento de QoS determinados pelo móvel.

No caso do pedido de conexão ser aceito, as estações base, destino e origem, devem ser, respectivamente, notificadas com o novo tráfego esperado; então elas podem decidir pela admissão por parte da unidade de rádio, e reservar os recursos necessários [14].

2.6 Conclusão

Esse capítulo abordou os conceitos básicos de redes de comunicação. Na seção que tratou da tecnologia ATM, foi explicado o uso do QoS que é utilizado como parâmetro para a aceitação, por meio dos nós da rede, de uma conexão. Como foi apresentado nesta seção, seção 2.2, a verificação do QoS ocorre à medida que os comutadores que compõem a chamada recebem o pedido de conexão. Esse conceito será usado no capítulo 3 quando for apresentada a sinalização de Conexão, da rede WATM, especificada.

Na seção 2.3 foram apresentados alguns tipos de topologias utilizadas nas redes de comunicação sem fio. Das topologias indicadas, a mais utilizada pelas rede WATM é a centralizada, portanto foi a escolhida para ser utilizada na especificação das sinalizações.

A seção 2.4 apresentou os conceitos básicos de telefonia móvel celular. Alguns desses conceitos são utilizados pela rede WATM.

Na seção 2.5 foram apresentadas as redes WATM: *BAHAMA* e *Hierarchial Wireless ATM LAN* - (topologia centralizada). Foram comentadas quais topologias estão sendo utilizadas, e quais protótipos estão em desenvolvimento. Devido à topologia escolhida para a rede WATM desse trabalho ser centralizada, foi dado um enfoque maior para os projetos desenvolvidos que utilizam essa topologia.

Foi mostrado como ocorre o registro e conexão de um terminal móvel que utilize a rede WATM com topologia centralizada. No capítulo 3 é apresentado as propostas dessas sinalizações com o objetivo de serem especificadas na linguagem formal SDL.

Capítulo 3

Rede ATM Sem Fio: Arquitetura e Sinalizações Propostas

3.1 Introdução

Para a comunicação de qualquer terminal, seja ele móvel ou fixo, é necessário que ele se registre na rede. Com essa ação, o equipamento responsável pela gerência da rede torna-se ciente da existência desse terminal. Entretanto, apenas o registro do terminal não implica que ele esteja trocando informações com todos os outros terminais existentes, é necessário que ele requeira a conexão com um terminal específico, ou receba um pedido de chamada.

Para ser determinado como será a sinalização de registro e chamada, deve ser determinado o tipo de rede que se está utilizando e quais os dispositivos que a compõem.

Nesse capítulo é apresentada a arquitetura da rede WATM utilizada nesse trabalho. Na seção 3.2, dedicada à arquitetura da rede, são mostrados os seus componentes e suas funções. Na seção 3.3 são relatadas as considerações feitas para definir o comportamento funcional das sinalizações da rede.

Na última seção, seção 3.4, são descritas as propostas das duas sinalizações especificadas, Registro e Conexão, de um terminal na rede WATM. Para a explicação dessas sinalizações é mostrada a ação tomada por cada elemento da rede nas duas situações, sem, entretanto, citar a nomenclatura dos sinais o qual será feita no capítulo de especificação formal da rede WATM, capítulo 4.

3.2 Arquitetura da Rede WATM

As topologias mais usadas para as redes sem fio são a centralizada e a distribuída, sendo que o IEEE 801.11 permite as duas topologias, enquanto que a padronização *HIPERLAN* (*High-Performance Radio LAN*), desenvolvida pelo grupo RES-10 do ETSI's, concebe apenas a topologia distribuída [18].

As padronizações existentes prevêem somente o caso de redes WLAN, sem a utilização do ATM. Para o uso da tecnologia ATM em redes sem fio, até o final de fevereiro 1999

não existia padronização, sendo previsto pelo ATM Forum uma primeira tentativa de padronização da especificação das redes WATM para novembro de 1999, e de especificação para interconexão dessas redes para abril de 2000 [5]. Entretanto há alguns protótipos de redes WATM desenvolvidos pela *Lucent* com o projeto *BAHAMAS* [12, 33] e pela *NEC* [35, 22, 24, 23] com o *WATMNet*, ambos nos Estados Unidos. Na Europa, desenvolvido pelo programa de pesquisa *ACTS* (*Advanced Communications Technologies and Services*), existe o projeto *Magic WAND* [6]. Desses três protótipos, o *Magic WAND* e o *WATMNet* utilizam topologia centralizada e o *BAHAMAS* permite topologia distribuída para os comutadores e centralizada para os terminais.

Para o presente trabalho utilizou-se o topologia centralizada e foi tomado como exemplo para a especificação das sinalizações a estrutura sugerida por J. Porter e A. Hopper [19, 20], e desenvolvido na *Olivetti Research Limited.* Na figura 3.1 é mostrada a arquitetura da rede WATM utilizada.

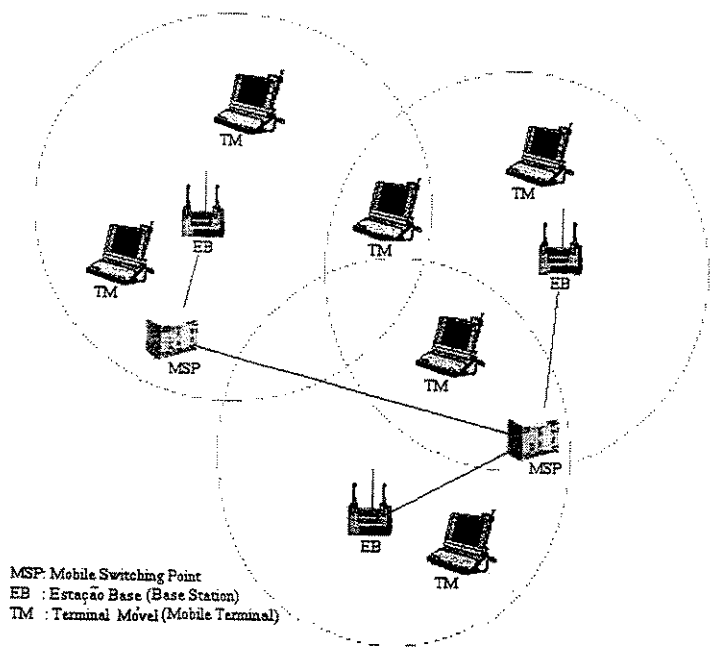

Figura 3.1: Arquitetura de uma rede WATM [3]

Essa rede é constituída por terminais móveis (**TM**), estações base (**EB**) e comutadores ATM (**MSP**).

Os terminais móveis, que podem ser exemplificados como laptops, trocam sinalização apenas com a estação base; essa é uma característica da topologia centralizada. Toda mensagem provinda do móvel é enviada ao comutador, e qualquer mensagem que vem do comutador para o móvel é transmitida pela estação base. Cada uma dessas estações é ligada apenas aos comutadores ATM. Essa conexão pode ser realizada tanto por meio de fios como por interface aérea, sendo a primeira opção a mais utilizada.

Os comutadores são do tipo dos utilizados em redes convencionais, com pequenas adaptações [14, 22]. Esses comutadores, ou MSP's (*Mobile Switching Point*) como é

comumente denominado na literatura, atuam como uma ponte entre a rede sem fio e as demais redes. Para as extremidades de uma conexão entre um terminal de uma rede com fio e uma sem fio, a natureza sem fio de uma das redes deve ser completamente transparente para ambos os lados da conexão. Cada MSP representa um domínio e a ele podem estar ligadas várias estações base; esta quantidade varia com o comutador utilizado.

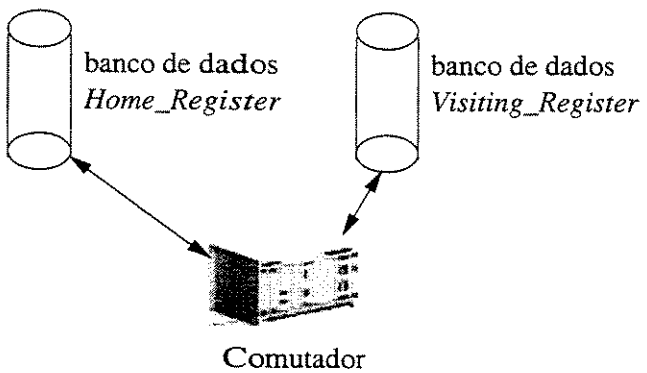

Figura 3.2: Arquitetura *two-tier database*

Cada comutador possui dois bancos de dados, figura 3.2. Um deles, o *Home_Register*, é utilizado para armazenar os registros realizados pelos terminais pertencentes ao domínio desse comutador. O outro, o *Visiting_Register*, guarda os registros de terminais de outros domínios que ao fazer a operação de registro estejam sob um comutador diferente do qual ele pertence, ou seja, estiver como visitante. Esse tipo de arquitetura com dois bancos de dados é conhecida como arquitetura *two-tier database* [1]. Essa estrutura é utilizada, por exemplo, pelo sistema de telefonia celular europeu, o GSM.

O conjunto de banco de dados, e o seu gerenciamento, formam o DLS [20] (*Domain Location Services).* Alguns autores como [1], [19] e [20], consideram esses bancos de dados como um elemento à parte do comutador. Enquanto outros como [22] e [32] não os mencionam, podendo estes estarem embutidos no comutador ou simplesmente não existirem.

Nesse trabalho incluiu-se o conjunto de banco de dados como um elemento distinto do comutador para evidenciar sua existência, assim como tornar a arquitetura a mais explícita possível. Optou-se pela arquitetura *two-tier database* [1] pois esta permite, se desejado, um melhor gerenciamento dos recursos da rede entre terminais locais e visitantes. Com um gereciamento mais apurado, é possível uma maior flexibilidade, em redes tarifadas, de uma taxa diferencial para terminais visitantes.

3.3 Considerações sobre a rede WATM proposta

Na rede sem fio desse trabalho considerou-se que os terminais movimentam-se apenas dentro da área de cobertura da célula por onde ele efetuou o registro. Isso faz com que

não sejam necessários alguns registros utilizados no sistema de telefonia celular, como por exemplo, o registro por tempo e o registro por zona.

Devido às considerações feitas, algumas características existentes nas redes de telefonia celular podem ser suprimidas das redes WATM de computadores, pois perdem suas utilidades. Dentre as funções não utilizadas, têm-se : função de *handoff* e função de localização[1].

A razão que prescinde a função de *handoff* nesse tipo de rede se dá ao fato de que, devido ao terminal não mudar de célula, ele não estará realizando *handoff*. Isso torna essa rede semelhante as redes WLL (*Wireless Local Loop*) [13]. Não obstante, ainda que fosse desejada a especificação do procedimento de *handoff*, deveria ser determinada a tecnologia de acesso ao meio (*TDMA*, *CDMA*), pois cada uma possui uma procedimento de *handoff* diferente do outro [13].

Para a ausência da função de localização do terminal, o motivo é análogo ao da função *handoff*. Como o terminal, após o registro, está sob a mesma estação base, sua localização torna-se conhecida para a rede, o que faculta o procedimento que encontra o terminal no sistema.

Em todo sistema celular, dependendo da proximidade das estações rádio base, e da existência, ou não, de obstáculos entre elas, podem existir áreas de intersecção entre duas células adjacentes. Nessas regiões o móvel está apto para trocar informações com qualquer uma das estações, ou até mesmo com as duas dependendo da tecnologia de acesso utilizada.

Nas redes sem fio de computadores o mesmo pode ocorrer. Apesar das considerações realizadas nesse trabalho, deve-se ponderar que, no momento do registro o terminal pode estar na região fronteiriça de duas células, podendo em um caso extremo, estas duas células pertencerem a domínios distintos. Esse caso particular do terminal estar localizado entre dois domínios distintos, não foi explicado de modo transparente por nenhum documento utilizado como apoio a esse trabalho.

Como característica das redes ATM tem-se a negociação da taxa de qualidade de serviço[2] (QoS) para determinar a possibilidade de realização da conexão. A rede sem fio utilizada nesse trabalho é baseada em tecnologia ATM, portanto uma função para verificar o QoS, requerido para a chamada, foi implementada. Ela é um dos quesitos para aceitação, ou rejeição, de uma conexão.

3.4 Sinalizações propostas

Nesse trabalho, dentre várias sinalizações possíveis de serem implementadas em uma rede WATM, escolheram-se as sinalizações de **Registro** - (*Registration*) e **Conexão** - (*Call Setup*). A opção se deu, pois essas duas são bastante genéricas, e necessárias para o funcionamento de qualquer rede, podendo ser utilizadas em qualquer rede WATM independente da tecnologia de acesso (*TDMA*, *CDMA*) utilizada pela rede.

[1] Uma explicação dessas funções encontra-se no capítulo 2.

[2] Expliações a respeito do QoS podem ser lidas no capítulo 2 ou nas referências [7, 21, 28].

3.4.1 Sinalização de Registro - (*Registration*)

Para se estabelecer uma conexão entre dois terminais em uma rede WATM, é necessário que o terminal de origem saiba onde está o terminal destino, contudo o terminal destino é um terminal móvel e portanto não possui um lugar fixo. Para resolver esse problema existem duas soluções aceitáveis [19].

A primeira solução é procurar o terminal destino em toda a rede. Isso pode ser feito usando um tipo de sinalização onde a procura é realizada em todas as estações base do sistema. Esse método é, contudo, inadequado para um sistema composto por um grande número de terminais distantes uns dos outros, isto é, uma rede complexa.

A outra solução é registrar o terminal em um banco de dados com sua posição atualizada. Esse banco de dados deve ser sempre renovado quando o terminal muda de localização. Utilizando essa solução, qualquer informação desejada a respeito de um terminal, que esteja ligado ao sistema, pode ser encontrada no banco de dados. A segunda solução é mais geral que a primeira; por essa razão foi o método escolhido para a especificação, capítulo 4.

O procedimento de registro é ativado quando um terminal móvel, que estava inicialmente desligado, é ligado. Quando o terminal é ligado, ele envia, em *broadcast*, uma sinalização contendo seu número de identificação e a identificação do domínio ao qual ele pertence [3]. Esses dados podem pertencer ao UId (*User Identify*) [27]. Todas as estações base, que estejam a uma determinada distância do terminal, distância esta que depende do alcance de recepção da estação base, irão captar o pedido de registro. Quando as estações recebem essa mensagem, elas enviam uma sinalização - (*acknowledge message*), para o terminal, confirmando o recebimento.

Nesse trabalho, sem perda de generalidade, o pedido de registro é enviado, para fins de exemplificação, apenas para três estações base distintas; pois devido a maneira como as estações base são distribuídas entre os MSP's (*Mobile Switching Point*), com essa quantidade de estações bases, sempre existe pelo menos dois MSP's envolvidos no processo, levando-se em conta que o sistema possua mais de um MSP.

As estações base estarão a distâncias diferentes de o móvel. Isso implica que, apesar do sinal enviado pelo móvel ser o mesmo, ele é recebido pelas estações base com uma intensidade de potência diferente.

Dependendo do sistema o próprio terminal é capaz de decidir, dentre as estações base que receberam o sinal, qual delas teve a melhor recepção. Isso pode ser realizado, pois as estações base podem informar, na mensagem de resposta, ou em outra mensagem, o nível de recepção.

Entretanto, essa escolha pode ser também realizada no MSP. Nesse caso alguns aspectos devem ser considerados. Por exemplo, existe a possibilidade de que as estações base que receberam o sinal pertençam a diferentes MSP.

Surgem algumas questões sobre esse cenário: Como cada MSP sabe que um outro também recebeu a mesma mensagem do mesmo móvel? E como eles sabem se o sinal chegado a eles não é recebido com uma intensidade de potência mais forte, ou mais fraca, que no outro? Esse problema não foi abordado claramente por nenhum dos documen-

tos utilizados para auxiliar esse trabalho, por isso ele foi escolhido para ser estudado, e especificado no capítulo 4.

Quando a estação base recebe o pedido de registro, ela o envia para o MSP. A estação incorpora a essa mensagem a potência com a qual ela é recebida [3].

Todo terminal móvel pertence a um domínio. A identificação desse domínio é enviada junto com a identificação do terminal. Portanto, quando o MSP recebe a sinalização de pedido de registro provinda de uma estação base, é verificado o domínio do terminal. Nesse trabalho, essa averiguação é realizada pelo DLS (*Domain Location Service*), pois este bloco é responsável pelo gerenciamento dos bancos de dados.

Se o MSP que receber o pedido de registro do terminal for o domínio dele, então é procurado no banco de dados *Home_Register* para saber se a potência de recebimento desse sinal é maior que a já registrada. Isso irá determinar a aceitação, ou rejeição, desse pedido de registro. Entretanto, se esse MSP não for o domínio do terminal, ele envia esse pedido de registro para o domínio que o terminal pertence, e aguarda uma resposta [3].

Quando o MSP, do domínio do terminal, recebe o pedido de registro proveniente de outro MSP, ele faz a verificação da potência de recepção do sinal, do mesmo modo como se o sinal estivesse sendo recebido por uma das estações base a ele ligadas.

Caso já haja registro desse terminal, é feita uma avaliação para determinar se ele deve ser registrado no novo domínio, como visitante [3]. Essa análise consiste na verificação da potência com que a estação base, ligada ao MSP do qual ele é visitante, está recebendo o pedido de registro.

A necessidade dessa averiguação ocorre, pois o móvel já está registrado no seu próprio domínio, e apesar de a estação base de um outro MSP ter recebido o pedido de registro, ela pode não ser o melhor servidor para o terminal. Entretanto, se o terminal estiver melhor servido pelo MSP que não a do seu domínio, este receberá uma mensagem para que a unidade móvel seja registrada no banco de dados *Visiting_Register* [3].

O terminal, mesmo estando como visitante, fica com seus dados armazenados no *Home_Register* do seu domínio. Esse procedimento é necessário, pois qualquer conexão, destinada a essa unidade, será primeiramente encaminhada para o MSP do seu domínio. Nos registros desse móvel constará sua atual localização para direcionar a conexão ao devido destino.

Pode ocorrer que quando o MSP do móvel receber o pedido de registro da unidade vinda de uma das suas estações base, esse terminal já esteja inscrito como visitante. A figura 3.3 mostra esse caso. O terminal (TM) pertence a MSP1 e registra-se primeiro no MSP2.

Nesse caso, se acontecer modificação do MSP, o MSP onde o móvel estava anteriormente registrado, deverá ser informado da alteração.

A necessidade dessa notificação foi verificada durante as simulações da especificação, e não está evidenciada nos trabalhos de outros autores [1, 14, 20]. No capítulo 5, seção 5.2.1, é apresentada uma simulação com esse exemplo.

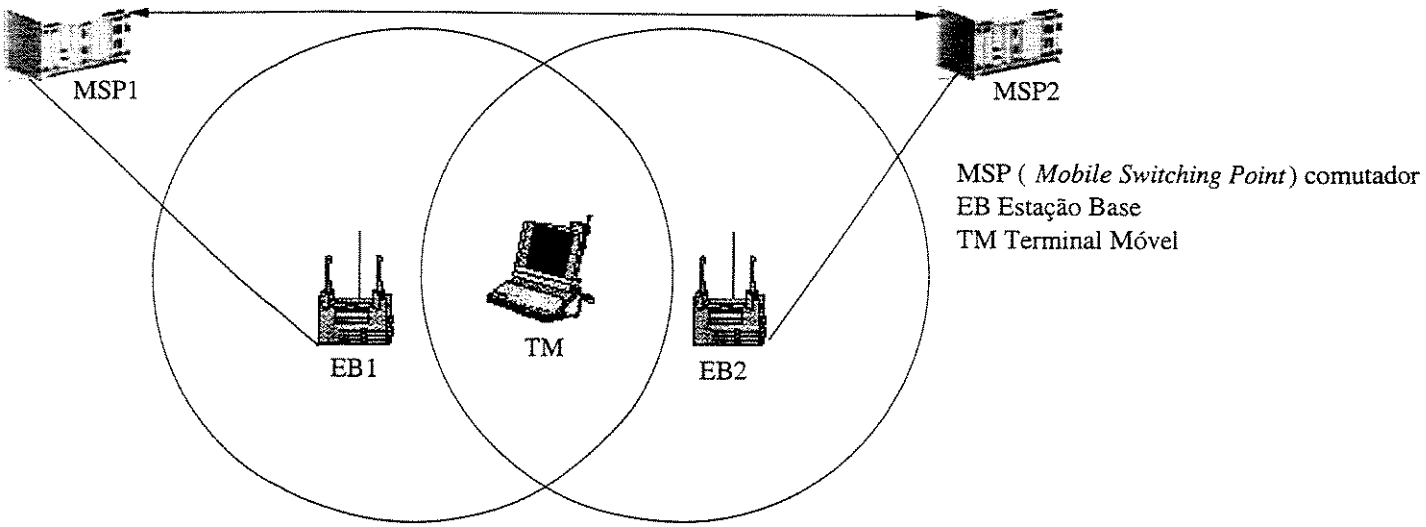

Figura 3.3: Registro do terminal

3.4.2 Sinalização de Conexão - (*Call Setup*)

A sinalização de conexão é utilizada quando o terminal deseja realizar uma conexão com outro terminal, móvel ou fixo. Para iniciar o processo do estabelecimento dessa conexão, o terminal origem envia um sinal que contém o *Q.2931 SETUP* [10].

A rede usada nesse trabalho utiliza tecnologia ATM, sendo então razoável levar em conta outros aspectos a serem analisados antes de a conexão ser realizada. Deve ser lembrado que o tráfego escoado por uma estação base é menor que o dimensionado para o comutador, pois este deve suportar o tráfego de mais de uma estação base.

A largura de faixa e o espaço de memória (*buffer*) disponível nessas estações base são dependentes não apenas da quantidade de tráfego, mas também do tipo de tráfego que transita através das estações base, e os recursos destas depende da qualidade da interface de rádio.

Quando o pedido de conexão atinge a estação base, ela envia esse requisito para o MSP. Essa unidade verifica se a conexão destino é para um terminal móvel.

No caso do alvo (*target*) da conexão ser um terminal fixo, o MSP verifica se a estação base do móvel é capaz de suportar a qualidade de conexão requerida pelo terminal. Uma vez que a estação base admite a conexão, o MSP envia o sinal de pedido de conexão, contendo a mensagem *Q.2931 SETUP*, para o comutador ATM do terminal destino. Como toda conexão ATM, ela somente será estabelecida se, e somente se, todos os nós de conexão intermediária aceitarem a qualidade de serviço requerida pelo terminal móvel em seu pedido de conexão.

Para o caso do destino da conexão ser outro terminal móvel, novamente a estação base da unidade que originou o pedido deve ser o primeiro elemento da conexão a concordar com os recursos pedidos. Apenas após essa aceitação a conexão passa adiante.

No trabalho [14], conforme explicado no capítulo 2, seção 2.5.4, é sugerido que essa verificação seja a última a ser feita. Entretanto, essa averiguação, nas redes ATM, é realizada quando cada conexão entre os nós é efetuada [2]. Devido a isso, é proposto aqui que a estação base do móvel que originou a chamada seja o primeiro dispositivo da rede, após o comutador, a analisar o recurso da ligação pedida pelo móvel.

Quando ocorre a situação de muitos terminais estarem utilizando a rede, as chances da rejeição de uma conexão, por falta de recursos, vir da estação base, são muito maiores que vir de um comutador, pois as estações base comportam menos tráfego que o comutador, uma vez que eles são projetados para escoar o tráfego de várias estações base. A averiguação sendo realizada primeiramente na estação base evitaria que, após a conexão ter sido feita, tenha que ser toda desfeita, pois a estação base do móvel que originou a chamada não comportou o recurso pedido.

Para realizar a conexão, o terminal destino deve ser localizado. Para tanto, o MSP do terminal que requisitou a chamada envia o pedido de conexão para o domínio a que o móvel destino pertence. Este verifica no seu banco de dados a localização do móvel, que pode estar como visitante em um outro MSP, e a envia para o MSP que requisitou essa localização.

Poder-se-ia sugerir que, uma vez que o móvel destino seja localizado no seu próprio domínio, o MSP já verifique as diretivas do *Q.2931 SETUP* para realizar a conexão. Entretanto, o pedido de localização não contém o *Q.2931 SETUP*, pois tem a função somente de verificar a localização do móvel destino. Deve ser lembrado que, para esse pedido de localização atingir o domínio do móvel, ele pode ter passado por vários nós da rede, e não é sabido se esses podem comportar a qualidade de conexão necessária para essa chamada.

Quando o MSP do móvel de origem recebe a localização do alvo da chamada, ele envia, inicialmente, o pedido de conexão para a estação base do terminal que fez a requisição. A estação base irá verificar a disponibilidade dos recursos necessários para a ligação; se não for possível, o móvel receberá uma mensagem indicando isso.

Sendo a estação base capaz de suportar os recursos solicitados, o MSP encaminha a mensagem de conexão, contendo o *Q.2931 SETUP* para o MSP do terminal chamado. Se qualquer nó no caminho para o MSP destino, ou mesmo a estação base do móvel chamado, não concordar com os recursos pedidos para a conexão, ela não será estabelecida e o móvel de origem será informado.

Essa verificação junto á estação base permite um policiamento da funcionalidade do terminal, implementado implicitamente pela alocação de largura de faixa da sua estação base. Isso também protege o MSP de casos não típicos onde, apesar de o MSP esperar uma disponibilidade de recursos do rádio, a unidade de rádio está esgotada devido a cabeçalhos adicionais da camada MAC, ou uma redução temporária na qualidade do enlace de rádio

Antes de a conexão ser realizada, é verificado, no terminal de destino, se o terminal de origem não compõe sua lista de bloqueio - (*block list*). Caso isso ocorra, o móvel que realizou o pedido de conexão recebe uma mensagem informando que a ligação foi recusada. Quando a conexão é completada, os terminais estão prontos para trocarem informações.

3.5 Conclusão

Esse capítulo apresentou a arquitetura da rede WATM sem fio a ser especificada no capítulo 4 e simulada no capítulo 5. Foram descritos os dispositivos que compõem a rede, assim como suas funções. Na seção 3.4 foram apresentadas as descrições das sinalizações de Registro e Conexão de um terminal na rede WATM. Partindo dessa estrutura, no capítulo 4 os elementos da rede são especificados em SDL (*Specification and Description Language*) por blocos, onde cada um desses blocos exerce a função do dispositivo que ele representa.

Capítulo 4

Especificação em SDL'92 da Rede WATM

4.1 Introdução

Em 1968 o antigo CCITT[1] (*Comité Consultatif International Téléphonique et Télégraphique*) foi requisitado, por uma assembléia do seu plenário, para estudar o impacto da padronização de uma linguagem formal na comunidade de Telecomunicações. Essa questão foi passada para o grupo de estudos *Study Group XI, Switching and Signalling.* Durante os anos de 1967 a 1975 foram realizados estudos a respeito desse problema. Investigando-se como eram resolvidos nas empresas de telecomunicações a especificação e a descrição de sistemas, foi verificado que a maioria das empresas usava técnicas informais baseadas em eventos. Vários formalismos gráficos eram usados, a maioria deles consistindo da interconexão de nós por arcos [26].

No final do período de estudos, em 1976, a primeira Recomendação foi publicada, Rec. Z100, consistindo da padronização de alguns símbolos gráficos para representação de eventos e ações. Essa recomendação também definiu o nome da linguagem como SDL, *Specification and Description Language* [8].

Nos anos seguintes, outros símbolos foram incorporados à primeira versão e, outras recomendações surgiram, nos anos de 80, 84, 88 e 92, sendo que nessa última, versão SDL'92, foram adicionados conceitos de orientação à objetos, como tipos, reusos e herança.

Para as especificações realizadas nesse trabalho, utilizou-se o ambiente SDT (*SDL Design Tool*). Este permite a geração de diagramas de seqüência de mensagens MSC, apresentados no capítulo 5, que são muito úteis para a detecção de falhas durante a especificação.

Nesse capítulo, é apresentada a especificação da rede WATM utilizando a linguagem formal SDL em sua versão de 1992. Inicialmente a rede WATM, proposta no capítulo 3, é representada por meio de blocos em SDL. Cada bloco é explicado individualmente, em seções distintas desse capítulo. Os blocos são compostos por processos, e os processos por

[1] *Atual ITU-T - (International Telecommunication Union - Telecommunication Sector).*

procedimentos. Como há muitos processos e procedimentos, em cada processo foram escolhidos, para serem apresentados nesse capítulo, apenas os que possuem as especificações das sinalizações de Registro e Conexão. Os demais processos e procedimentos em SDL estão incluídos no apêndice B.

Dentro de cada bloco existem pelo menos dois processos. O único bloco com apenas um processo é o **MobileSwitchingPoint**. Em alguns blocos, apenas um dos processos está ligado à especificação da sinalização, tendo os outros processos a função de auxiliar o principal. Por isso, apenas o processo envolvido na especificação das sinalizações de Registro e Conexão é explicado.

Cada variável, sinal e estrutura utilizados na especificação completa estão definidos no apêndice A. Esse apêndice pode ser utilizado para consulta durante a leitura desse capítulo. Contudo, para que o acesso ao apêndice não se torne muito freqüente, alguns sinais utilizados têm sua explicação repetida nesse capítulo.

4.2 Especificação da rede WATM

Os componentes da rede WATM, definidos no capítulo 3, são representados em forma de blocos compondo o sistema WATM. A figura 4.1 mostra as duas páginas, **1(2)** e **2(2)**, que formam esse sistema.

Na página **1(2)** são apresentados, em forma de blocos, os elementos da rede WATM. O bloco **Mobile** representa os terminais móveis. O bloco **BaseStation** representa as estações base. O bloco **MobileSwitchingPoint** representa o MSP. E o bloco **DomainLocationService** representa o DLS.

Cada bloco está ligado por **rotas de sinais** representadas no diagrama por linhas com indicadores de direção nas pontas. Essa rotas podem ser uni ou bidirecionais, onde o sentido que o sinal trafega é representado pelo símbolo (>). Todos os sinais que trafegam em uma determinada direção da rota devem estar indicados dentro de colchetes ([ ]).

Para simplificar o gráfico, vários sinais que trafeguem no mesmo sentido de uma rota podem compor uma lista de sinais definida pelo comando *Signallist*. Na figura 4.1, página **1(2)**, o conjunto **L6** é composto pelos sinais *MSP_Begin*, *MSP_End* e *MSP_PId*. Na parte inferior da página **1(2)**, da figura 4.1, estão todas as listas de sinais definidas para esse sistema.

As rotas que ligam os blocos do sistema possuem um nome associado. Na figura 4.1, a rota que liga o bloco **DomainLocationService** com o bloco **MobileSwitchingPoint** é denominado **MSP_DLS**. A rota **MSP_DLS** transporta, no sentido do MSP para o DLS, o conjunto de sinais **L5**, **[(L5)]**. No sentido oposto, trafega o conjunto de sinais **L4**, **[(L4)]**.

A página **2(2)** da figura 4.1 contém a definição de todos os sinais trocados entre os blocos. Também são definidas nessa página as estruturas utilizadas no sistema. Quaisquer sinais, ou estruturas, definidos nessa página podem ser utilizados por qualquer bloco do sistema.

Os sinais são definidos abaixo do comando *Signal*. O sinal *Verify_blocking* transporta

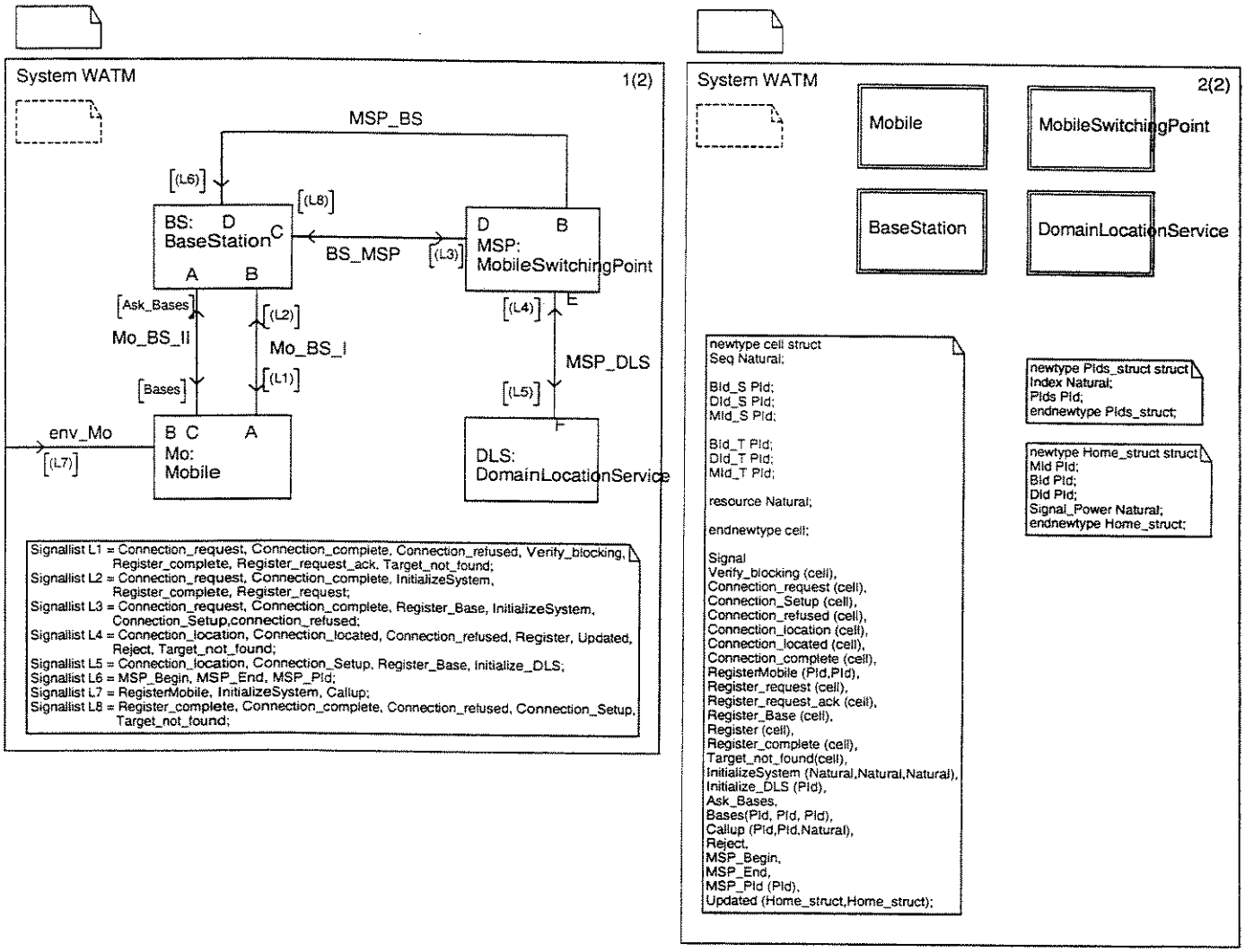

Figura 4.1: Sistema WATM

o tipo *cell*, *Verify_blocking (cell)*. O tipo *cell*, definido como um *newtype*, é uma estrutura formada pelos campos **Seq**, que é um número *Natural*; **BId_S**, **DId_S**, **MId_S**, **BId_T**, **DId_T** e **MId_T** que são do tipo PId[2]; e **resource** que é do tipo *Natural*.

4.3 Bloco Mobile

Na figura 4.2 é apresentado o conteúdo do bloco **Mobile**. Ele é composto por três processos: **Signal_Manager**; **Mobile_Id_Table**; e **Mobile**. Dos três processos que compõem o bloco, apenas o processo **Mobile** é instanciável.

O processo **Mobile** possui a seguinte indicação: **Mo_P (1,10):**. Isso indica que cada

[2]O tipo PId é reservado do SDL e identifica o número da instância de um processo.

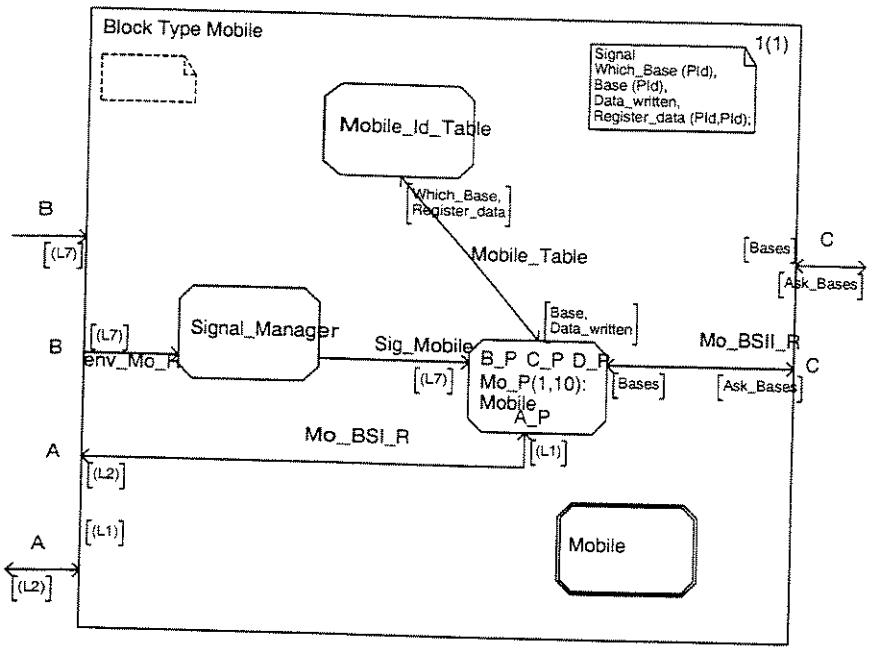

Figura 4.2: Bloco Mobile/Sistema WATM

instância desse processo será indicada pelo nome **Mo_P:x**, onde o **x** representa o número da instância. Isso será observado no capítulo 5 quando forem mostrados os exemplos de simulação. Os números entre parênteses **(1,10)** indicam que esse processo iniciará com uma instância, e poderá ter até dez.

Entretanto, o que é uma instância? Em um Sistema de Telecomunicações existem objetos que são logicamente equivalentes do ponto de vista de comportamento, por exemplo, todos os assinantes de um determinado tipo devem se comportar de acordo com um conjunto definido de procedimentos. Em um sistema real esse evento ocorre corriqueiramente; isso indica que existem várias instâncias desse comportamento. Todas as instâncias agem de acordo com um determinado conjunto de procedimentos previamente definidos. Entretanto isso não significa que em um dado momento duas dessas instâncias estejam realizando sempre a mesma tarefa, por exemplo, um usuário pode desligar o terminal enquanto outro, no mesmo instante de tempo, pode estar atendendo [26].

Todos os processos instanciáveis são representados por figuras geométricas desenhadas com linha dupla. Portanto, observando a figura 4.2, apenas o processo **Mobile** é instanciável.

Assim como é feito no sistema, em que os blocos trocam informações pelas rotas de sinais, os processos dos blocos também trocam informações pelo mesmo meio.

O processo **Signal_Manager** é responsável por direcionar qualquer sinal que venha do ambiente - **enviroment** para uma instância específica do processo **Mobile**. Os sinais proveniente do ambiente são os que dão início aos processos de Registro, Conexão, ou Inicialização do sistema. O processo **Signal_Manager** não faz parte das especificações das sinalizações, é um bloco auxiliar ao funcionamento do sistema WATM.

O processo **Mobile_Id_Table** relaciona, após o registro do terminal, cada instância do processo **Mobile** registrada com uma respectiva instância do processo **BaseStation**. Esse processo não pertence ás sinalizações, mas é necessário para que, após o registro do terminal, quando uma instância do processo **Mobile** for enviar alguma mensagem, possa identificar a qual instância do **BaseStation** a mensagem deve ser direcionada. O processo **Mobile_Id_Table** é apresentado no apêndice B.

4.3.1 Processo Mobile

O processo **Mobile** é capaz de executar as funções relacionadas em um terminal móvel. Nele é definida a especificação, das sinalizações de Registro e Conexão, por parte do móvel.

Na figura 4.3 são apresentadas as duas páginas, **1(2)** e **2(2)**, que formam o processo **Mobile**. Esse processo possui quatro procedimentos, dois na página **1(2)** : **Initialize-System** e **Callup**, e dois na página **2(2)** : **Register_complete** e **RegisterMobile**.

Na página **1(2)** do processo **Mobile**, figura 4.3, são definidas as variáveis utilizadas pelo processo. Elas são declaradas abaixo do comando *DCL*. Na frente de cada uma dessas variáveis é identificado o tipo a que elas pertencem, por exemplo, a variável **V1** é do tipo *Natural*.

Ao ser iniciado, esse procedimento atribui o valor inicial para a variável **New_Mobile** e entra no estado *Idle*. Todo processo, ou procedimento, só sairá de um estado com a chegada de um sinal. No processo da figura 4.3, para que ele saia do *Idle* é necessário que chegue um dos seguinte sinais: *Callup*; *InitializeSystem*; *Connection_request*; *Register_complete*; *Verify_blocking*; ou *Register_request_ack*. Se algum outro sinal, diferente dos listados, chegar enquanto o processo está em *Idle*, ele será propositalmente descartado.

Acompanhando algumas mensagens que chegam ao processo, existem os nomes de algumas variáveis, definidas no processo, entre parênteses. Isso significa que a mensagem transporta alguma informação para o processo. Essa informação ficará armazenada nas variáveis entre parênteses, por isso elas devem ser definidas com o mesmo tipo das variáveis que a mensagem carrega.

O mesmo se aplica às mensagens que são enviadas pelo processo, ou pelos procedimentos, que possuem variáveis entre parênteses. Nesse caso o conteúdo da variável será enviado pela mensagem.

Na página **2(2)** do processo **Mobile**, na figura 4.3, a mensagem *Verify_blocking* armazena o seu conteúdo na variável **Cell_WATM**, que é definida no processo como do tipo *cell*. Essa variável é posteriormente enviada como conteúdo da mensagem *Connection_complete*.

4.3.2 Especificação da sinalização de Registro para o terminal móvel

Conforme a figura 4.2, quando o móvel vai registrar-se, ele recebe do processo **Signal_Manager** a mensagem *RegisterMobile*. Essa mensagem informa a instância do **Mobile** a ser registrada, e a instância do processo **MobileSwitchingPoint** que será o domínio.

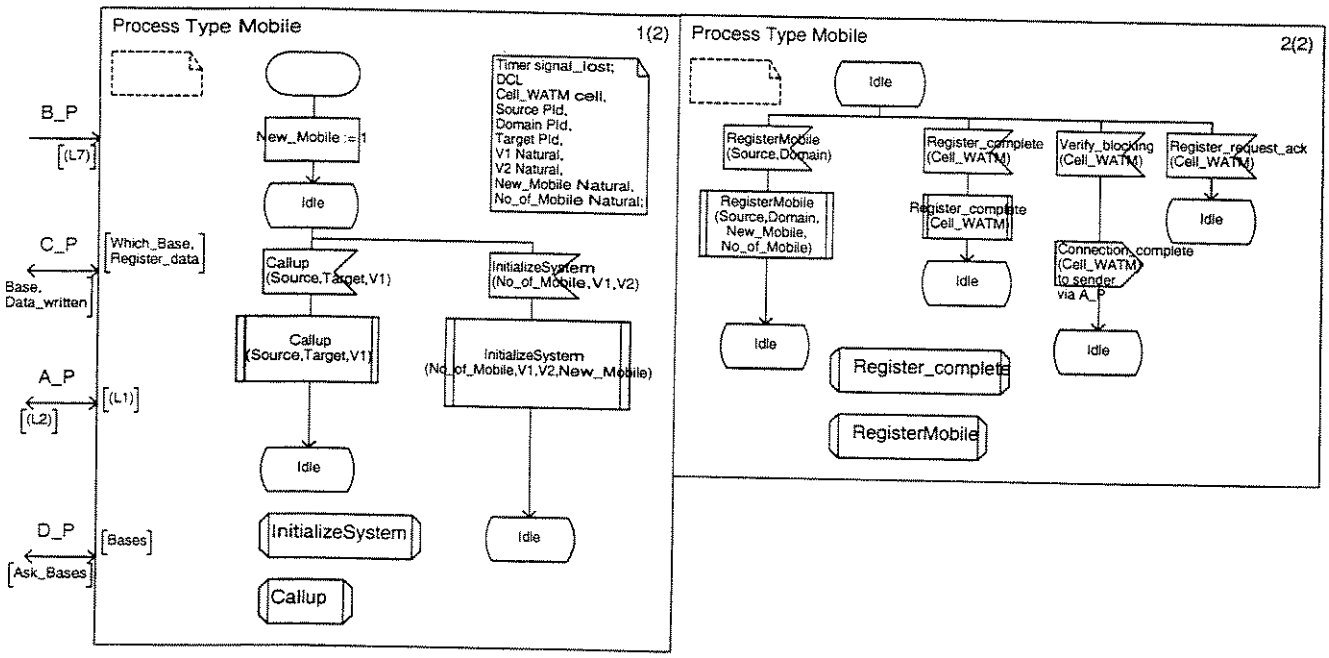

Figura 4.3: Processo Mobile/Bloco Mobile

Essa mensagem inicia o procedimento **RegisterMobile**, seção 4.3.2.1.

Após o móvel pedir o registro, as estações base, quando recebem essa sinalização, enviam uma mensagem, *Register_request_ack*, que avisa ao terminal que o pedido foi recebido. No procedimento **RegisterMobile** a mensagem *Register_request_ack* é esperada de alguma das estações base para onde foi enviado a requisição de registro. Como o pedido de registro é feito para três estações base distintas, três *Register_request_ack* são recebidos, um deles é consumido no procedimento **RegisterMobile** e os outros dois, para não serem perdidos, são consumidas no processo **RegisterMobile**. Após a mensagem *Register_request_ack* ser recebida, o processo retorna ao estado *Idle*.

O recebimento da mensagem *Register_complete*, que transporta a variável **Cell_WATM**, indica ao terminal que o registro foi realizado. Esse evento invoca o procedimento **Register_complete** a ser apresentado na seção 4.3.2.2.

4.3.2.1 Procedimento RegisterMobile

Na figura 4.4 é apresentado o procedimento responsável para iniciar o registro do móvel, o procedimento *RegisterMobile*. No canto superior da figura são determinadas as variáveis utilizadas nesse procedimento. Os parâmetros - *FPAR*[3] são usados para passar informações para o procedimento - **(IN)**, ou para tornar visível para o procedimento as variáveis do processo - **(IN/OUT)** . A variável **Source** é do tipo *PId*, e é um *FPAR IN*, enquanto a variável **New_Mobile** é do tipo *Natural*, e é um **FPAR IN/OUT**.

[3] *Formal Parameters.*

As variáveis locais a esse procedimento são declaradas abaixo do comando *DCL*. A variável **BS1**, visível apenas para o procedimento **RegisterMobile**, é do tipo *PId*.

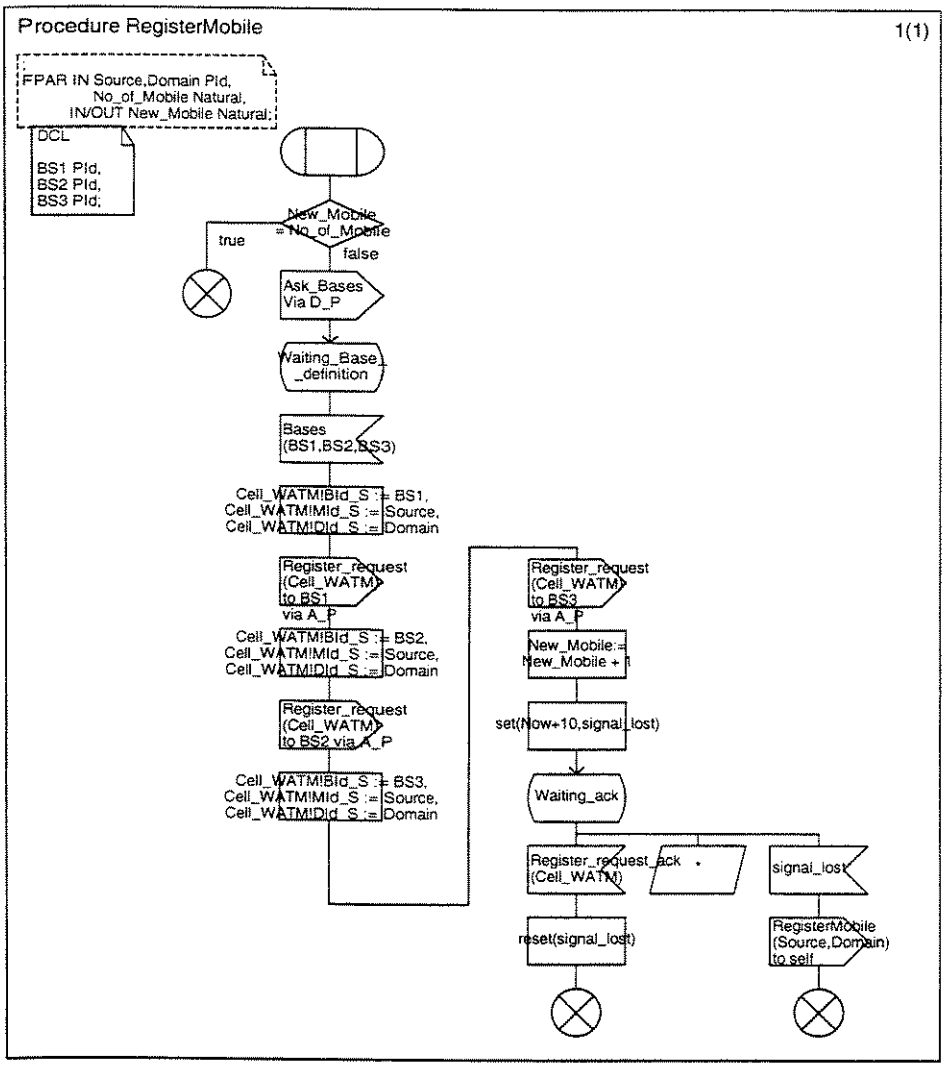

Figura 4.4: Procedimento RegisterMobile/Processo Mobile/Bloco Mobile

Quando um móvel deseja se registrar, é verificado se o número de móveis que o sistema comporta, **No_of_Mobile**, é igual ao número de móveis já registrados, **New_Mobile**. Se esse número for diferente, então o procedimento envia a mensagem *Ask_Bases*.

A mensagem *Ask_Bases* requisita do processo **Base_Id_Table** o PId de três instâncias do processo **BS_P**. O procedimento aguarda a resposta da mensagem enviada no estado *Waiting_Base_definition*, saindo desse estado somente com a chegada da mensagem *Bases*. A escolha dos PId's no processo **Base_Id_Table** é feita de modo aleatório, pois na prática um terminal móvel nem sempre registra-se na mesma estação base. O número de PId's escolhidos não influencia na especificação da sinalização de Registro. Apesar de ter sido determinado que o processo **Base_Id_Table** retornaria três PId's, essa quantidade pode

ser modificada.

A resposta, mensagem *Bases*, do procedimento **Base_Id_Table**, transporta o PId de três estações base distintas. Esses PId's são passados ao processo por intermédio das variáveis **BS1**, **BS2** e **BS3**.

O procedimento copia nos campos **BId_S**, **MId_S** e **DId_S**, da variável **Cell_WATM**, o PId da primeira estação base, do móvel e do domínio do móvel, respectivamente. Feito isso, esses valores são enviados para a estação base pela mensagem *Register_request*.

Os passos acima descritos para a primeira estação base são repetidos para as outras duas. Concluído esse conjunto de comandos, é iniciado um temporizador com o comando *set(Now + 10, signal_lost)*[4] e o processo entra em estado *Waiting_ack*, saindo deste apenas com a chegada das mensagens *Register_request_ack* ou *signal_lost*. Caso qualquer outra mensagem, diferente dessa duas, chegue ao processo **Mobile**, ela será armazenada em uma fila do tipo primeira que entra, primeira que sai (FIFO[5]) e será atendida apenas após o processo sair desse estado. O comando que salva as mensagens em uma fila FIFO é o paralelogramo com um asterisco inscrito, que representa o símbolo *SAVE* em SDL.

Se nenhum dos pedidos de registro chegar ao seu destino, o temporizador expirará, e o procedimento **RegisterMobile** será executado novamente até obter sucesso no envio da mensagem.

4.3.2.2 Procedimento Register_complete

Na figura 4.5 é apresentado o procedimento **Register_complete**. Nesse procedimento o móvel recebe a identificação de qual a estação base o está servindo, e armazena essa informação.

A mensagem *Register_data* é enviada para o processo **Mobile_Id_Table**, figura 4.2, com os dados do PId da instância **Mo_P**, e da instância **BS_P**. No processo **Mobile_Id_Table**, uma tabela armazena essa informação.

Após a mensagem ser enviada, o procedimento entra no estado *Waiting_writing_data*. Esse estado só será modificado com o recebimento da mensagem *Data_written*. Qualquer outra mensagem que chegar para essa instância do processo **Mobile** será guardada em uma fila FIFO.

4.3.3 Especificação da sinalização de Conexão para o terminal móvel

Quando um terminal deseja fazer uma conexão com outro, ele recebe do processo **Signal_Manager**, figura 4.2, a mensagem *Callup*. Essa mensagem é direcionada a instância do processo **Mobile** que irá originar a chamada.

---

[4] O número 10 de *NOW + 10* é do tipo *Duration*. A operação aritmédica de um tipo *Duration* com outro tipo qualquer, resulta em um tipo *Duration*. Nenhuma unidade é especificada em SDL para o tipo *Duration* [4].

[5] *First In First Out*.

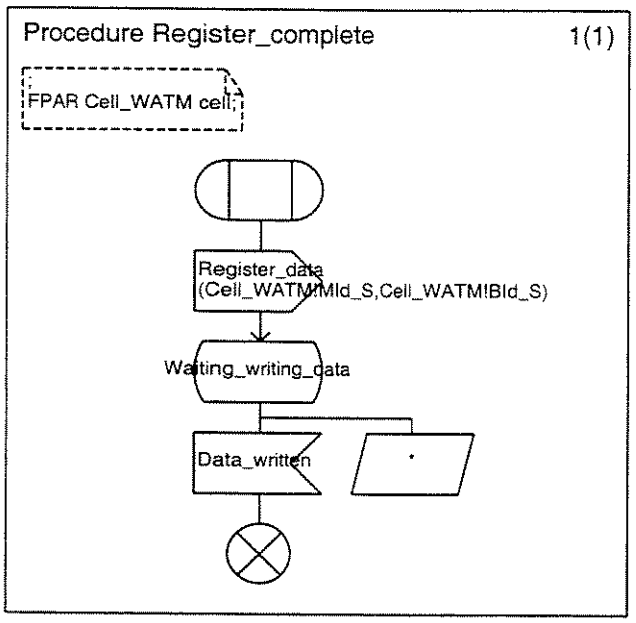

Figura 4.5: Procedimento Register_complete/Processo Mobile/Bloco Mobile

Essa mensagem transporta o PId da instância do móvel de origem, a instância do móvel destino, e a classe de QoS[6] para a conexão. Na página **1(2)** do processo **Mobile**, figura 4.3, a mensagem *Callup* é apresentada com os variáveis **(Source, Target, V1)** que recebem, respectivamente, os valores transportados. Após a mensagem ser recebida, é dado início ao procedimento **Callup**, que é mostrado na seção 4.3.3.1.

Quando o pedido de conexão é recebido pela estação base que cobre o móvel chamado, é enviado para esse terminal a mensagem *Verify_blocking*. O processo **Mobile** ao receber essa mensagem do processo **BaseStation**, figura 4.8, deveria verificar em uma tabela o PId da instância do processo **Mobile** que originou a chamada, para certificar se conexão para este terminal é permitida ou não. Nessa especificação considerou-se que a conexão sempre será aceita, contudo o sistema está pronto para, se for necessário, ser implementado uma tabela de PId's que determina quais terminais podem ter suas conexões aceitas. Essa tabela não está relacionada com a especificação das sinalizações de Registro ou Conexão.

4.3.3.1 Procedimento Callup

Esse procedimento é iniciado quando o terminal móvel deseja conectar-se a outro terminal. Esse procedimento é mostrado na figura 4.6.

No início desse procedimento, a variável **Cell_WATM**, que é definida como do tipo *Cell*, recebe em seus campos **MId_S**, **MId_T** e **resource**, o PId do terminal móvel que realizou a chamada, o PId do terminal que é o alvo da chamada e os recursos necessários para a chamada, respectivamente. O campo **resource** representa o QoS.

[6] A definição de QoS, e a tabela das classes, foram apresentadas no capítulo 2.

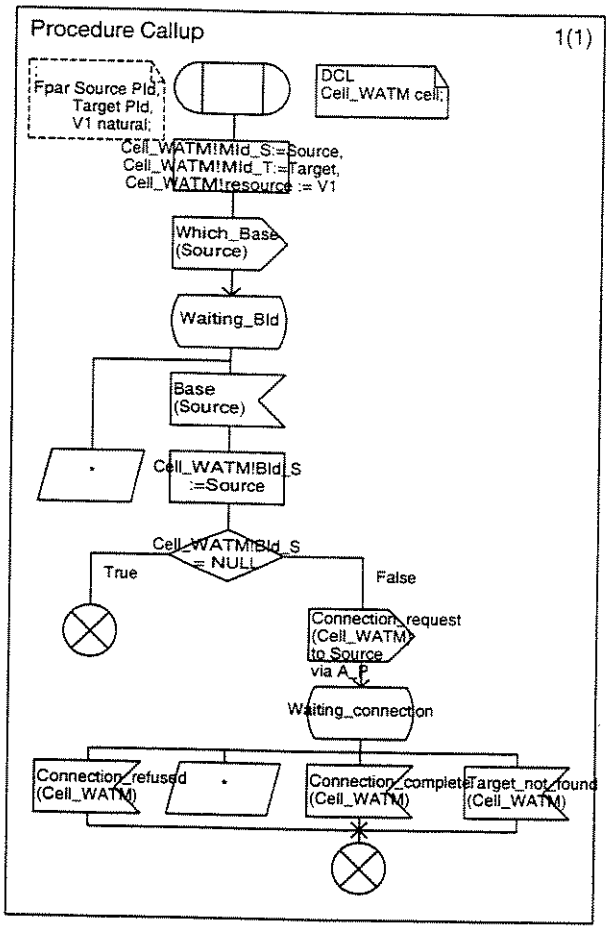

Figura 4.6: Procedimento Callup/Processo Mobile/Bloco Mobile

O procedimento envia a mensagem *Which_Base* para o procedimento **Mobile_Id_Table**, que pertence ao bloco **Mobile**, figura 4.2, e entra no estado *Waiting_BId* até que a mensagem *Base* seja recebida. Essas mensagens são internas desse bloco. Se alguma mensagem, exceto a mensagem *Base*, chegar a essa instância do processo **Mobile**, ela será guardada em uma fila FIFO.

A mensagem *Base* transporta o PId da instância do processo **BaseStation** para a qual a mensagem com o pedido de conexão deve ser enviado. Essa instância do processo **BaseStation** representa a estação base que assiste o móvel que deseja realizar a conexão.

O conteúdo da mensagem *Base* é passado para a variável **Source**. A variável **Source** copia, então, seu valor para o campo **BId_S** da variável **Cell_WATM**.

É verificado se o campo **BId_S**, da variável **Cell_WATM** é nulo. Se for verdade, implica que um móvel não registrado deseja fazer uma chamada, então o procedimento é finalizado e o pedido de conexão ignorado.

Caso o campo **BId_S** seja diferente de nulo, então é enviado, para a instância do processo **BaseStation** identificada na variável **Source**, a mensagem *Connection_request* carregando os valores da variável **Cell_WATM**. Após o envio dessa mensagem, a instância

desse processo entra em estado *Waiting_connection*.

Para sair do estado *Waiting_connection* é necessária a chegada de uma das mensagens: *Connection_refused*; *Connection_complete* e *Target_not_found*. Se alguma mensagem, diferente das esperadas pelo estado *Waiting_connection*, for recebida por essa instância do processo **Mobile** será guardada em uma fila FIFO.

A mensagem *Connection_refused* é recebida pelo processo **Mobile** quando a conexão for rejeitada. A mensagem *Target_not_found* indica ao processo que o terminal destino da conexão não foi encontrado. A mensagem *Connection_complete* é a última mensagem de uma conexão realizada com sucesso. Ela informa ao terminal que a conexão foi completada.

4.4 Bloco BaseStation

O bloco **BaseStation**, apresentado na figura 4.7, é composto por dois processos: **Base_Id_Table** e **BaseStation**.

O processo **BaseStation** é o único instanciável desse bloco. Ele pode ter até seis instâncias. Esse processo está ligado ao processo **Mobile** pela rota **Mo_BS_R**, e ao **MobileSwitchingPoint** pela rota **BS_MSP_R**. É nesse processo que estão especificadas as sinalizações de registro e conexão para uma estação base.

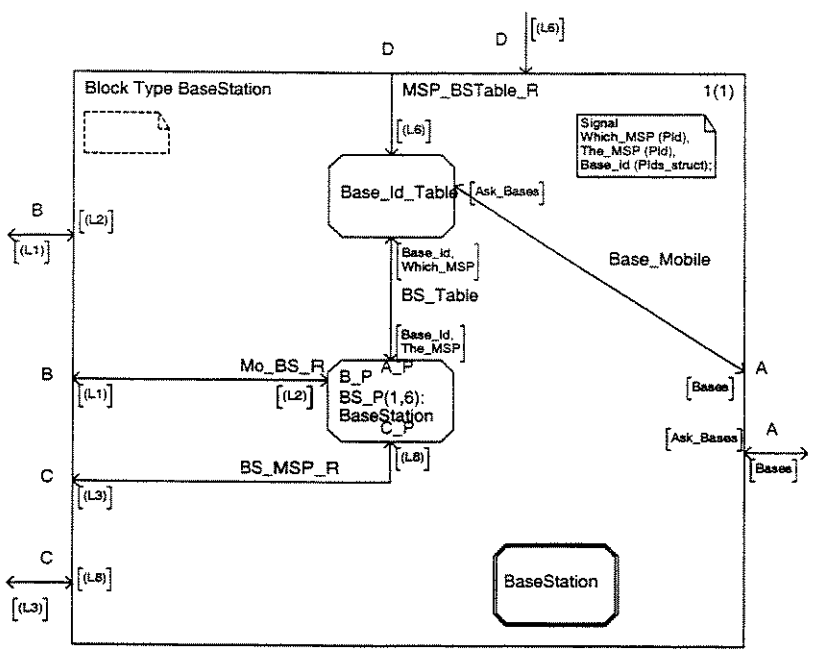

Figura 4.7: Bloco BaseStation/Sistema WATM

O processo **Base_Id_Table** tem função semelhante ao processo **Mobile_Id_Table** comentado na seção 4.3; esses dois processos são apresentados no apêndice B. Durante a inicialização do sistema, quando as estações base são divididas entre os MSP's, é

no processo **Base_Id_Table** onde ficam relacionadas as instâncias do processo **BaseStation** com as instâncias do processo **MobileSwitchingPoint**. Portanto, sempre que uma instância do processo **BaseStation** necessitar enviar uma mensagem para o processo **MobileSwitchingPoint**, terá que consultar o processo **Base_Id_Table** para obter a instância do processo **MobileSwitchingPoint**, à qual a mensagem será destinada.

Quando o terminal móvel deseja registrar-se, ele recebe o PId de três instâncias do processo **BaseStation**. Esses PId's são pedidos ao processo **Base_Id_Table**, pois este possui o PId de todas as instâncias do processo **BaseStation** que existem no sistema.

4.4.1 Processo BaseStation

O processo **BaseStation** é capaz de realizar as funções relacionadas a uma estação base. Nele é definida a especificação, das sinalizações de Registro e Conexão, por parte da estação.

Na figura 4.8 são apresentadas as duas páginas, **1(2)** e **2(2)**, que formam o processo **BaseStation**. Esse processo possui seis procedimentos, dois na página **1(2)** : **Register_request** e **InitializeSystemII**, e quatro na página **2(2)** : **Verify_blocking**; **Connection_request**; **Connection_Setup**; e **Connection_completeI**.

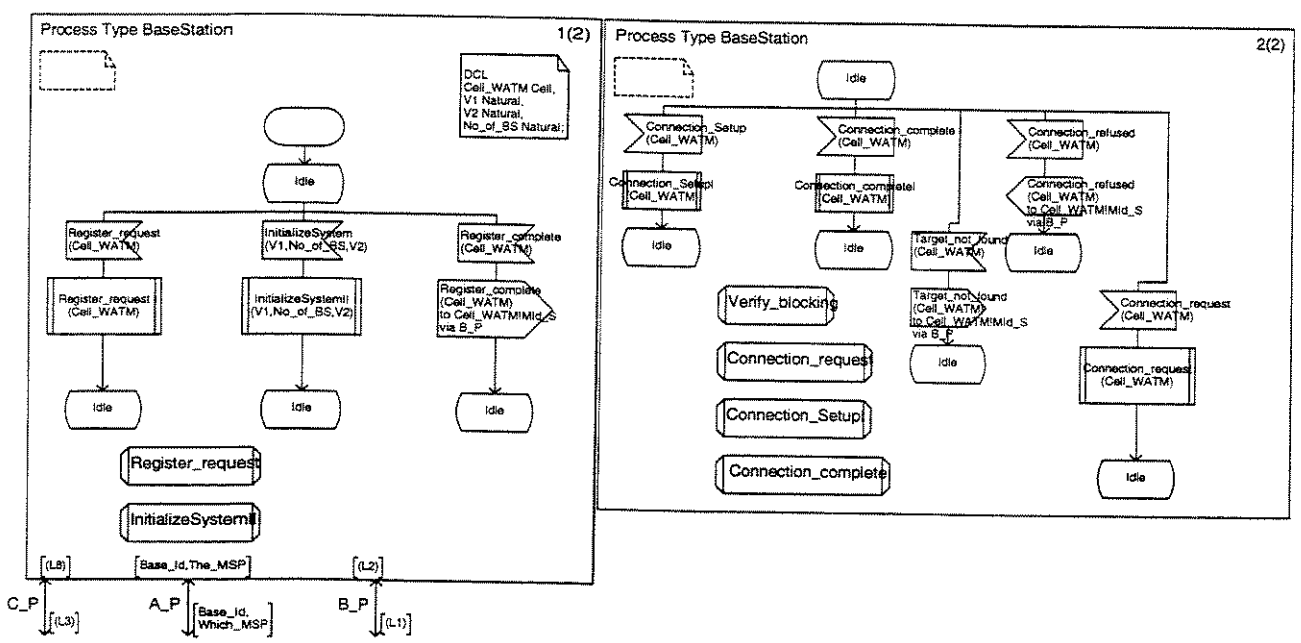

Figura 4.8: Processo BaseStation/Bloco BaseStaion

Esse processo inicia no estado *Idle*, e nele permanece até a chegada de uma das seguintes mensagens: *Connection_request*, *InicializeSystem* e *Register_complete*, na página **1(2)**; *Connection_Setup*, *Connection_complete*, *Target_not_found*, *Connection_refused* e *Connection_request*, na página **2(2)** .

4.4.2 Especificação da sinalização de Registro para a estação base

Quando o terminal móvel inicia seu registro, é enviada para a estação base a mensagem *Register_request*. Essa mensagem, recebida por alguma instância do processo **BaseStation**, executa o procedimento **Register_request**. A mensagem *Register_request* é a única envolvida, por parte da estação base, no registro do terminal.

4.4.2.1 Procedimento Register_request

No canto direito superior da figura 4.9, onde estão definidas as variáveis utilizadas nesse procedimento, é definido um subconjunto, *SmallNet*, do conjunto *Natural*. Os elementos do subconjunto são os números de 1 até 10. A criação do *SmallNet* é realizada com o uso do comando *Syntype*.

O procedimento, mostrado na figura 4.9, é iniciado com o recebimento, pelo processo **BaseStation**, da mensagem *Register_request*. A primeira ação desse procedimento é informar ao móvel que o pedido foi recebido, portanto é enviada a mensagem *register_request_ack*.

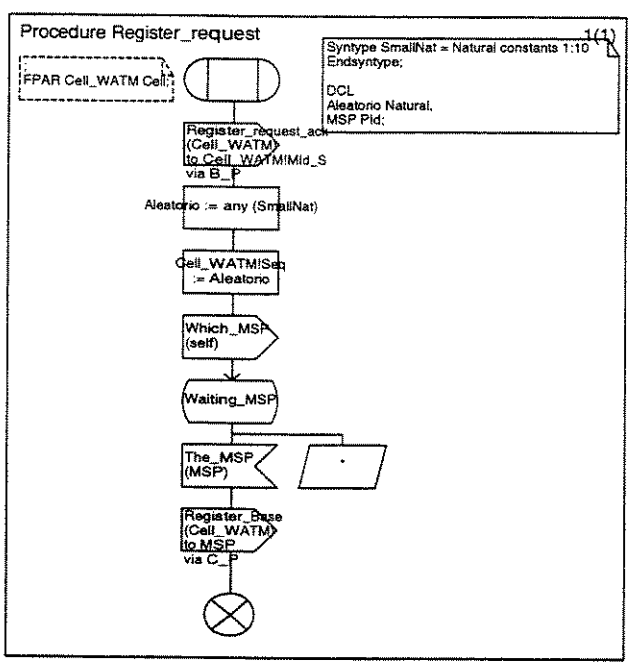

Figura 4.9: Procedimento Register_request/Processo BaseStation/Bloco BaseStation

Como não faz parte dessa especificação gerar um algoritmo para medir a intensidade de potência com que a estação base recebe o pedido de registro do móvel, é gerado um número aleatório, entre 1 e 10, para representar esse valor. O número randômico gerado é atribuído à variável **Aleatorio**, que posteriormente copia o conteúdo dela para o campo **Seq** da variável **Cell_WATM**.

Para esse procedimento saber qual instância do processo **MobileSwitchingPoint**, figura 4.15, essa instância deve enviar mensagens; é enviada a mensagem *Which_MSP* para o processo **Base_Id_Table** contendo o PId dessa instância do processo **BaseStation**. Feito isso, o procedimento vai para o estado *waiting_MSP*, saindo deste somente com a chegada da mensagem *The_MSP*. Se qualquer outra mensagem chegar a essa instância do processo **BaseStation**, ela será gravada em uma fila FIFO.

O processo **Base_Id_Table**, ao receber essa mensagem, verificará, em sua tabela, a qual PId do processo **MobileSwitchingPoint**, corresponde a instância do processo **BaseStation**. O resultado da pesquisa é enviado na mensagem *The_MSP*.

Quando o procedimento *Register_request* recebe a informação, pela mensagem *The_MSP*, de qual é o PId da instância do processo **MobileSwitchingPoint** que ele deve se comunicar, a mensagem *Register_Base* será enviada para essa instância. Todas as informações vindas do processo **Mobile**, figura 4.3, acrescido do número que representa a potência com que a estação base recebeu o pedido de registro do sinal, são enviadas pela mensagem *Register_Base*.

4.4.3 Especificação da sinalização de Conexão para a estação base

Quando um móvel deseja realizar uma conexão, ele envia a mensagem *Connection_request* para a instância do processo **BaseStation** definida para o terminal no momento do registro. Essa mensagem é recebida pela instância do processo **BaseStation**, página **2(2)** da figura 4.8, contendo a instância do processo **Mobile** que é destino da conexão. Todas essas informações são armazenadas na variável **Cell_WATM**. O recebimento da mensagem *Connection_request* inicia o procedimento **Connection_request** que será apresentado na seção 4.4.3.1.

Do pedido de conexão de um terminal podem-se ter três resultados: ou a conexão é completada; ou recusada; ou o terminal destino não é encontrado.

Quando o terminal destino não é encontrado, a instância, do processo **BaseStation**, que troca mensagens com a instância, do processo **Mobile**, que fez o pedido de conexão, recebe a mensagem *Target_not_found*. Essa mensagem é repassada, assim que recebida, para a instância do **Mobile** que pediu a chamada. No capítulo 5, seção 5.2.1, é apresentado um exemplo de simulação com uso dessa mensagem.

Se o pedido de conexão for recusado, a instância, do processo **BaseStation**, que troca mensagens com a instância, do processo **Mobile**, que fez o pedido de conexão, recebe a mensagem *Connection_refused*. Essa mensagem é repassada, assim que recebida, para a instância do **Mobile** que pediu a chamada.

A mensagem *Connection_complete*, que confirma a aceitação da conexão, pode ser recebida tanto da instância destino do processo **Mobile**, quanto da instância do processo **MobileSwitchingPoint** que troca informações com a instância do processo **BaseStation** que recebeu a mensagem com o pedido de conexão. A recepção da *Connection_complete* inicia o procedimento **Connection_completeI**, mostrado na seção 4.4.3.2.

Durante a conexão de um terminal, as estações base envolvidas são questionadas a

aceitar, ou rejeitar, a conexão avaliando a classe QoS pedida para a chamada. Para isso, a instância do processo **MobileSwitchingPoint** que estiver conectada à estação base de origem, ou à estação base de destino, envia a mensagem *Connecion_Setup*. Essa mensagem é recebida tanto pela estação base de origem, enviada pelo MSP de origem, quanto pela estação base destino, enviada pelo MSP destino. A chegada dessa mensagem em uma instância do processo **BaseStation** invoca o procedimento **Connection_SetupI**, que é abordado na seção 4.4.3.3.

4.4.3.1 Procedimento Connection_request

Esse procedimento, apresentado na figura 4.10, é iniciado quando o processo **BaseStation**, figura 4.8, recebe a mensagem *Connection_request*.

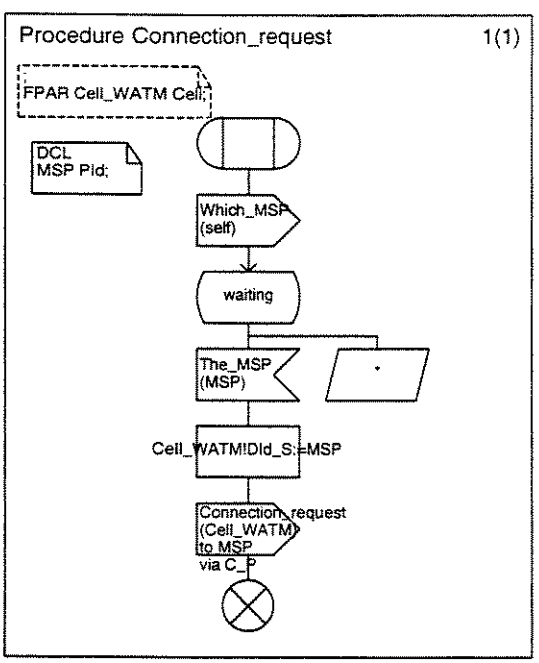

Figura 4.10: Procedimento Connection_request/Processo BaseStation/Bloco BaseStation

No seu início, esse procedimento envia a mensagem *Which_MSP*, e vai para o estado *waiting*. Essa instância do processo **BaseStation** permanece, nesse estado, até o recebimento da mensagem *The_MSP*. Caso qualquer outra mensagem, diferente da esperada, chegue, ela será guardada em uma fila FIFO.

Quando esse procedimento recebe a resposta da mensagem *Which_MSP*, por intermédio da variável **MSP**, ele passa esse valor para o campo **DId_S** da variável **Cell_WATM**.

A mensagem *Connection_request* é então enviada para a instância do processo **MobileSwitchingPoint** transportando o conteúdo da variável **Cell_WATM**.

4.4.3.2 Procedimento Connection_completeI

A mensagem *Connection_complete* pode ter sido proveniente tanto de uma instância do processo **Mobile**, quanto de uma instância do processo **MobileSwitchingPoint**. Se a mensagem tiver sua origem em uma instância do processo **Mobile**, esta instância, do processo **Mobile**, representa o móvel destino da chamada, conforme comentado na seção 4.4.3.

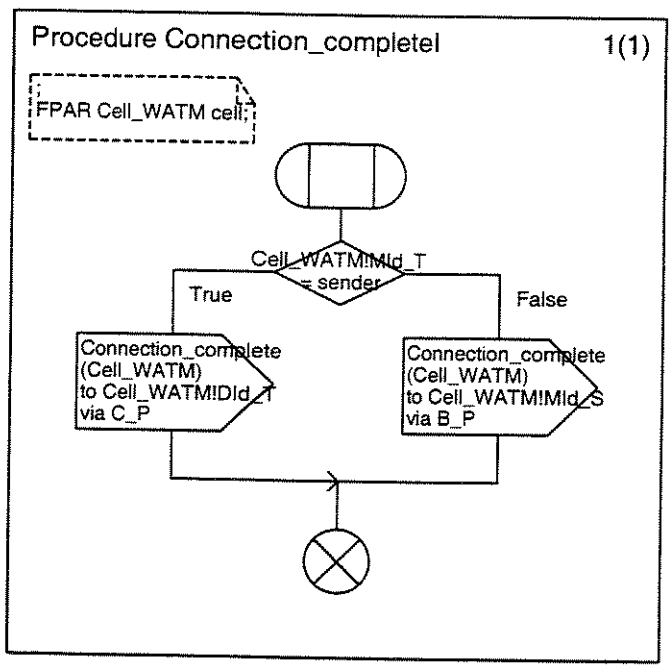

Figura 4.11: Procedimento Connection_completeI/Processo BaseStation/Bloco BaseStation

Ao ser iniciado, esse procedimento, mostrado na figura 4.11 verifica o emissor da mensagem *Connection_complete*. Se tiver sido uma instância do processo **Mobile**, a mensagem **Connection_complete** é enviada para a instância do processo **MobileSwitchingPoint** que representa o MSP do móvel destino da chamada.

Se a mensagem não tiver vindo do processo **Mobile**, obrigatoriamente sua origem é o processo **MobileSwitchingPoint** que representa o MSP do móvel que originou o pedido de chamada, então a mensagem *Connection_complete* é enviada a instância do processo **Mobile** equivalente ao terminal que requereu a conexão.

4.4.3.3 Procedimento Connection_SetupI

Esse procedimento, apresentado na figura 4.12, verifica se a estação base é capaz de suportar a classe QoS requisitada pela conexão. Arbitrariamente, foi definido que a estação base só suporta uma conexão cuja Classe QoS é menor que **3**.

O procedimento inicia comparando se o recurso da conexão é maior ou igual a **3**. Se for, é verificado se a mensagem *Connection_Setup* foi enviada pela instância do pro-

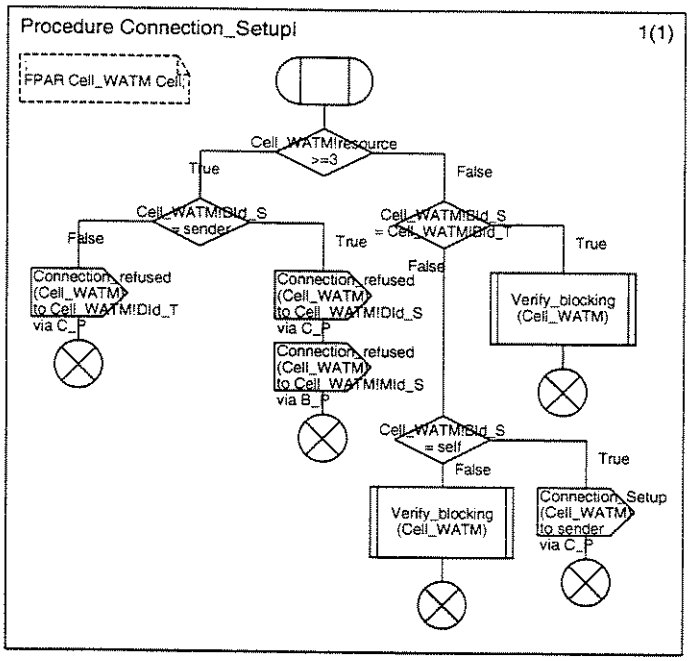

Figura 4.12: Procedimento Connection_SetupI/Processo BaseStation/Bloco BaseStation

cesso **MobileSwitchingPoint**, figura 4.15, que representa o MSP que atende ao móvel originador da chamada (MSP origem).

Na verificação da origem da mensagem *Connection_Setup*, se ela for oriunda da instância do processo **MobileSwitchingPoint** que representa o MSP origem, é enviado para essa instância do processo **MobileSwitchingPoint**, e depois para a instância do processo **Mobile**, figura 4.3, que pediu a conexão, a mensagem *Connection_refused*.

Entretanto, caso a mensagem *Connection_request* não tenha sido originada do MSP origem, ela terá sido envida do processo do MSP destino, pois apenas esses dois MSP's enviam a mensagem *Connection_Setup* para as estações base. Quando isso ocorre, a mensagem *Connection_refused* é enviada para a instância do processo **MobileSwitchingPoint** que atua como o MSP destino.

Para o caso da estação base aprovar a Classe QoS pedida para a conexão, isto é, o recurso pedido ser menor que **3**, o procedimento verifica se os dois terminais, origem e destino, são atendidos pela mesma instância do processo **BaseStation**, figura 4.8. Se for, é iniciado o procedimento *Verify_blocking*, que será apresentado na seção 4.4.3.4.

Contudo, se os terminais não forem atendidos pela mesma instância do processo **BaseStation**, o procedimento **Connection_SetupI** verifica se ele pertence a instância do processo **BaseStation** que troca informações com o móvel origem. Se for, a mensagem *Connection_Setup* é enviada para o MSP origem. Caso contrário, é iniciado o procedimento **Verify_blocking**.

4.4.3.4 Procedimento Verify_blocking

Esse procedimento, apresentado na figura 4.13, envia para a instância do processo **Mobile**, que representa o terminal destino da conexão, a mensagem *Verify_blocking* . Após a mensagem ser enviada, o procedimento **Verify_blocking** vai para o estado *waiting_blocking.*

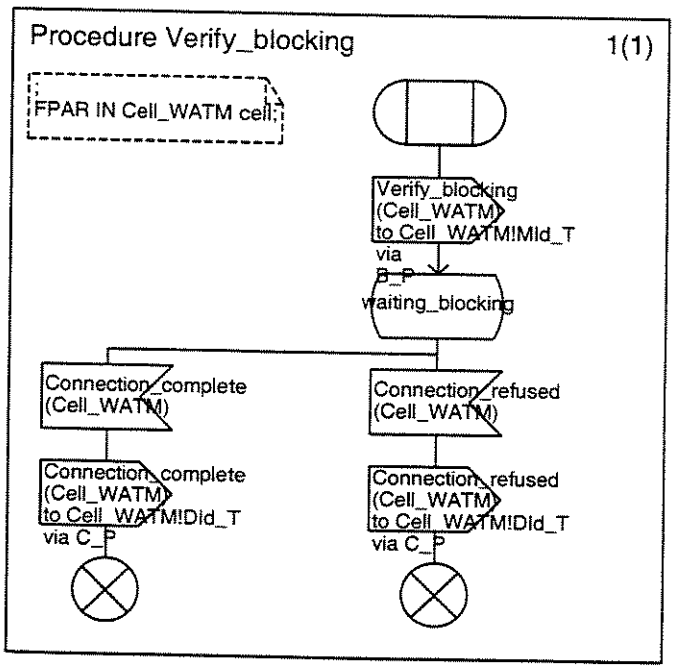

Figura 4.13: Procedimento Verify_blocking/Processo BaseStation/Bloco BaseStation

Para o procedimento sair do estado *waiting_blocking* é necessário que seja enviado para a instância do processo **BaseStation**, onde o procedimento está atuando, uma das duas mensagens: *Connection_complete* ou *Connection_refused.*

Independentemente de qual das duas mensagens chegue ao procedimento, a atitude será a mesma: enviar a mensagem recebida à instância do processo **MobileSwitchingPoint**, que atua como MSP destino.

4.5 Bloco MobileSwitchingPoint

Na figura 4.14 é apresentado o conteúdo do bloco **MobileSwitchingPoint**. Ele é formado por apenas um processo, que possui o mesmo nome do bloco, processo **MobileSwitchingPoint**. Esse processo é instanciável, podendo ter até cinco instâncias, **MSP_P (1,5)**.

4.5.1 Processo MobileSwitchingPoint

Ao processo **MobileSwitchingPoint** são atribuídas as funções executadas pelo MSP. Nele é definida a especificação das sinalizações de Registro e Conexão, por parte do MSP.

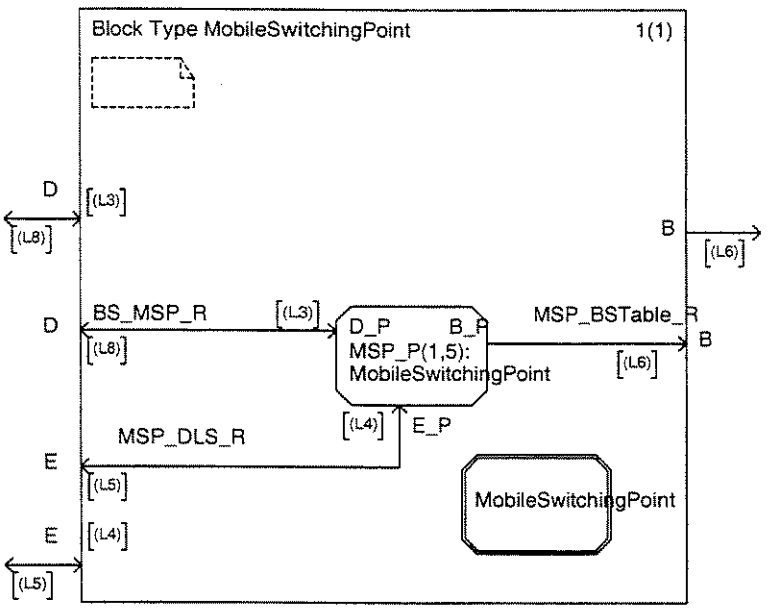

Figura 4.14: Bloco MobileSwitchingPoint/Sistema WATM

Na figura 4.15 são apresentadas as duas páginas, **1(2)** e **2(2)**, que formam o processo **MobileSwitchingPoint**. Esse processo possui seis procedimentos, três na página **1(2)**: **Update**; **Connection_refused**; e **InicializeSystem**, e três na página **2(2)**: **Connection_complete**; **Connection_located**; e **Connection_Setup**.

O processo **MobileSwitchingPoint** inicia no estado *Idle*, e nele permanece até que uma das seguintes mensagens seja recebida pelo processo: *Reject*; *InitializeSystem*; *Register_Base*; *Register*; *Connection_refused* e *Updated*, na página **1(2)**, e *Connection_request*; *Connection_located*, *Target_not_found*, *Connection_complete* e *Connection_Setup*, na página **2(2)**.

4.5.2 Especificação da sinalização de Registro para o MSP

O pedido de registro de um terminal, é recebido no processo **MobileSwitchingPoint**, na página **1(2)** da figura 4.15, por intermédio da mensagem, *Register_Base*, que foi enviada do **BaseStation**, figura 4.8. O processo **MobileSwitchingPoint**, ao obter a mensagem *Register_Base*, transmite-a para o processo **DomainLocationService**, seção 4.6.1.

A resposta da mensagem *Register_Base* poderá ser uma das mensagens: *Reject*, página**1(2)**; *Register* ou *Updated*, página **2(2)**. Se a resposta por a mensagem *Reject*, o processo **MobileSwitchingPoint** a recebe e volta para o estado *Idle*. A mensagem *Reject* é enviada ao MSP quando os dados do registro do móvel provenientes, do próprio MSP, não foram aceitos. Como o registro do terminal, nessa rede, sempre vai ocorrer, com os dados de uma das três estações base sorteadas, não é necessário que a mensagem *Reject* seja enviada para o móvel.

UNICAMP
BIBLIOTECA CENTRAL

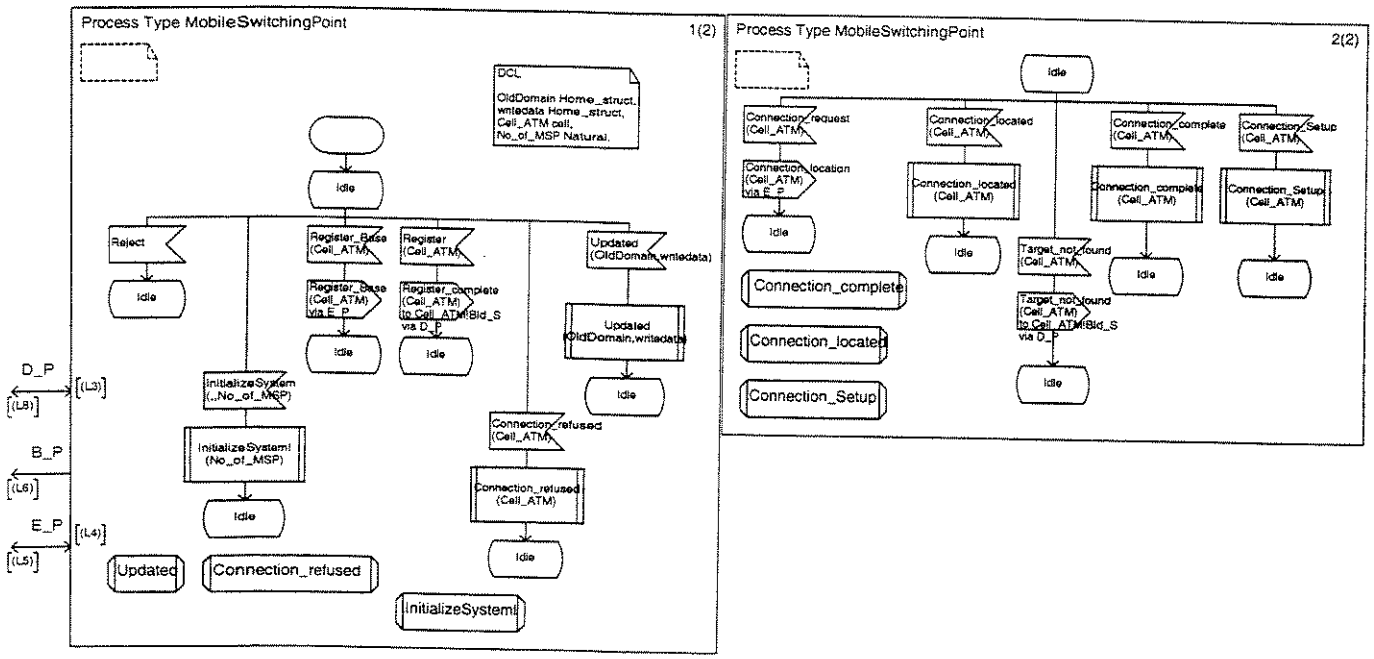

Figura 4.15: Processo MobileSwitchingPoint/Bloco MobileSwitchingPoint

Caso a resposta da mensagem *Register_Base* for a mensagem *Register*, é enviada ao processo **BaseStation**, a mensagem *Register_complete*. Entretanto, se a resposta for a mensagem *Updated*, será iniciado o procedimento **Updated**. Esse procedimento é abordado na seção 4.5.2.1. A mensagem *Updated* transporta os dados antigos de registro do móvel, variável **OldDomain**, e os dados atualizados, variável **writedata**.

4.5.2.1 Procedimento Updated

Na figura 4.16 é apresentado o procedimento **Updated**.

Ao ser iniciado, esse procedimento atribui o valor dos campos **MId**, **BId** e **DId**, da variável **writedata**, para os respectivos campos **MId_S**, **BId_S** e **DId_S**, da variável **Cell_ATM**. Com isso, os valores da variável **Cell_ATM** são atualizados.

Os dados atualizados são enviados para a instância do processo **BaseStation** que teve seus dados substituídos durante o registro do terminal.

4.5.3 Especificação da sinalização de Conexão para o MSP

O pedido de conexão é recebido pelo processo **MobileSwitchingPoint** por intermédio da mensagem *Connection_request*, página **2(2)** da figura 4.15. A chegada dessa mensagem faz com que o processo **MobileSwitchingPoint** envie para o processo **DomainLocationService** a mensagem *Connection_location*. A resposta, do processo **DomainLocationService**, à mensagem *Connection_location* é uma das seguintes mensagens: *Connection_located* ou *Target_not_found*, ambas na página **2(2)** da figura 4.15.

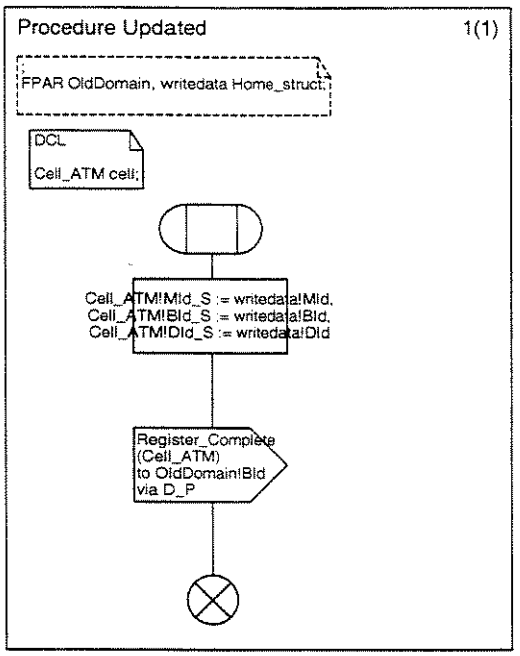

Figura 4.16: Procedimento Updated/Processo MobileSwitchingPoint/Bloco MobileSwitchingPoint

A mensagem *Target_not_found* indica que o terminal destino não foi encontrado. Ao ser recebida pelo processo **MobileSwitchingPoint**, a mensagem é enviada para a instância do processo **BaseStation** que enviou o pedido de conexão.

A mensagem *Connection_located* informa a localização, MSP e estação base, do terminal destino da conexão. O processo **MobileSwitchingPoint** quando obtiver a mensagem *Connection_located*, inicia o procedimento **Connection_located**, seção 4.5.3.1.

Durante a conexão do terminal, o MSP destino verifica se há possibilidade de atender os recursos necessários para a conexão. Essa verificação é realizada por intermédio da mensagem *Connection_Setup*. No MSP origem, essa verificação é realizada no procedimentos **Connection_located**. A recepção da mensagem *Connection_Setup*, pelo processo **MobileSwitchingPoint**, inicia o procedimento **Connection_Setup**, seção 4.5.2.1.

Caso o pedido de conexão seja recusado, é enviado, para o processo **MobileSwitchingPoint**, a mensagem *Connection_refused*. O recebimento dessa mensagem inicia o processo **Connection_refused**, seção 4.5.3.3.

Quando a conexão é aceita, a mensagem *Connection_complete* é enviada pelo processo **MobileSwitchingPoint**. A chegada dessa mensagem executa o processo **Connection_complete**, seção 4.5.3.4.

4.5.3.1 Procedimento Connection_located

Quando esse procedimento é iniciado, figura 4.17, já é conhecida a localização do terminal destino da chamada. Entretanto, falta realizar os testes, nos dispositivos que compõem

essa conexão, para a aceitação do QoS necessário para a chamada. Esse procedimento, **Connection_located**, é invocado apenas pela instância do processo **MobileSwitchingPoint** que representa o MSP que atende ao terminal que originou a chamada.

O algoritmo para a determinação do QoS dos elementos da rede está fora do escopo desse trabalho, por isso para os dispositivos MSP's foi fixado o valor **5** figura 4.17.

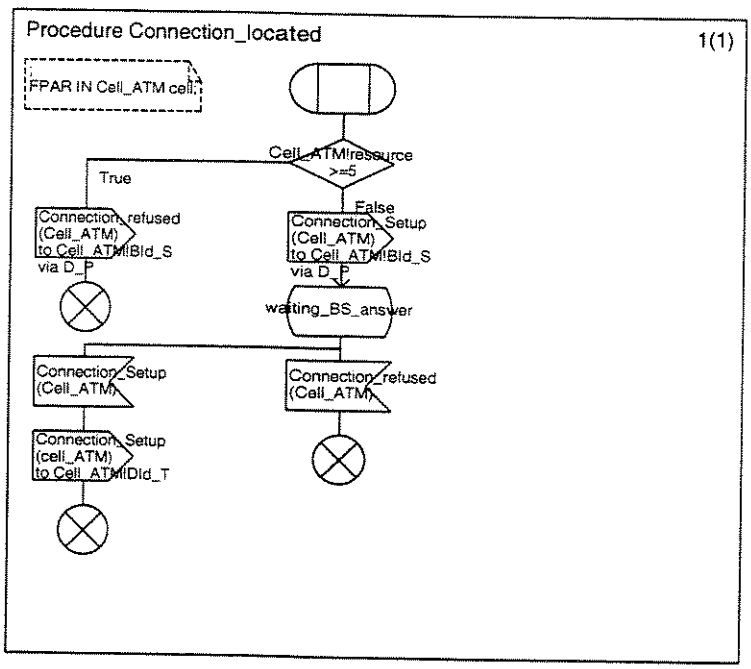

Figura 4.17: Procedimento Connection_located/Processo MobilSwitchingPoint/Bloco MobileSwitchingPoint

É comparado, nesse procedimento, figura 4.17, se o recurso pedido para a conexão é maior ou igual a **5**. Se for, a mensagem *Connection_refused* é enviada para a instância do processo **BaseStation**, que representa a estação base que cobre o terminal que originou a conexão - (estação base origem). Caso contrário é enviada para a instância do processo **BaseStation**, que representa a estação base origem, a mensagem *Connection_Setup*. Isso é diferente do que é sugerido por [14], que propõe o envio dessa mensagem para a estação base origem apenas após a conexão ser completada. Essa comparação entre as duas propostas foi apresentada no capítulo 3, seção 3.4.2. Após enviar a mensagem, o procedimento vai para o estado *waiting_BS_answer*.

Duas são as mensagens que podem ser recebidas nesse estado: *Connection_Setup* e *Connection_refused*. Caso a instância do processo **BaseStation**, que recebeu a mensagem, não suporte o recurso pedido, é enviada a mensagem *Connection_refused* e encerrado o procedimento.

Se a instância do processo **BaseStation** aceitar o recurso pedido, é enviada a mensagem *Connection_Setup* para a instância do processo **MobileSwitchingPoint**, que representa o MSP que atende ao móvel que originou a chamada. Ao receber essa mensagem, a instância do processo **MobileSwitchingPoint** a transmite para a instância do processo

MobileSwitchingPoint que representa o MSP que atende ao terminal chamado - (MSP destino).

4.5.3.2 Procedimento Connection_Setup

Esse procedimento, apresentado na figura 4.18, verifica se o recurso pedido para a conexão é aceito. Arbitrariamente foi assumido que qualquer valor menor que **5** seria aceito.

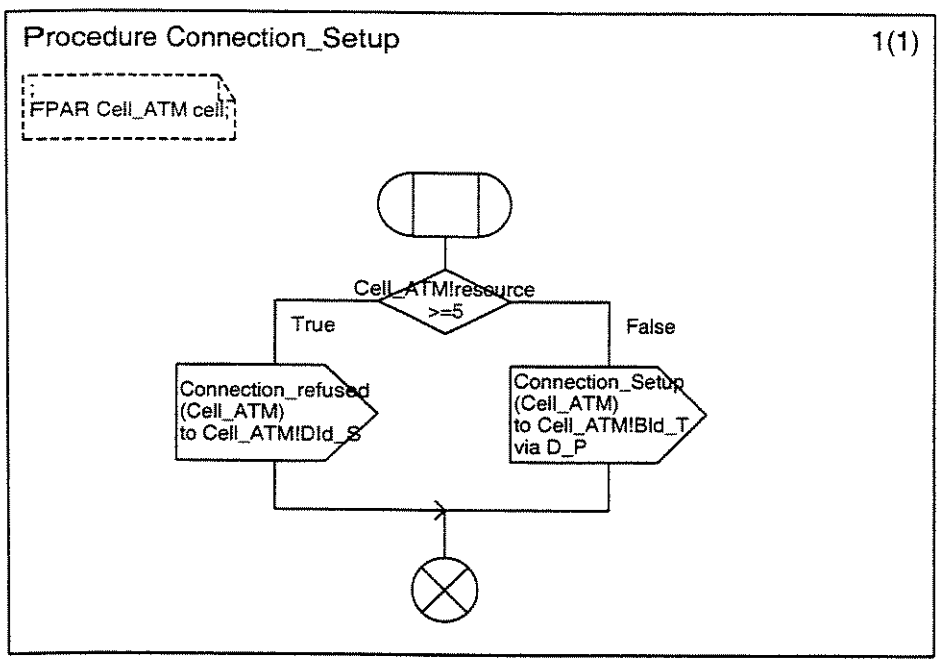

Figura 4.18: Procedimento Connection_Setup/Processo MobileSwitchingPoint/Bloco MobileSwitchingPoint

O procedimento inicia verificando o recurso pedido pela conexão. Se for maior ou igual a **5** a conexão é recusada, e a mensagem é enviada para o MSP que atende ao terminal que originou a chamada. Entretanto se o recurso for aceito, menor que **5**, a mensagem *Connection_Setup* é enviada para a instância do processo **BaseStation** que representa a estação base destino.

A verificação da aceitação, pelo MSP que atende ao terminal que originou a chamada, dos recursos pedido para a conexão, não é realizado neste procedimento, mas no procedimento **Connection_located**, figura 4.17.

4.5.3.3 Procedimento Connection_refused

A conexão pode ser recusada, quer pelos MSP's, quer pelas estações base, ou mesmo pelo móvel destino. A não aceitação por parte do MSP de origem, é feita no procedimento **Connection_located**, seção 4.5.3.1, e não envolve o envio da mensagem *Connection_refused* para o processo **MobileSwitchingPoint**. A recusa da conexão pelo terminal destino, é

enviada para a estação base, seção 4.4.3, e esta a envia para o processo **MobileSwitchingPoint**.

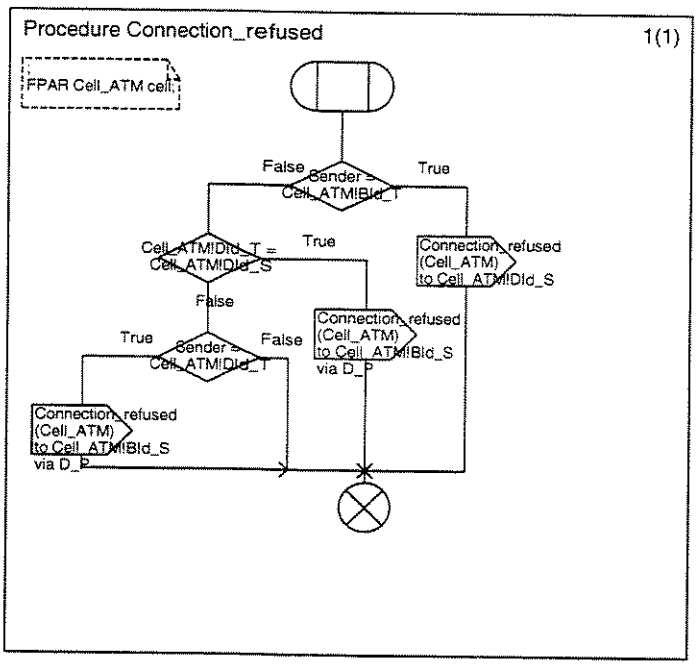

Figura 4.19: Procedimento Connection_refused/Processo MobileSwitchingPoint/Bloco MobileSwitchingPoint

O procedimento, mostrado na figura 4.19, verifica se a instância que enviou a mensagem *Connection_refused* foi a do processo **BaseStation**, figura 4.8, que representa a estação base destino. Se a mensagem tiver vindo dessa instância, a mensagem *Connection_refused* é enviada para a instância do processo **MobileSwitchingPoint** que atua como o MSP que serve o terminal que requereu a chamada.

Contudo, se a mensagem não tiver vindo de uma instância do processo **BaseStation**, então ela é proveniente do MSP. O procedimento **Connection_refused** só é executado quando a mensagem vem da estação base do terminal destino, ou de um MSP.

É verificado se as instâncias, do processo **MobileSwitchingPoint** que representam o MSP, são iguais. Se forem, isso implica que os dois terminais estão sob o mesmo MSP, então a mensagem *Connection_refused* indica que esse MSP, estando tanto como MSP origem quanto MSP destino, não consegue suportar essa conexão. Devido a isso, é enviada a mensagem *Connection_refused* para a instância do processo **BaseStation** que representa a estação base de origem.

Se os MSP's, destino e origem, forem diferentes, então é verificado se quem enviou a mensagem foi a instância do processo **MobileSwitchingPoint** que atua como MSP destino. Caso o resultado da verificação seja verdadeira, a mensagem *Connection_refused* é enviada para a instância do processo **BaseStation** que age como estação base origem.

Entretanto, se a mensagem não tiver vindo do MSP destino, ela é oriunda da instância do processo **BaseStation** que representa a estação base origem. Como nenhuma conexão

tinha sido realizada, o procedimento é encerrado e o processo **MobileSwitchingPoint** volta ao estado *Idle*.

Se a especificação seguisse o sugerido por [14], quando a instância do processo **MobileSwitchingPoint** (que representa o MSP origem) recebesse da instância do processo **BaseStation** (que representa a estação base origem) a mensagem *Connection_refused*, esta mensagem deveria ser repassada para o MSP destino, depois para a estação base destino e por fim para o terminal destino da conexão. Com a modificação, menos mensagens são trocadas quando a rejeição da conexão ocorre na estação base de origem.

4.5.3.4 Procedimento Connection_complete

Esse procedimento, apresentado na figura 4.20, verifica o destino da mensagem *Connection_complete*. Para isso é verificado se a mensagem *Connection_complete* foi recebida pela instância do processo **BaseStation**, que representa a estação base do terminal destino.

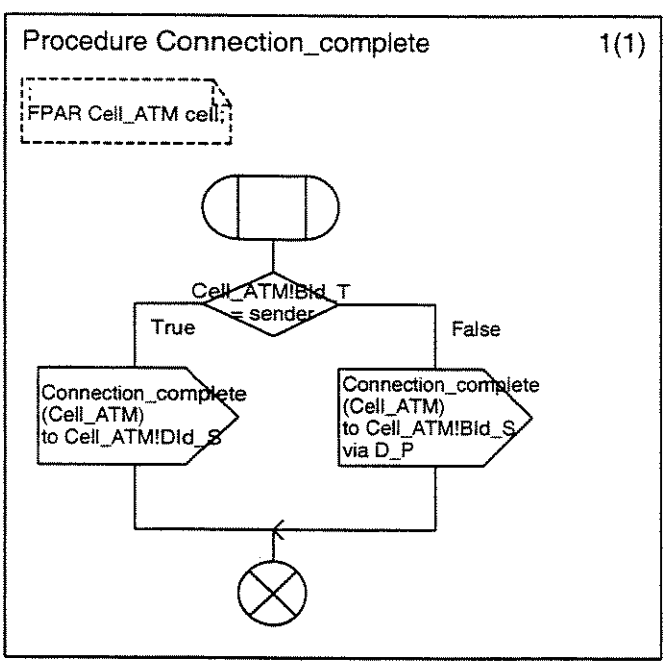

Figura 4.20: Procedimento Connection_complete/Processo MobileSwitchingPoin/Bloco MobileSwitchingPointt

Caso a mensagem tenha sido recebida da instância do processo **BaseStation**, então a instância do processo **MobileSwitchingPoint** que recebeu a mensagem representa o MSP destino, portanto a mensagem *Connection_complete* será enviada para o MSP origem.

Entretanto, se a mensagem não tiver vindo da instância do processo **BaseStation**, significa que a instância do processo **MobileSwitchingPoint** que recebeu a mensagem representa o MSP origem, então a mensagem *Connection_complete* será transmitida para a estação base origem.

4.6 Bloco DomainLocationService

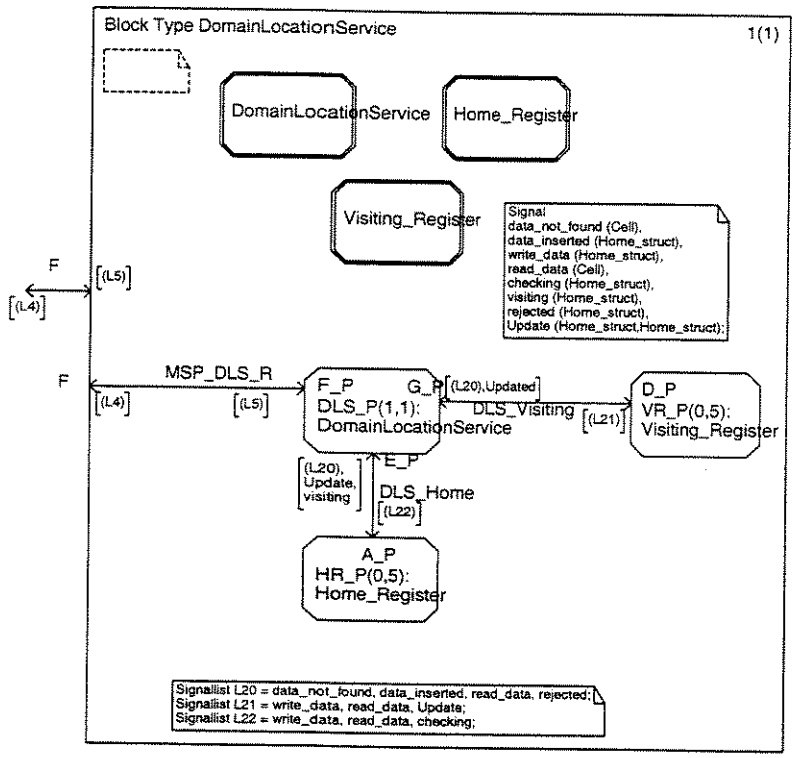

Figura 4.21: Bloco DomainLocationService/Sistema WATM

Na figura 4.21 é mostrado o bloco **DomainLocationService**. Este bloco é composto por três processos: **DomainLocationService**; **Home_Register**; e **Visiting_Register**. Dos três processos, os dois últimos são instanciáveis e possuem até cinco instâncias. O número de instâncias que cada bloco efetivamente terá depende do número de instâncias do processo **MobileSwitchingPoint** que compõe a rede. Para cada instância do processo **MobileSwitchingPoint**, existe uma instância do processo **Home_Register**, e uma do processo **Visiting_Register**. O processo **DomainLocationService** também é instanciável, contudo possui apenas uma instância.

4.6.1 Processo DomainLocationService

As duas páginas, **1(2)** e **2(2)**, que formam o processo **DomainLocationService**, são mostrados na figura 4.22. Esse processo é responsável por gerenciar todos os dados do processo **MobileSwitchingPoint**. O processo **DomainLocationService** é encarregado por gerenciar o registro dos terminais, localizar os terminais destino durante os pedidos de conexão, e informar se esse terminal foi encontrado ou não.

O processo **DomainLocationService** é composto por cinco procedimentos, todos mostrados na página **2(2)**, figura 4.22: **read_data**; **data_inserted**; **Register_Base**;

Initialize_DLS; e **data_not_found**.

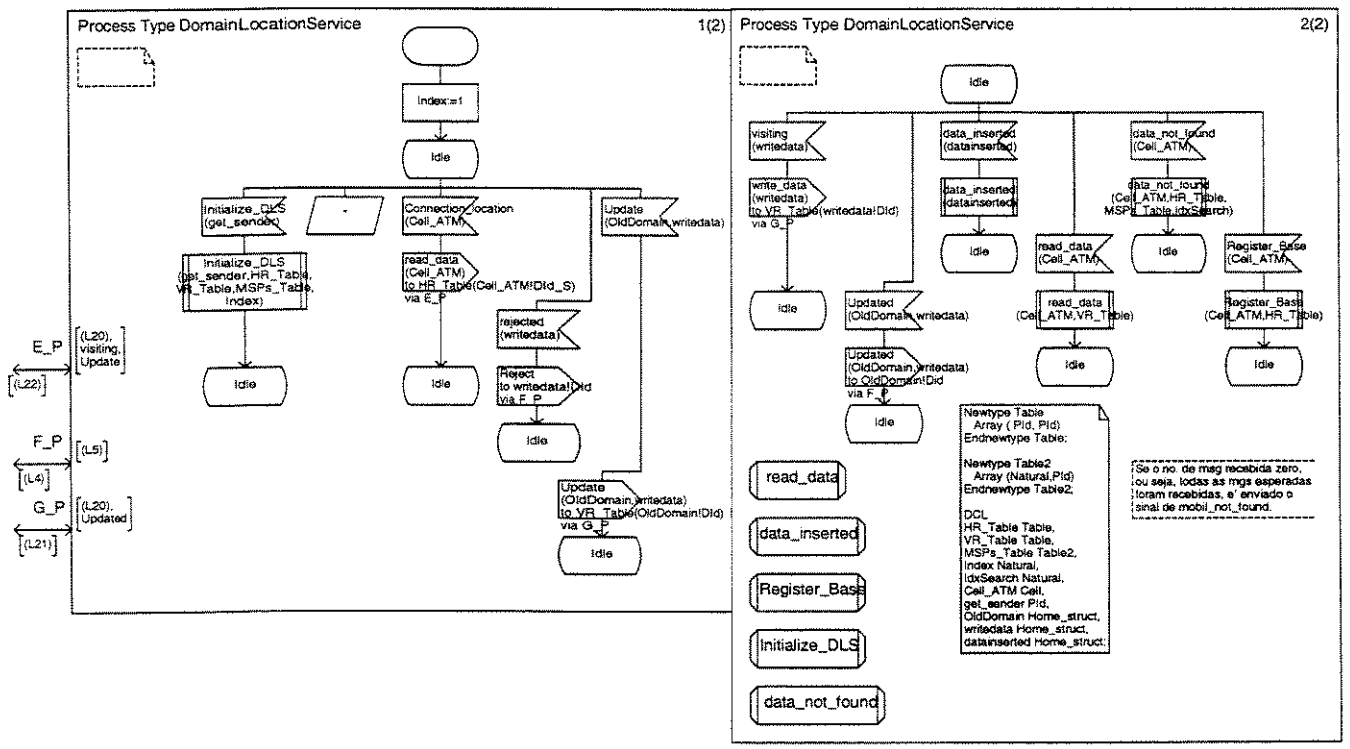

Figura 4.22: Processo DomainLocationService/Bloco DomainLocationService

4.6.2 Processo Home_Register

O processo **Home_Register**, apresentado na figura 4.23 é responsável por armazenar os dados dos terminais registrados que pertencem a um determinado MSP. Cada instância do processo **MobileSwitchingPoint** possui uma instância do processo **Home_Register** correspondente para armazenar os dados dos terminais que pertencem a instância do processo **MobileSwitchingPoint**. O processo **Home_Register** é composto por três procedimentos: **read_dataI**; **Checking**; e **write_data**. Nenhum desses três procedimentos é utilizado diretamente nas sinalizações de registro ou conexão. As funções desses procedimentos são de acesso ao banco de dados.

4.6.3 Processo Visiting_Register

O processo **Visiting_Register**, mostrado na figura 4.24, é responsável por armazenar os dados dos terminais, que se registraram em um determinado MSP como visitante. O processo **Visiting_Register** é composto por três procedimentos: **Update**; **write_data**; e **read_dataII**. As funções do processo **Visiting_Register**, e do processo **Home_Register**, são bastante semelhantes. Contudo, um trata dos dados dos terminais

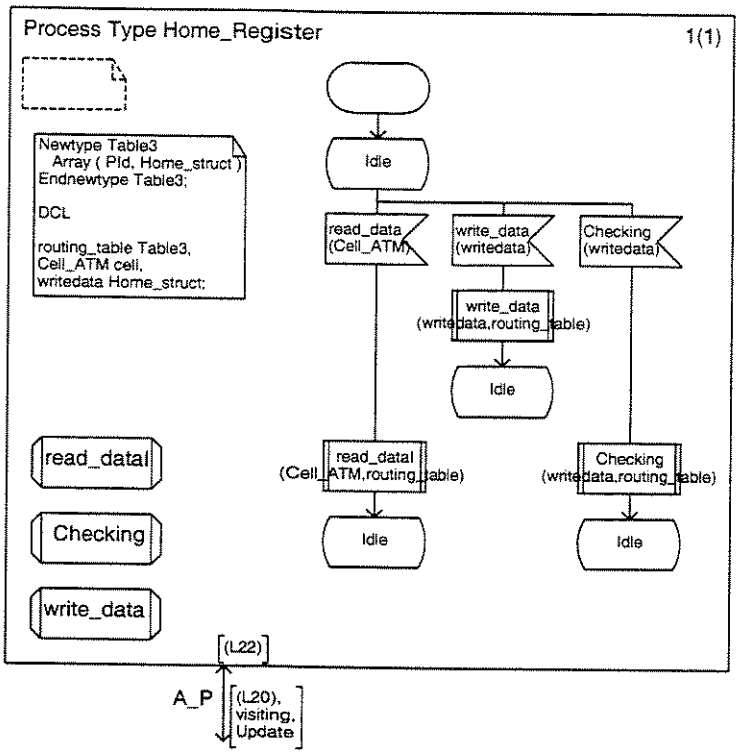

Figura 4.23: Processo Home_Register/Bloco DomainLocationService

visitantes, e o outro dos terminais locais. Esses dois processos compõem a estrutura *two-tier database.*

4.7 Conclusão

O objetivo desse trabalho é a especificação, em linguagem formal SDL, das sinalizações de Conexão e Registro de um terminal na rede WATM. Devido a isso, só foram apresentados nesse capítulo os procedimentos que executavam alguma função dentro da especificação das sinalizações.

A rede WATM foi modelada sob a forma de um sistema WATM, onde cada um dos dispositivos foi representado por um bloco. Os blocos exercem as funções devidas aos elementos da rede WATM, e para realizarem essas funções, eles possuem processos.

Os processos que formam os blocos, podem ser instanciáveis ou não. Isso dependeria de quantas unidades desse processo serão necessárias para o funcionamento da rede. Alguns processos não possuíam funções ligadas as sinalizações, contudo eram processos necessários para a execução da simulação das sinalizações do capítulo 5.

O bloco **DomainLocationService**, por exemplo, é, por completo, dedicado a funções de acesso aos bancos de dados e roteamento. Essas funções estão fora do objetivo desses trabalho, entretanto existe necessidade da existência desse bloco, e seus componentes, para

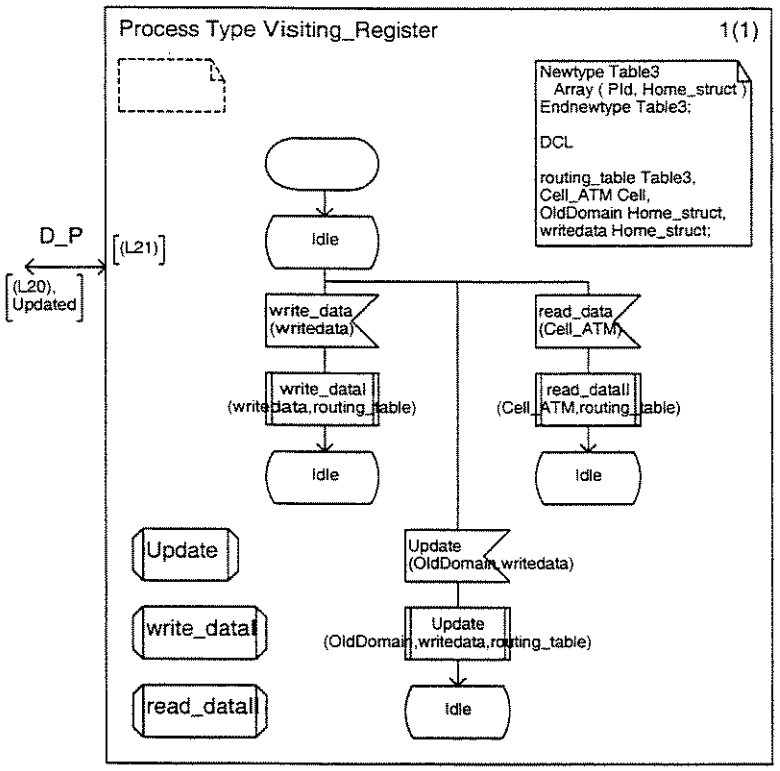

Figura 4.24: Processo Visiting_Register/Bloco DomainLocationService

a realização das simulações das sinalizações de Registro e Conexão. Devido à característica desse bloco, pouco foi comentado a respeito dele. Todavia, todos os procedimentos e processos que não foram apresentados aqui estão no apêndice B.

Capítulo 5

Simulação das Sinalizações de Registro e Conexão para a Rede WATM

5.1 Introdução

Durante uma especificação em SDL, a simulação é de valiosa ajuda para entender e analisar o comportamento do sistema. O simulador do SDT possui comandos para o controle da execução do sistema simulado, permite examinar o status de objetos em SDL, e possibilita obter informações para teste de validação. Essa interface do SDT, como resultado de simulação, gera comandos para obtenção de gráficos MSC (*Message Sequence Chart*) [30].

O MSC é uma linguagem textual e gráfica para a descrição e especificação das interações entre os componentes de um sistema. Essa linguagem é padronizada pelo ITU-TS(*International Telecommunication Union - Telecommunication Standardization*), na recomendação Z.120 [9].

O MSC foi utilizado informalmente por um longo tempo pelos *Study Groups* do ITU, em suas recomendações, e na indústria. Sua padronização foi sugerida no 4o. SDL Forum em outubro de 1989 na cidade de Lisboa, e aceito no encontro do ITU na cidade de Helsinki, em junho de 1990. No período de 1989 à 1992 foram realizadas sessões de estudos reservados do ITU. Em Geneva, maio de 1992, a recomendação Z120 foi aprovada [25].

A razão para a padronização do MSC foi dar um suporte às ferramentas semânticas, para facilitar a troca entre essas diferentes ferramentas, e para tornar fácil o mapeamento com a especificação em SDL. Devido à padronização, a importância dos MSC's para os sistemas de engenharia tem crescido consideravelmente.

A principal aplicação dessa linguagem, juntamente com o SDL, é na área de sistemas de telecomunicação. Seu uso, entretanto, não é restrito à metodologia SDL ou ao meio das telecomunicações (vide [15]). Entretanto, ele é muito mais difundido nesse meio que em outros, sendo usado em vários momentos do desenvolvimento dos sistemas tal como interface de especificação, simulação, validação e documentação [16], não sendo apenas

uma simples simulação.

Esse capítulo apresenta os resultados das simulações com as especificações das sinalizações de Registro e Conexão (seções 3.4.1 e 3.4.2 e capítulo 4) propostas para rede a WATM. Devido ao tamanho dos gráficos MSC's, foram selecionados dois exemplos de cada sinalização para serem apresentados, sendo sempre, um dos exemplos da sinalização mostrada, um caso particular do outro.

Para a sinalização de Registro, é mostrado um exemplo que aborda o registro de um terminal sendo realizado no seu próprio domínio. O outro caso apresentado explica o registro do móvel sendo efetuado em um MSP (*Mobile Switching Point*) diferente do domínio. Nesse segundo exemplo, é comentado o uso da mensagem *Updated*, seção 4.5.2.1, e mostrado a necessidade desta.

Para a sinalização de Conexão, o primeiro exemplo mostra a conexão entre dois terminais pertencentes a domínios diferentes, Contudo, estando o terminal chamado no mesmo MSP que o terminal que realizou o pedido de conexão. O outro exemplo desse processo, mostra uma conexão não efetivamente realizada, devido ao terminal destino não ter sido localizado.

Além das simulações das sinalizações, são apresentados dois exemplos de falhas realizadas durante a especificação, e que foram detectadas com a simulação, assim como as soluções adotadas para resolvê-las.

5.2 Simulação do Registro de um Terminal Móvel

Nessa seção é apresentada a troca de sinalização existente entre os dispositivos de uma rede WATM durante o registro de um terminal. Para essa simulação foram considerados dois MSP's, seis estações bases, e o móvel que será registrado.

Toda a troca de sinalização entre os componentes de uma rede WATM durante o procedimento de registro de um móvel é apresentado nas figuras 5.1, 5.2 e 5.3. O gráfico MSC foi dividido nestas três figuras para que fosse melhor compreendido e explicado.

A primeira mensagem da figura 5.1 é a *RegisterMobile*. Ela é enviada do **environment** - (**env_0**) para o processo **Signal_Manager**. O conteúdo dessa mensagem indica que ela registrará a instância 1 do processo **Mobile** - (**Mo_P:1**), pertencente ao domínio representado pela instância 1 do processo **MobileSwitchingPoint** - (**MSP_P:1**).

O processo **Signal_Manager** envia para a instância do processo **Mobile**, que irá se registrar, a mensagem recebida do **env_0**. O **Mo_P:1**, ao receber o sinal, envia a mensagem *Ask_Bases* para o processo **Base_Id_Table**. A resposta dessa mensagem, *Bases*, transporta o PId de três diferentes instâncias do processo **BaseStation**: **BS_P:3**, **BS_P:4** e **BS_P:5**.

Quando a instância **Mo_P:1** recebe os PId's do processo **BaseStation** indicando o destino do pedido de registro, ele envia a mensagem *Register_request*, para cada umas das instâncias deste processo. Essa mensagem transporta, nesse momento, o PId da estação base que está recebendo o pedido; o PId do MSP que é domínio do móvel; e o PId do móvel que requereu o registro.

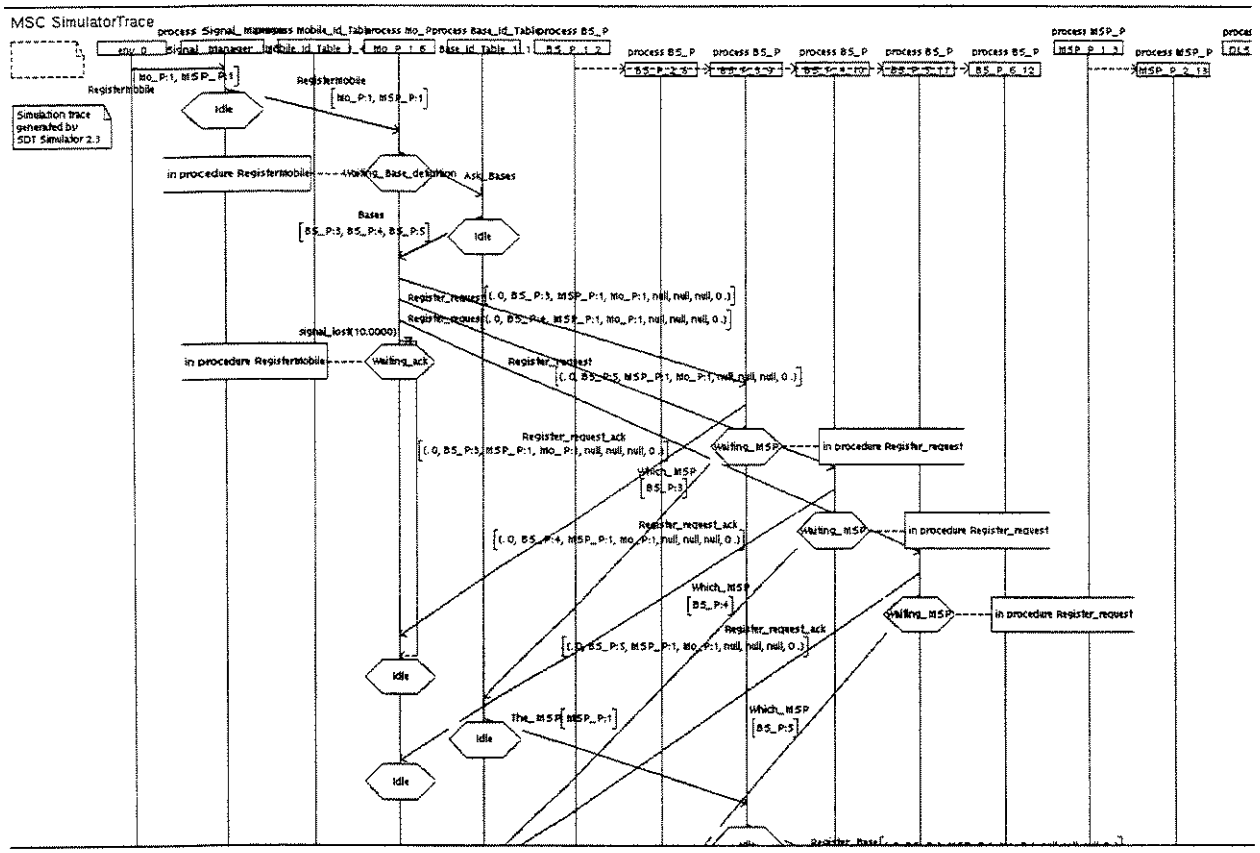

Figura 5.1: MSC do Registro do Terminal Móvel (1/3)

A mensagem *Register_request*, possui, entre colchetes, o seu conteúdo. Para a primeira mensagem *Register_request*, o conteúdo é : **[(. 0, BS_P:3, MSP_P:1, Mo_P:1, null, null, null, 0.)]**. Isso representa que a mensagem irá para **BS_P:3**, que o domínio do móvel é **MSP_P:1**, e que o terminal é o **Mo_P:1**. Os campos com **0** e **null** não foram preenchidos ainda.

Como as estações base podem pertencer a diferentes MSP's, quando qualquer instância do processo **BaseStation** recebe algum pedido vindo do móvel, para o MSP, ela precisa consultar no processo **Base_Id_Table** o PId da instância do processo **MobileSwitchingPoint** para a qual deve mandar a mensagem. Essa averiguação é feita com a mensagem *Which_MSP*. Ela transporta o PId da instância do processo *BaseStation* que deseja a informação. No exemplo da figura 5.1, essa mensagem é enviada pelos três processos que receberam o pedido de registro do móvel.

A figura 5.2 inicia com o envio, a partir do procedimento **Base_Id_Table**, da mensagem *The_MSP*. Esta transporta a instância do processo **MobileSwitchingPoint** para o qual a estação base deve enviar o pedido de registro.

Quando essa mensagem, *The_MSP*, atinge sua instância destino, ela envia o pedido de registro através da mensagem *Register_Base*. O valor do primeiro campo desta, muda

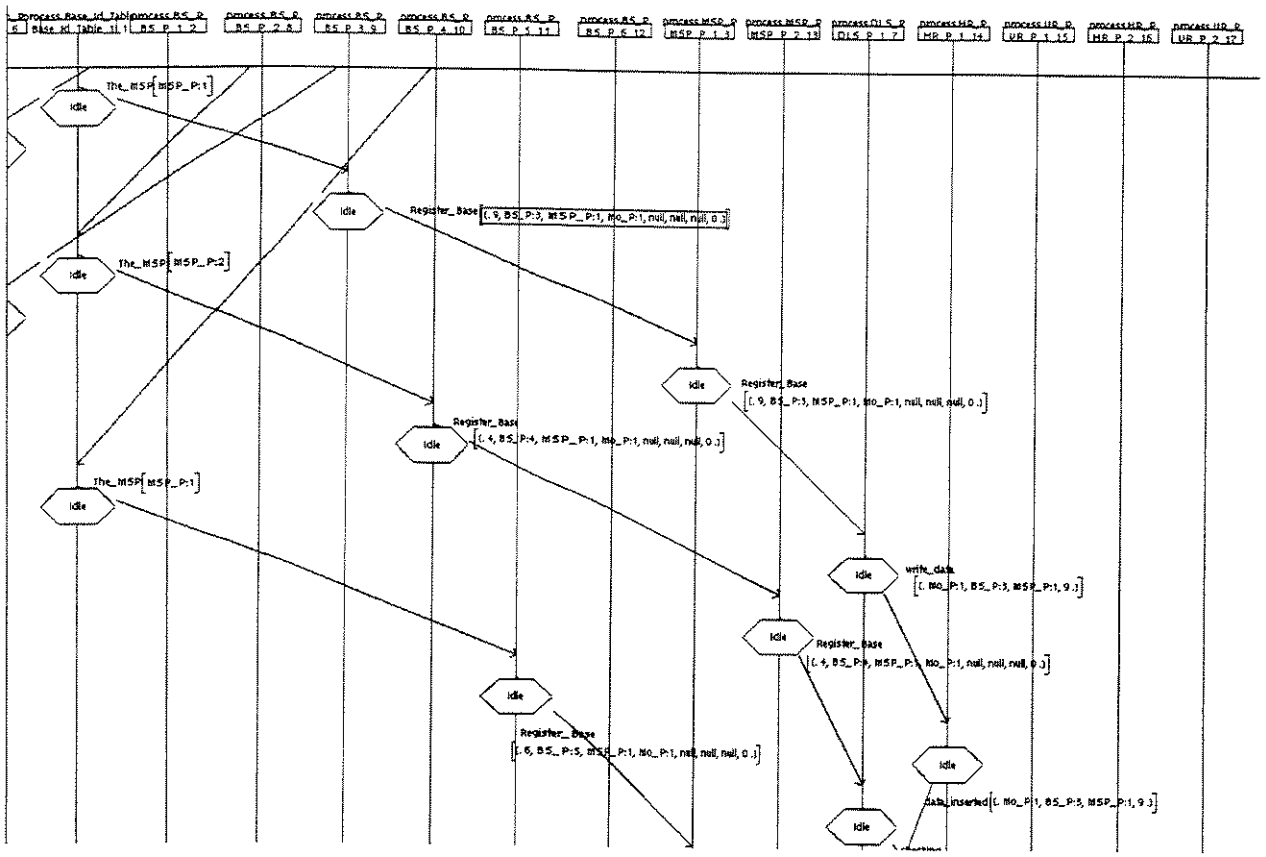

Figura 5.2: MSC do Registro do Terminal Móvel (2/3)

de **0** para um número *Natural*, e representa a potência[1] com que a estação base recebe o pedido do terminal. Os outros campos são idênticos aos da mensagem *Register_request* que apareceu na figura 5.1.

Quando a *Register_Base* atinge a instância do processo **MobileSwitchingPoint** - (**MSP_P**), alvo, esta a repassa para o processo **DomainLocationService** - (**DLS_P**).

Ao receber a *Register_Base* do processo **MSP_P**, o **DLS_P** verifica se ela é proveniente do MSP domínio do móvel. Se for, que é o que aconteceu com a primeira *Register_Base* que chega ao **DLS_P** da figura 5.2, a mensagem *write_data* é enviada para a instância do processo **Home_Register** - (**HR_P**) que contém os dados do domínio do terminal, transportando a instância dos processos **Mo_P**, **BS_P** e **MSP_P**, além da representação da potência com que a estação recebeu o pedido de registro. No exemplo da figura 5.2, a mensagem *write_data* é enviada para **HR_P:1**, e seus dados são [(**.Mo_P:1, BS_P:3, MSP_P:1, 9.**)].

O processo **HR_P**, ao receber a mensagem *write_data*, avaliará a inserção dos dados de registro do terminal. No caso da figura 5.2 os dados são aceitos e inseridos no banco de dados, então a mensagem *data_inserted* é enviada para **DLS_P**, contendo os mesmos

[1]Esse valor possui significado apenas representativo e, conforme mencionado nos capítulos 3 e 4, é um número aleatório.

dados da *write_data*.

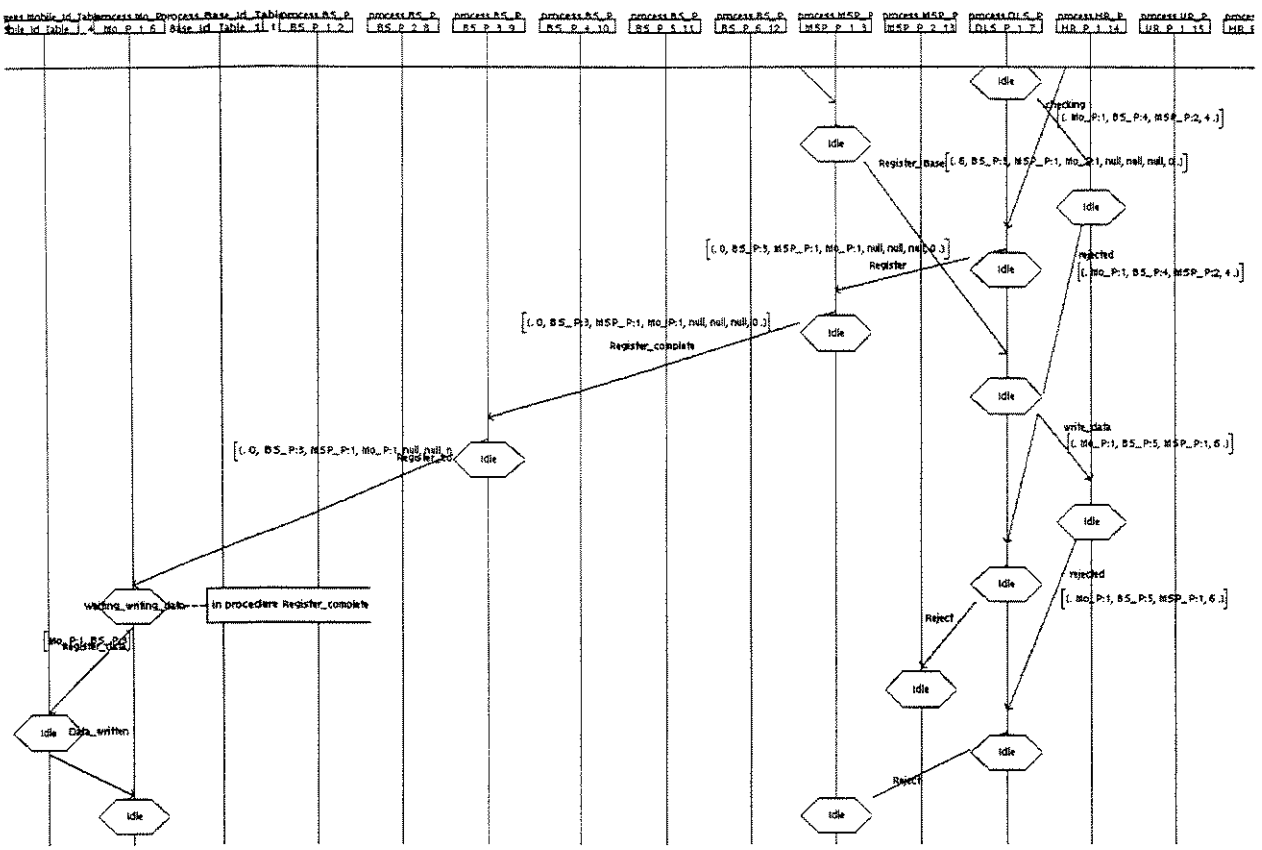

Figura 5.3: MSC do Registro do Terminal Móvel (3/3)

A mensagem *Register_Base*, na figura 5.2, cujo conteúdo é [(**.4, BS_P:4, MSP_P:1, Mo_P:1, null, null, null, 0.**)], chega ao *DLS_P* vinda do **MSP_P:2**, que é a instância dois do processo **MSP_P**. O móvel pertence, entretanto, ao **MSP_P:1**, como indica o corpo da mensagem *Register_Base*. Devido a isso, o **DLS_P** envia para o **HR_P**, do domínio do móvel - (**HR_P:1**), uma outra mensagem, a *checking*, que aparece no canto superior direito da figura 5.3. Esta transporta os mesmos dados que a *write_data*.

O **HR_P:1** verificará os dados da *checking* e decidirá pelo registro, ou não, do móvel, assim como fez com a *write_data*. No exemplo da figura 5.3, ele foi rejeitado, então é enviada, para o **DLS_P**, a mensagem *Rejected*, com o mesmo conteúdo da *checking*.

Quando o **DLS_P** recebe a mensagem *Rejected*, ou *data_inserted*, ele transmite, para o *MSP_P* que enviou o pedido de registro, a mensagem *Reject*, ou *Register*, respectivamente.

Se o **MSP_P** recebe a mensagem *Register*, ele avisa o **BS_P** que o pedido foi aceito. Na figura 5.3 essa mensagem é enviada para **BS_P:3**.

O **BS_P**, ao receber a *Register_complete*, a repassa para o **Mo_P** que pediu o registro; no exemplo **Mo_P:1**. O processo **Mo_P**, quando recebe o indicativo da confirmação do registro, envia para o **Mobile_Id_Table** a mensagem *register_data* contendo o PId, da

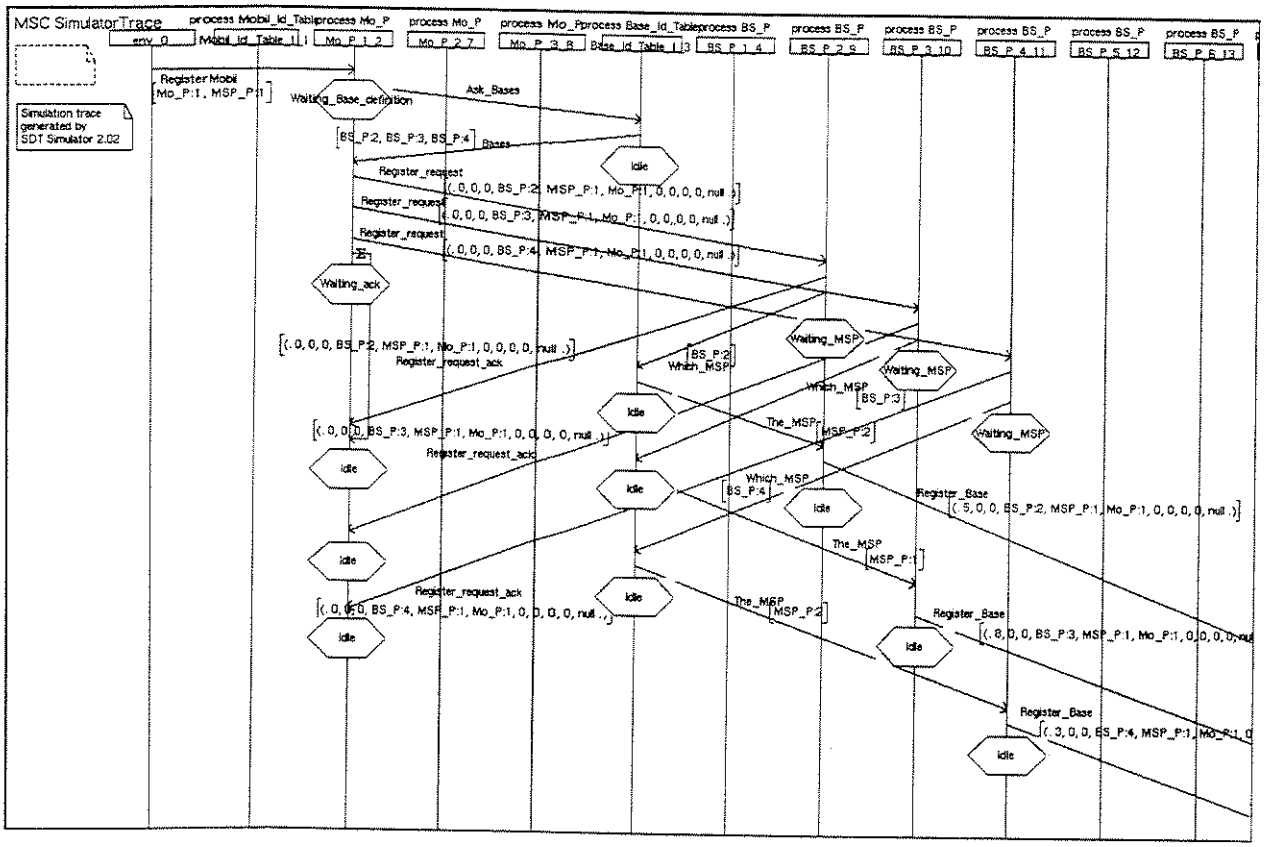

Figura 5.4: MSC do registro de um terminal com uso da mensagem *Updated* (1/3)

instância registrada, e da estação base que assiste o móvel. O **Mobile_Id_Table**, retorna ao **Mo_P** a confirmação com a mensagem *Data_written.*

5.2.1 Exemplo do Uso da Mensagem *Updated*

Durante as simulações foi observado que, quando o terminal registrava-se primeiro em um MSP que não era domínio dele, ou seja, registrava-se como visitante, e depois atualizava os dados de registro com valores obtidos dos dispositivos do próprio domínio dele, isto é, registrava-se como *home*, ocorria de o terminal receber primeiro a confirmação de que ele estava no *home*, e depois receber a informação de que ele estaria registrado como visitante. Isso é o inverso do que deveria acontecer, pois os dados de registro do terminal obtidos a partir do domínio dele substituíam os dados do MSP que não era o domínio dele. Para solucionar isso foi introduzido o sinal *Updated.* O MSC que apresenta a simulação onde ocorre o uso da mensagem *Updated* é separado nas três figuras 5.4, 5.5 e 5.6.

As estações base envolvidas nesse registro são representadas pelas instâncias 2, 3 e 4 - (**BS_P:2**, **BS_P:3** e **BS_P:4**. Como mostra a figura 5.4 essas estações informam, pela mensagem *Register_Base*, que receberam o pedido de registro do móvel com os seguintes valores de potência: **5**, **8** e **3**.

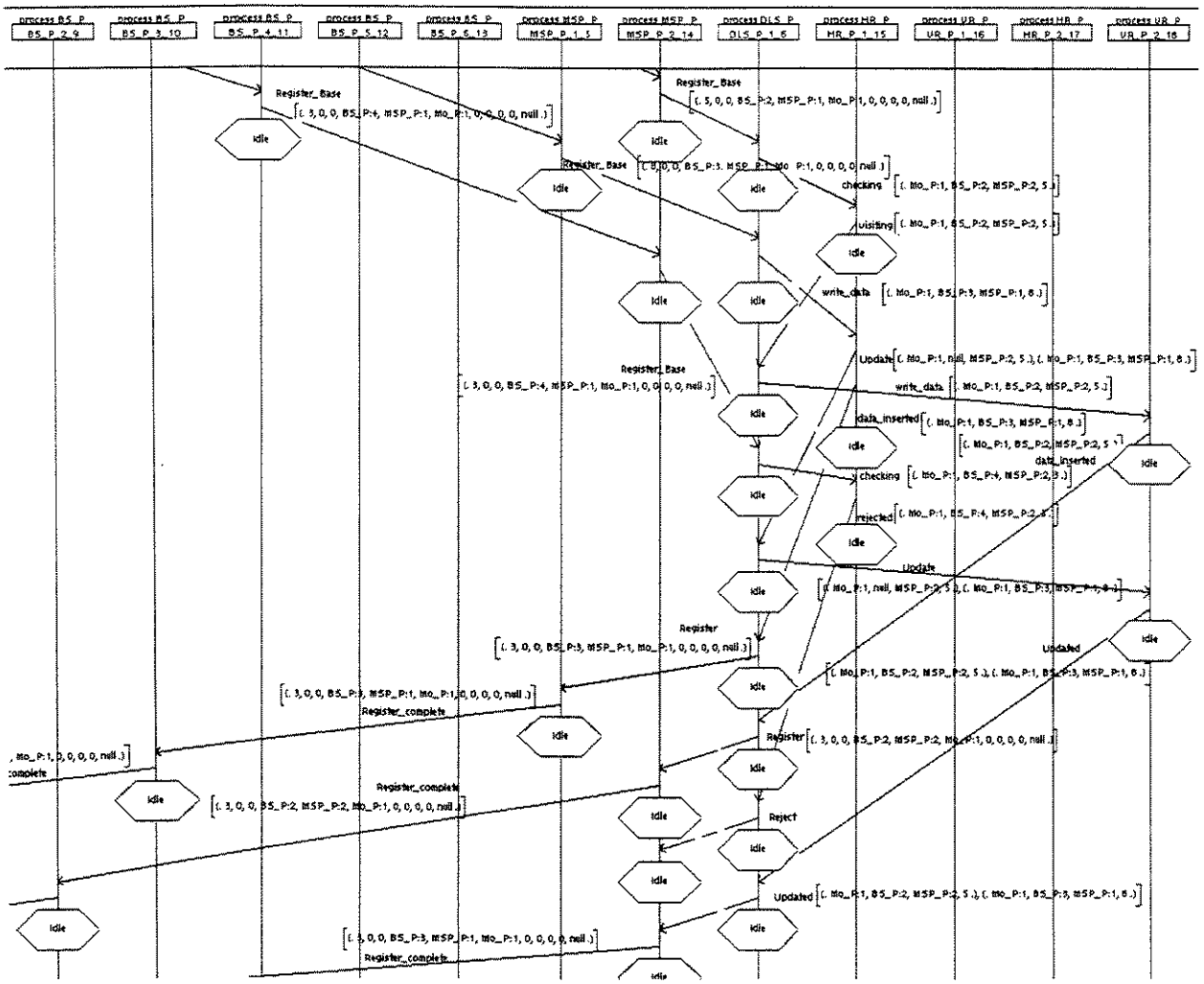

Figura 5.5: MSC do registro de um terminal com uso da mensagem *Updated* (2/3)

A escolha de qual estação base assistirá o terminal é feita observando os valores de potência, portanto a estação base que serviria o terminal seria a representada pelo **BS_P:3**, pois possui o maior valor - **8**.

A mensagem *Register_Base*, proveniente do processo **BS_P:3**, chega ao **DLS_P** após a recepção da *Register_Base* originada pelo processo **BS_P:2**. Devido à ordem de chegada das mensagens no **DLS_P**, as informações do **BS_P:2** são registradas e depois atualizadas com as do **BS_P:3**.

A estação base do processo **BS_P:3** que deveria servir o móvel pertence ao mesmo domínio que o terminal; por isso seus dados são inseridos apenas no **HR_P:1**, como mostra a figura 5.5, ao passo que os dados do **BS_P:2** são inseridos no **HR_P:1** e no **VR_P:2**.

Isso faz com que a mensagem *Register_complete*, na figura 5.6, chegue ao móvel, **Mo_P:1**, primeiro informando que ele trocará mensagens com **BS_P:3**. Depois, a mes-

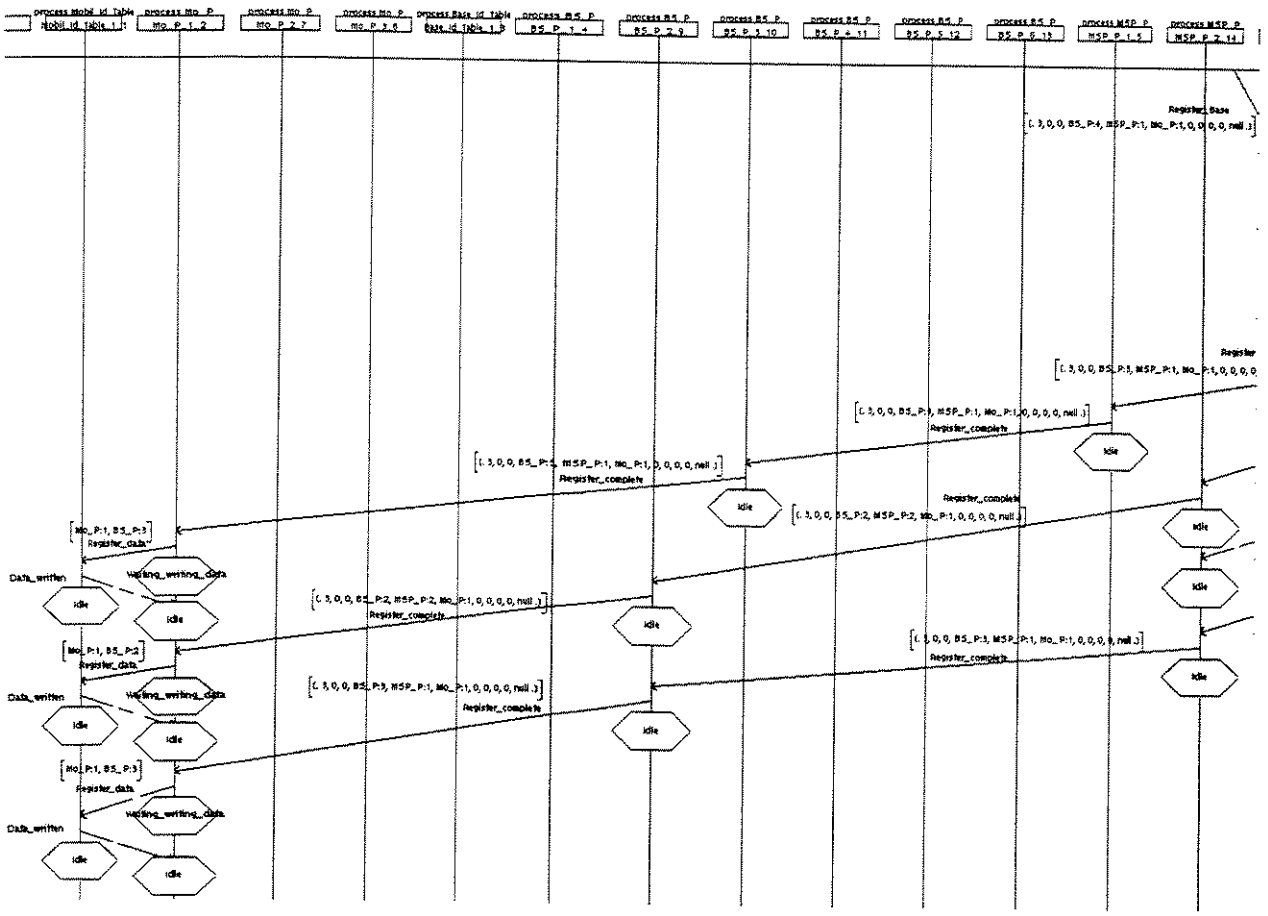

Figura 5.6: MSC do registro de um terminal com uso da mensagem *Updated* (3/3)

ma mensagem chega indicando que o terminal comunicar-se-á com **BS_P:2**. O **Mo_P:1** acatará a última informação recebida, e atualizará seus dados guardados no processo **Mobile_Id_Table**. Com isso, o móvel não estaria trocando informações com a estação base esperada. Esse problema apareceu em simulações como a composta pelas figuras 5.4, 5.5 e 5.6.

Para solucionar isso foi proposto o sinal *Updated* [3], que aparece na metade inferior lado direito da figura 5.5. Dessa forma, sempre que o móvel tem os valores dos dados de registro dele alterados por outros valores provenientes de uma estação base pertencente ao domínio do móvel, é enviado ao *VR_P* do antigo MSP a mensagem de *Update*, para que ele apague os dados do móvel e, por segurança, envie os novos pela antiga estação base que servia o terminal. Como resultado disso, tem-se a última mensagem *Register_Complete* da figura 5.6 transportando os mesmos dados que a primeira. O **Mo_P:1** atualiza seus dados e passa a se comunicar com **BS_P:3**.

5.3 Simulação de uma Conexão do Terminal Móvel

Nessa seção é apresentada a troca de sinalização existente entre os dispositivos de uma rede WATM durante a conexão entre dois terminais pertencentes a uma rede WATM. Para essa simulação foram considerados três MSP's, seis estações bases, e dois móveis. O diagrama MSC referente a simulação da conexão de um móvel foi separado em quatro figuras para facilitar o entendimento. As figuras que compõem este procedimento são: figura 5.7, 5.8, 5.9 e 5.10.

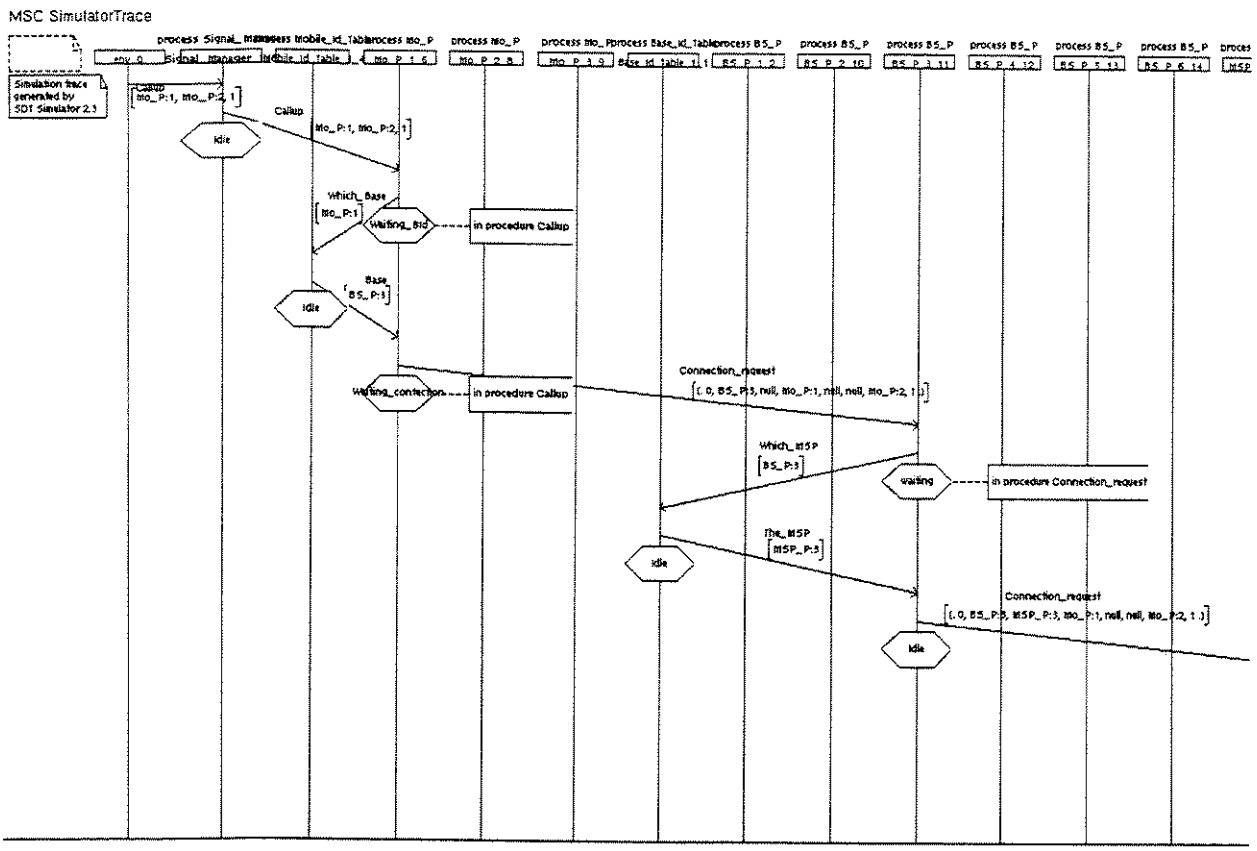

Figura 5.7: MSC do Conexão do Terminal Móvel (1/4)

O gráfico da figura 5.7 começa com o envio da mensagem *Callup*, que vem do ambiente, **environment** - (**env_0**), para o processo **Signal_Manager**. Essa mensagem transporta o PId da instância que representa o móvel origem; o PId da instância que representa o móvel destino; e a classe QoS da conexão[2] - [**Mo_P:1, Mo_P:2, 1**].

A mensagem de *Callup* é direcionada para a instância determinada, **Mo_P:1**. Ao ser recebida, é enviada a mensagem *Which_Base* para o processo **Mobile_Id_Table**, figura 5.7. Esse processo, informa ao móvel, como resposta a mensagem enviada, a instância do processo **BaseStation** - (**BS_P**) para a qual o pedido de registro deve ser enviado.

[2]Para descrição das classes, consultar a tabela 2.1, capítulo 2.

A mensagem *Base* responde à mensagem *Which_Base* indicando para a instância do móvel, que deseja fazer a conexão, a instância destino para a mensagem *Connection_request*. No exemplo da figura 5.7 a mensagem indica que o destino é para a instância três do processo **BS_P** - (**BS_P:3**).

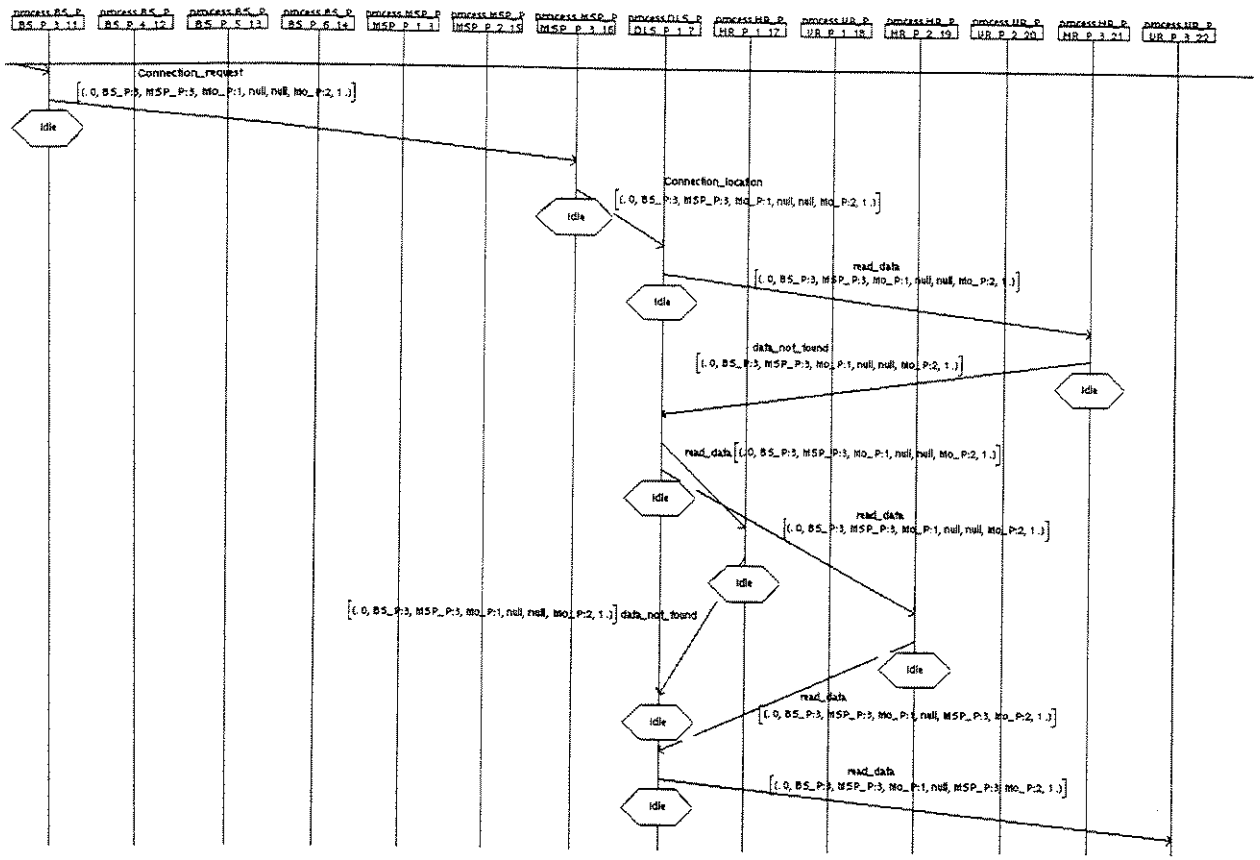

Figura 5.8: MSC do Conexão do Terminal Móvel (2/4)

A mensagem *Connection_request* é enviada para **BS_P:3** contendo a informação: **BS_P:3**, no segundo campo; **Mo_P:1**, no quarto campo; **Mo_P:2** no penúltimo campo; e **1** no último campo. Esses campos representam respectivamente a estação base fonte, o móvel fonte, o móvel destino e a classe.

Quando essa mensagem chega à instância **BS_P:3**, é usada a seqüência de mensagens *Which_MSP* e *The_MSP* comentadas na seção 5.2. A instância do processo **MobileSwitchingPoint** - (**MSP_P**) destino para essa mensagem de pedido de conexão do móvel é recebida na *The_MSP*, e indica o processo **MSP_P:3**.

Na figura 5.8 a mensagem que indica o pedido de conexão, *Connection_request*, é enviada para **MSP_P:3**, transportando os mesmos dados que chegaram na estação base acrescido, no terceiro campo da mensagem, da resposta da *Which_MSP*.

Quando a mensagem é recebida no **MSP_P:3**, este encaminha para o processo **DomainLocationService** - (**DLS_P**) a mensagem *Connection_location*, possuindo todas as informações trazidas na *Connection_request*.

O **DLS_P** quando recebe o pedido de localização de um terminal[3], averigua, inicialmente no processo **Home_Register** - (**HR_P**) onde o **MSP_P**, que enviou o pedido de localização, registra os seus terminais, a presença do terminal destino.

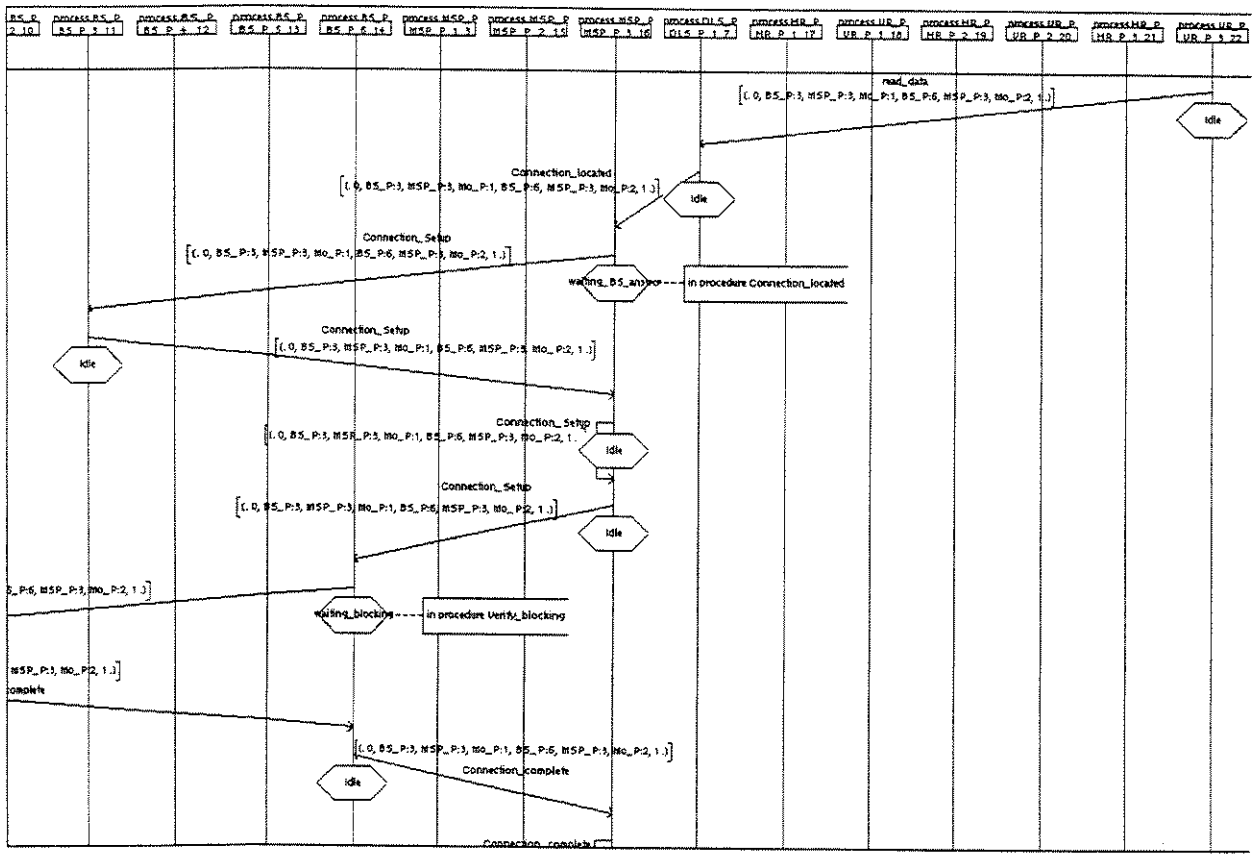

Figura 5.9: MSC do Conexão do Terminal Móvel (3/4)

A mensagem *read_data* é enviada para o **HR_P:3**. Esse contém os mesmos dados da *Connection_location*. Se o **DLS_P** recebe a mensagem *data_not_found*, que determina a inexistência do terminal destino na sua tabela, então é enviado para todas as instâncias do processo **HR_P** a mensagem *read_data*. Caso o terminal destino esteja como visitante, o **HR_P** que possui os dados do terminal destino envia para o **DLS_P** a informação de que este terminal está como visitante. O**DSL_P** ao receber a informação envia a mensagem *read_data* para o processo **VR_P** identificado pelo **HR_P** do domínio do móvel destino.

Na figura 5.8 o **HR_P** que possui os dados do móvel destino, envia a mesma mensagem de volta, *read_data*, para o **DLS_P** informando que o terminal destino encontra-se como visitante na instância **MSP_P:3**. Na figura 5.8 é a única mensagem *read_data* que chaga ao **DLS_P**. Essa informação de que o terminal móvel está em **MSP_P:3** é identificada no sexto campo da mensagem *read_data*. O **DLS_P** envia a mesma mensagem *read_data*

[3]Não houve o intuito de se desenvolver um algoritmo eficiente de roteamento, pois não está no escopo desse trabalho

para o processo **VR_P:3**, na figura 5.8 é a última mensagem *read_data* que sai do **DLS_P**.

Na figura 5.9 o **VR_P:3** que possui os dados do móvel destino, envia a mesma mensagem de volta, *read_data*, para o **DLS_P**. Essa mensagem possui no quinto campo a instância do processo **BS_P** que atende ao terminal chamado, e a instância do processo **MSP_P** onde o **BS_P** está conectado. Neste exemplo os dados são **BS_P:6** e **MSP_P:3**. O **DLS_P**, ao receber a mensagem *read_data*, envia a mensagem *Connection_location* para o **MSP_P:3**.

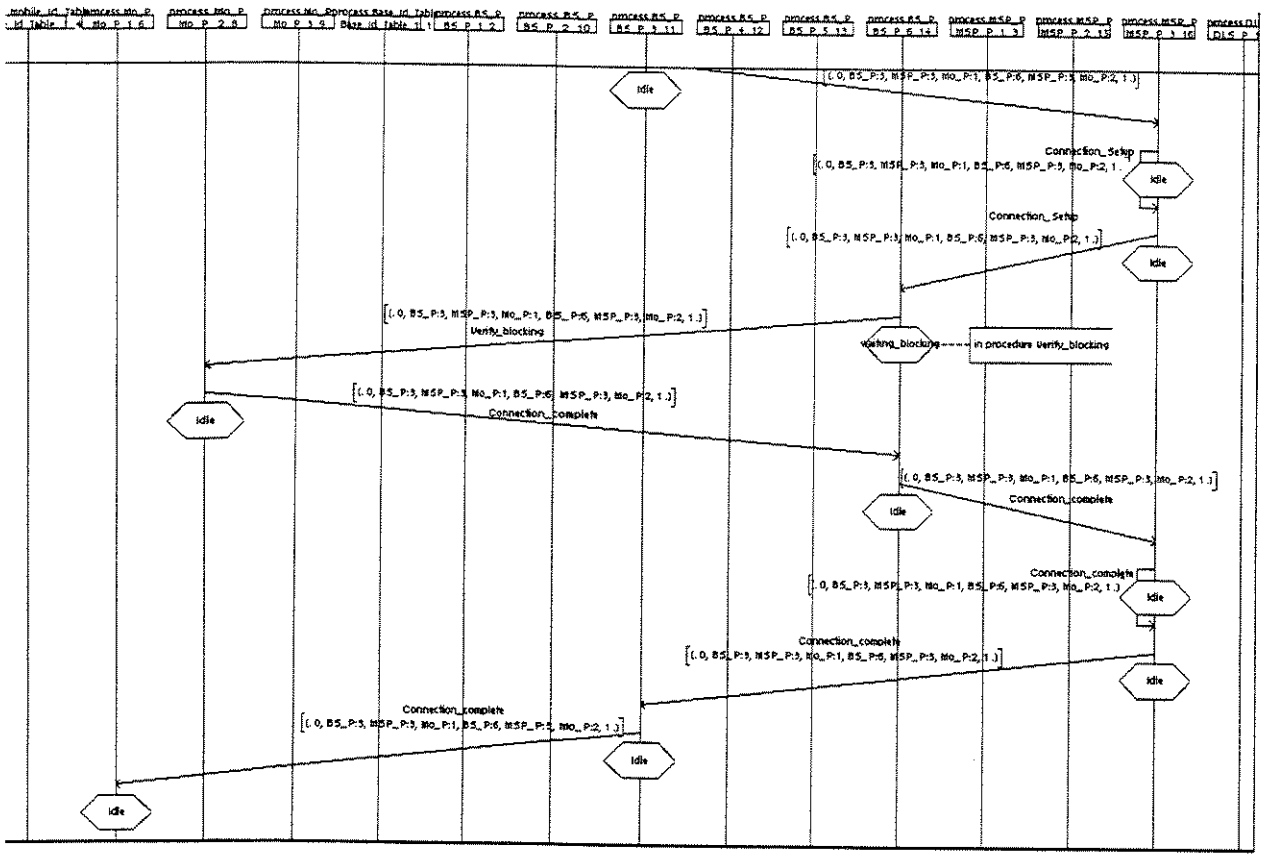

Figura 5.10: MSC do Conexão do Terminal Móvel (4/4)

A definição da estação base que assistirá ao terminal após o registro dele é feito de modo aleatório. No exemplo utilizado para a conexão, o terminal destino estava como visitante no mesmo MSP do móvel de origem, contudo isso nada modifica a troca de mensagens.

Após o **MSP_P:3**, na figura 5.9, receber a mensagem *Connection_located*, do **DLS_P**, ele envia a mensagem *Connection_Setup* para **BS_P:3** que representa a estação base de origem. Esta responde com a mesma mensagem depois de ter aceito a classe pedida na conexão.

O **MSP_P:3**, na figura 5.9, quando recebe a aprovação do **BS_P**, envia para o **MSP_P** destino a mesma mensagem, *Connection_Setup*, com o mesmo objetivo de aceitação da

classe de conexão. Como no exemplo o MSP alvo e destino são o mesmo, o **MSP_P:3** envia a mensagem para ele mesmo.

O **MSP_P:3** envia agora a mensagem *Connection_Setup* para a instância **BS_P:6** que representa a estação base destino. Essa troca de mensagens é apresentada nas figuras 5.9 e 5.10.

Na figura 5.10, após a conexão ser aprovada , o **BS_P:6** envia a mensagem *Verify_blocking* para o móvel destino, **Mo_P:2**. O processo **Mo_P:2** envia de volta a mensagem *Connection_complete* para o **BS_P:6**, que repassa para **MSP_P:3**.

O processo **MSP_P:3** ao receber a mensagem *Connection_complete* da estação base destino, **BS_P:6**, envia a para o MSP origem, no exemplo **MSP_P:3**, por isso a mensagem é enviada para a própria instância **MSP_P:3**.

O MSP origem, **MSP_P:3**, ao receber a mensagem *Connection_complete*, a envia para a estação base de origem, **BS_P:3**, que a transmite para o móvel que pediu a conexão **Mo_P:1**.

5.3.1 Exemplo de Conexão onde o móvel não é encontrado

Esse exemplo de simulação de pedido de conexão mostra o caso em que o terminal destino não é encontrado. Quando esse fato ocorre, é feito uso da mensagem *Target_not_found* para informar ao elementos da rede a não localização do terminal ao qual destina-se a chamada. O gráfico MSC que apresenta a simulação desse cenário é composto pelas figuras 5.11, 5.12 e 5.13.

A figura 5.11 apresenta uma troca de mensagens entre os elementos do sistema **WATM** similar a mostrada na figura 5.7. Essa parte inicial do gráfico MSC tem que ser semelhante para qualquer pedido de conexão, pois apresenta as trocas de mensagens realizadas entre os processo do bloco **Mobile** (**Signal_manager**, **Mobile_Id_Table** e **Mo_P**) e os processos do bloco **BaseStation** (**Base_Id_Table** e **BS_P**).

A figura 5.12 apresenta a diferença com relação a figura 5.8 que determina que o terminal destino não foi encontrado. Na figura 5.8, uma das instâncias do processo **HR_P**, que recebeu a mensagem *read_data*, devolveu essa mensagem para o processo **DLS_P** indicando a localização do terminal destino. Na figura 5.12, todas as instâncias do processo **HR_P**, que receberam a mensagem *read_data*, retornaram, para **DLS_P**, a mensagem *data_not_found*. Como o processo **DLS_P** não conseguiu obter a localização do terminal destino, ele envia para **MSP_P** que fez o pedido de conexão, no exemplo o **MSP_P:3**, a mensagem *Target_not_found*, indicando que o terminal destino não foi encontrado.

O **MSP_P:3**, ao receber essa informação, a transmite, usando a mesma mensagem, para o **BS_P:3**, que representa a estação base que assiste o móvel. O terminal que pediu a conexão é avisado, pela estação base, também por intermédio da mensagem *Target_not_found*, de que o terminal destino não foi encontrado. Isto é apresentado na figura 5.13.

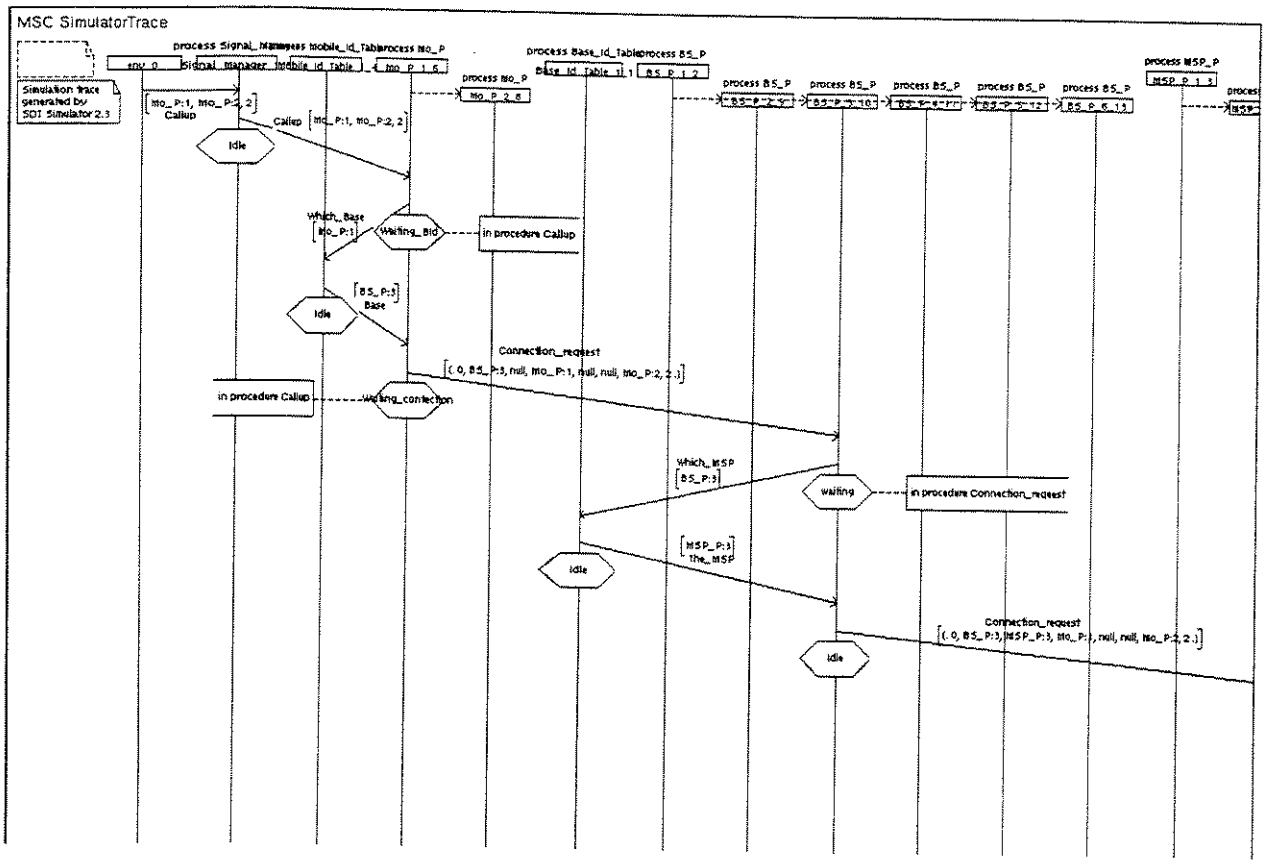

Figura 5.11: Exemplo onde o terminal destino não é localizado (1/3)

5.4 Exemplos de falhas de especificação detectadas em simulações

Durante a especificação das sinalizações de Registro e Conexão realizadas nesse trabalho, ocorreram diversas falhas que tiveram que ser corrigidas até que fosse obtido um sistema final estável. Nesta seção são apresentados dois exemplos de falhas, uma cometida na especificação da sinalização de Registro e outra na sinalização de Conexão.

Quando as simulações são executadas com o objetivo de testar as especificações, usa-se uma opção no *SDT Simulator* que apresenta cada detalhe da execução da especificação, como, por exemplo, o valor que cada variável recebe, o resultado de uma decisão (se é falsa ou verdadeira), ou mesmo se um procedimento foi executado. Como os gráficos MSC apresentados nessa seção foram obtidos a partir de simulações que tiveram o intuito de procurar falhas nas especificações, eles são mais ricos em detalhes que os demais gráficos já apresentados nesse capítulo.

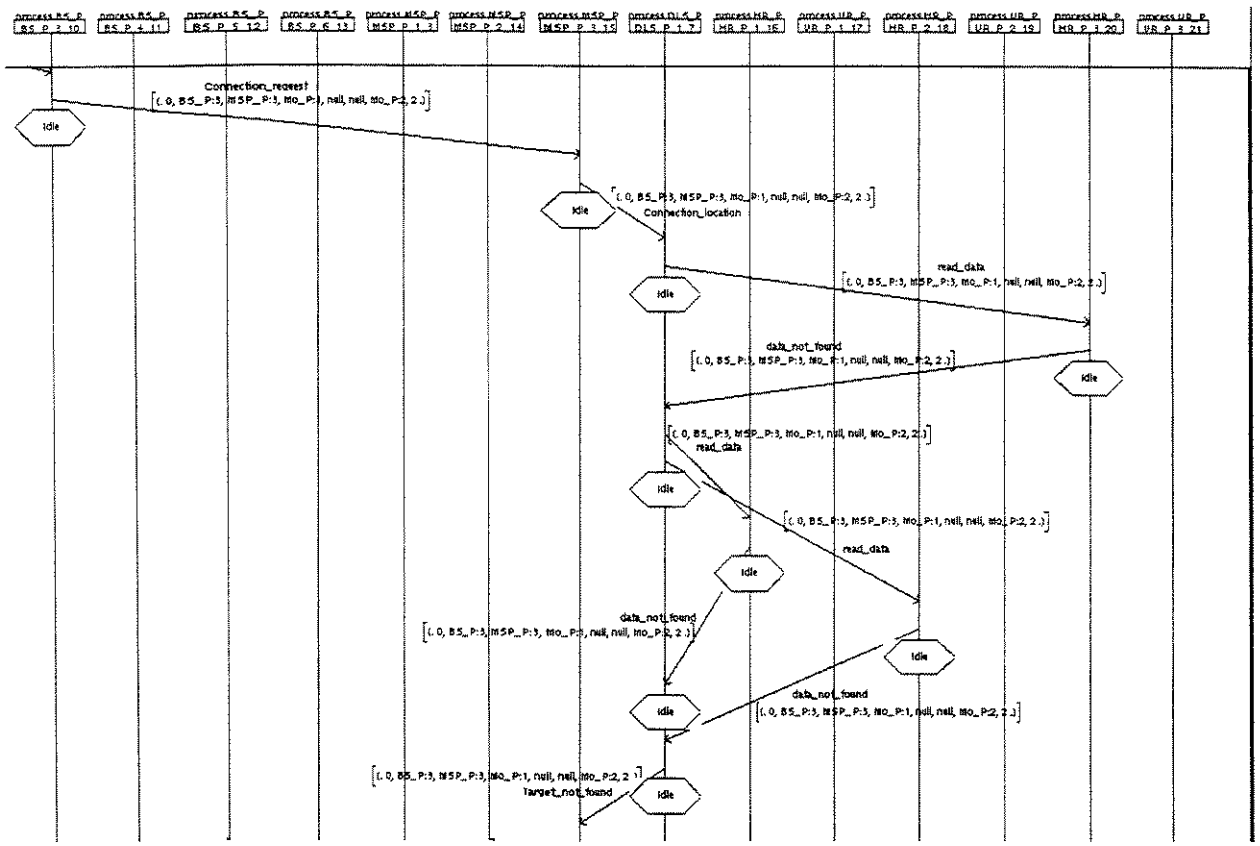

Figura 5.12: Exemplo onde o terminal destino não é localizado (2/3)

5.4.1 Exemplo com a sinalização de Registro

Nessa seção é apresentada uma simulação da sinalização de Registro onde foi detectada uma falha na especificação formal dessa sinalização. A figura 5.14 apresenta a parte do diagrama MSC onde o erro foi identificado. Durante a falha, o sistema estava no instante de recepção, e averiguação, dos dados de registro de um terminal. Essa tarefa é realizada pelo processo **DomainLocationService - (DLS_P)**.

A figura 5.14 apresenta três mensagens identificadas com um asterisco: *Register_Base*; *visiting*; e *write_data*. Entretanto, as mensagens *Register_Base* e *visiting* diferem da *write_data*, pois possuem início e fim, o que não ocorreu com a *write_data*.

Nesse exemplo, os asteriscos nas duas primeiras mensagens citadas, indicam que elas não foram processadas no destino, ou seja, não foram consumidas. Entretanto, isso não caracteriza um erro de especificação, pois a simulação foi interrompida antes do final quando estão estas mensagens seriam processadas

A última mensagem do gráfico, não só deixou de ser atendida, como também não chegou ao seu destino. Isso provocou a interrupção da simulação. O modo como esse sinal está representado indica que ele foi perdido. O motivo para esse problema foi que durante a especificação do procedimento **Register_Base**, do processo **DomainLocationService**,

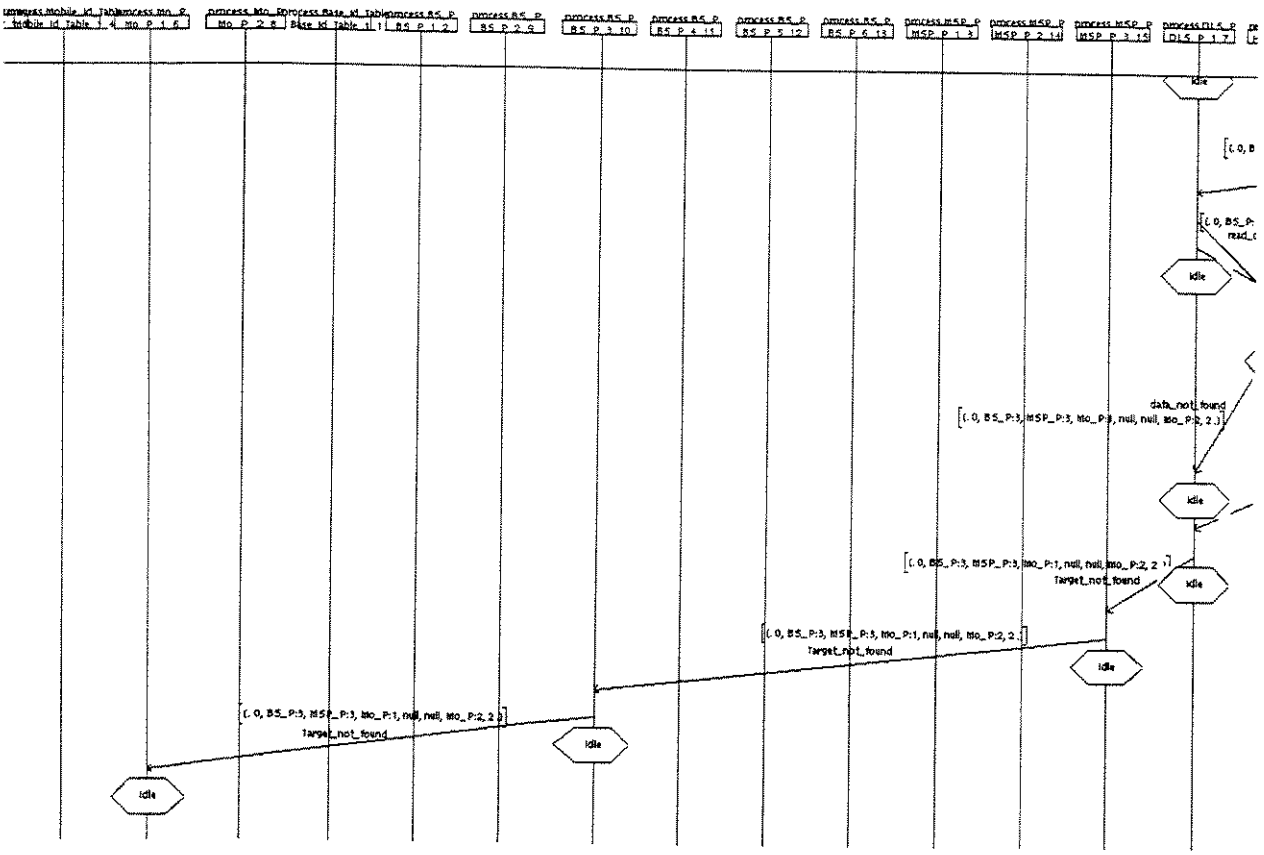

Figura 5.13: Exemplo onde o terminal destino não é localizado (3/3)

a matriz **HR_Table** não foi especificada nos parâmetros formais, *FPAR*, desse procedimento. Como a mensagem *write_data* necessitava que a matriz estivesse declarada para o seu conteúdo podesse ser enviado pelo bloco **DomainLocationService**, ela não pôde ser transmitida corretamente. Esse problema foi corrigido com a declaração da matriz **HR_Table** nos parâmetros formais do procedimento **Register_Base**.

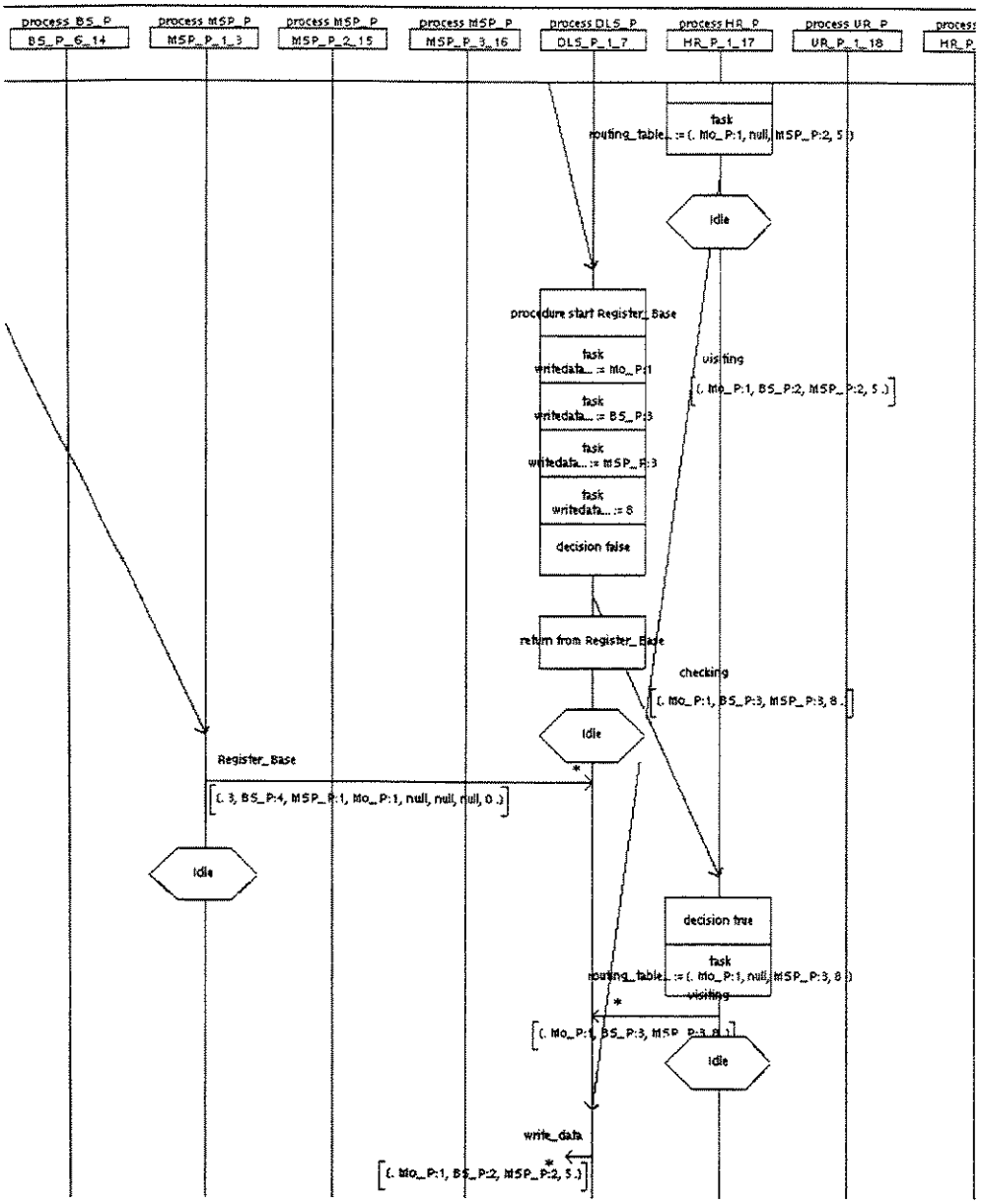

Figura 5.14: Exemplo de falha na sinalização de Registro do terminal

5.4.2 Exemplo com a sinalização de Conexão

Nessa seção é apresentada uma simulação da sinalização de Conexão onde foi detectada uma falha na especificação formal dessa sinalização. A figura 5.15 apresenta o diagrama MSC onde o erro foi identificado.

Na figura 5.15, a mensagem *read_data* não atinge o destino esperado, ou seja, é perdida. Observando o conteúdo da mensagem, que é última da figura 5.15, [(**.0, BS_P:4, null, Mo_P:1, null, null, Mo_P:1, 1.**)], o terceiro campo está **null**. Enquanto deveria constar

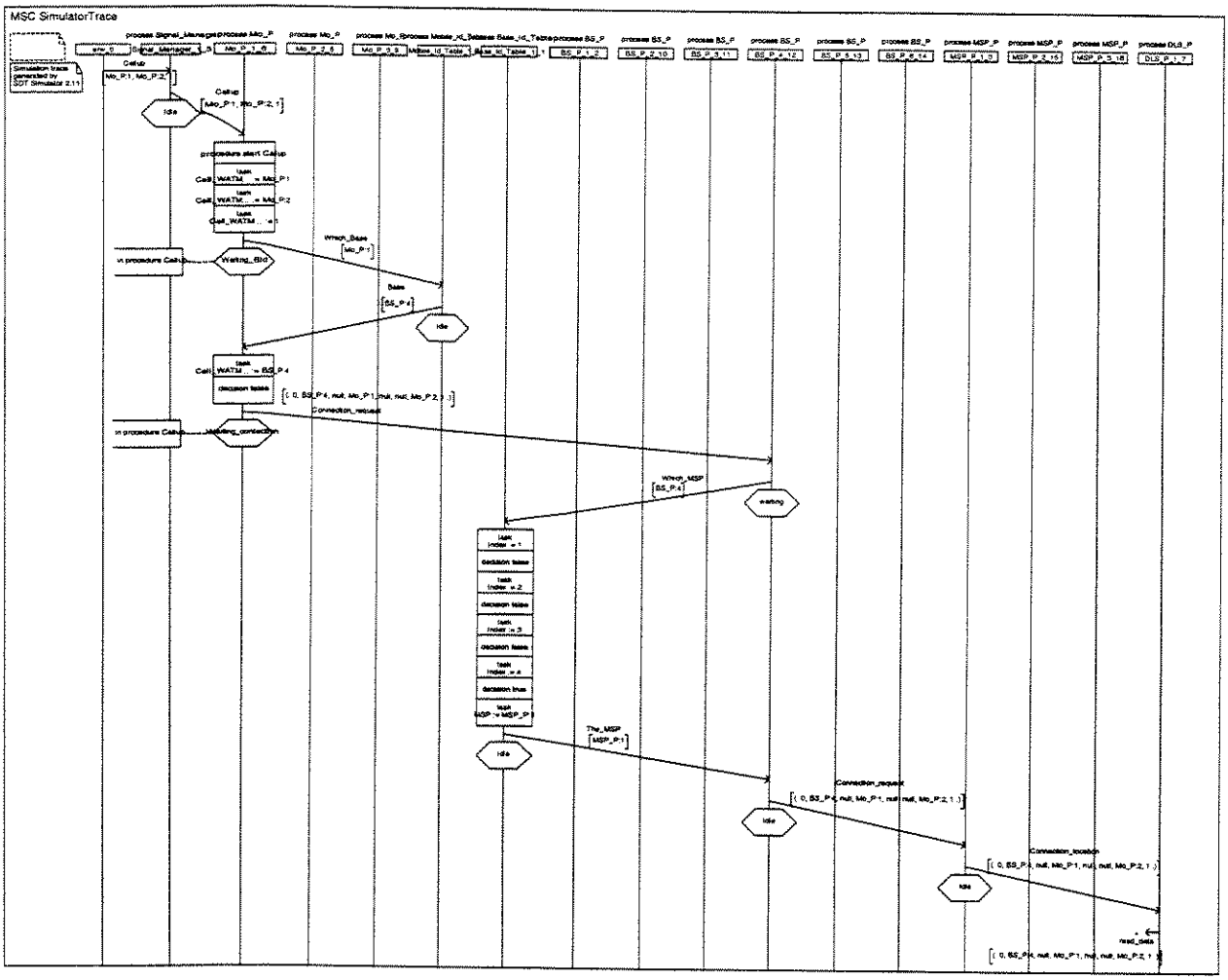

Figura 5.15: Exemplo de falha na sinalização de Conexão do terminal

o conteúdo do sinal *The_MSP*, [**MSP_P:1**].

Esse erro passou por algumas simulações sem ser acusado, até o momento em que as funções executadas dentro dos processos, foram redefinidas em procedimentos. Com o uso de procedimentos, todas as variáveis utilizadas na execução das funções tinha seus valores apagados quando o procedimento era finalizado, evitando que as variáveis guardessem valores já utilizados. Com essa modificação (uso de procedimentos), foi observado que o sinal *Connection_request* possuia o valor **null** no seu terceiro campo, quando deveria possuir o valor enviado pelo sinal *The_MSP*, a ausência desse valor causou o erro mostrado na figura 5.15. A solução foi atribuir o conteúdo da mensagem *The_MSP* para esse campo.

5.5 Conclusão

Esse capítulo apresentou os diagramas MSC (*Message Sequence Chart*) que resultaram de simulações das sinalizações de Registro (*Registration*) e Conexão (*Call Setup*). Essas

simulações tiveram o intuito de demonstrar o funcionamento da sinalizações especificadas, e apresentadas no capítulo 4, com o objetivo de sua validação.

Os diagramas MSC, dos casos de registro do móvel, mostraram o comportamento das trocas de sinalizações entre os dispositivos do sistema. Foi abordado, nesse caso, o uso do *Updated* [3], onde foi explicado sua importância, e o que pode ocorrer com a ausência dessa mensagem.

As simulações de conexão mostraram o caso onde o móvel que requereu a conexão estava no mesmo MSP que o terminal com o qual ele desejava se comunicar, apesar de ambos pertencerem a domínios diferentes. É mostrado em uma outra simulação, da sinalização de conexão, figuras 5.11, 5.12 e 5.13, o caso onde o terminal destino não é encontrado, acarretando no envio da mensagem *Target_not_found* para a unidade que pediu a conexão.

Os exemplos de falha na especificação apresentados tiveram como intuito, demonstrar como foram realizadas as análises e correções das especificações propostas utilizando a simulação e os gráficos MSC associados.

Capítulo 6

Conclusão

6.1 Considerações Gerais

Nesse trabalho foram especificadas as sinalizações de registro e conexão de um terminal em uma rede WATM. As sinalizações especificadas nesse trabalho foram obtidas da união das sinalizações descritas nos trabalhos [1], [20], [6], [14].

Para a especificação foi utilizada a linguagem formal SDL, *Specification and Description Language*, que é usada no desenvolvimento de produtos da área de telecomunicação. A especificação em SDL foi realizada com o auxílio da ferramenta CASE (*Computer Aided Software Engineering*) SDT (*SDL Design Tool*). O SDT permite executar uma simulação da especificação que apresenta as trocas de mensagens entre os dispositivos da rede WATM. A simulação apresenta gráficos temporais MSC (*Message Sequence Chart*), que foram de valiosa ajuda para detecção de falhas de especificação.

Durante as simulações da especificação de registro, para que se pudesse fazer uso do processo *Visiting_Register*, foi necessário usar mais de uma instância do processo **MobileSwitchingPoint**. Para simular esse caso, o cenário mínimo necessário é: dois comutadores (**MSP1** e **MSP2**), seis estações bases (**EB**), e um terminal móvel (**TM**) cujo domínio é MSP1. A simulação mostrou que se **MSP2** receber o pedido de registro do **TM** antes de **MSP1**, este envia a confirmação de registro antes de **MSP2**. Como os dados de **MSP2** chegam depois dos de **MSP1**, eles substituem os valores enviados, para **TM**, por **MSP1**. Para evitar esse fato, foi sugerido a inclusão na especificação do sinal *Updated* [3], capítulo 3, seção 3.4.1.

Nas simulações com a sinalização de conexão, foi observado que se houvesse vários nós para completar a chamada, e se a conexão, sob o critério de QoS (*Quality of Service*), fosse recusada pela estação base do terminal que pediu a conexão, haveria um fluxo de mensagens maior entre os nós para informar que a conexão foi recusada. Esse fluxo entre os nós ocorre pois, conforme proposto por [14], a estação base do terminal que origina a chamada deve ser o último dispositivo da rede a ser consultado a respeito do QoS. Na especificação da sinalização de conexão desta tese, foi proposto que, após a aceitação do QoS por parte do comutador que serve o terminal, seja verificada a aceitação por parte da estação que assiste ao terminal.

As propostas realizadas nas sinalizações de Registro e Conexão foram provenientes das análises realizadas nas simulações de validação de suas especificações formais. A versão do SDL utilizada aqui permite o uso de técnicas de orientação a objeto, portanto as especificações foram idealizadas de modo a facilitar uma migração para eventuais modificações necessárias e evolução em direção a uma padronização internacional.

6.2 Sugestões para trabalhos futuros em redes WATM

A partir desse trabalho, outros podem ser desenvolvidos, como:

- Estudo de performance das sinalizações propostas de Registro e Conexão;
- Especificação da sinalização de *Handoff*: com essa sinalização o terminal móvel seria capaz de uma mobilidade maior que a definida pela área de cobertura da célula onde ele registrou-se;
- Estudo e desenvolvimento de um algoritmo para roteamento: caso esse algoritmo preveja o uso de *handoff*, deve ser especificado como a informação destinada para uma determinada estação móvel que mudou de célula será redirecionada para chegar ao destino;
- Especificação das sinalizações de Registro e Conexão para a rede BAHAMA: como os dispositivos dessa rede são semelhantes aos da rede WATM utilizada nessa tese, a especificação pode ser facilitada se, utilizando técnicas de herança e reuso a partir das estruturas (*System-Block-Process-Procedures Types*) definidas nessa tese;
- Proposta das adequações necessárias nas camadas de protocolos das redes ATM para suportarem as funções associadas à mobilidade dos terminais de uma rede WATM.

Apêndice A

Notação e Conceitos em SDL usados na Especificação da rede WATM

A.1 Introdução

Em SDL, os tipos de dados da linguagem de programação são chamados de tipo (*sort*). O SDL define alguns tipos, e possibilita a especificação de novos tipos. Os tipos predefinidos não são estritamente necessários, mas têm sido definidos para o benefício dos usuários. Isso permite que os usuários usem os dados do mesmo modo como estão acostumados a utilizar em linguagens de programação. Abaixo estão listados alguns tipos predefinidos no SDL:

- Integer - Esse tipo cobre o conjunto dos números inteiros de menos infinito a infinito. Esse tipo utiliza todos os operadores aritméticos (+), (-), (*) e (÷), além dos operadores utilizados para comparação como por exemplo (=), (>) e (<).
- Natural - Esse tipo cobre o conjunto dos números inteiros positivos de zero à infinito. Esse tipo utiliza todos os operadores aritméticos (+), (-), (*) e (÷), além dos operadores utilizados para comparação como por exemplo (=), (>) e (<).
- Real - Esse tipo cobre o conjunto dos números reais de menos infinito a mais infinito. Esse tipo utiliza todos os operadores aritmédicos (+), (-), (*) e (÷), além dos operadores utilizados para comparação como por exemplo (=), (>) e (<).
- Boolean - Esse tipo é assume somente valores lógicos, *true* (verdadeiro) e *false* (falso). Utiliza os operadores **not**, **xor**, **or** e **and**, além dos operadores comparativos (=), (/=) (diferente) e (=>).
- Charater - Esse tipo cobre os caracteres ASCII como *A*, *b*, *CR*, *LF*e *FF*. Os operadores utilizados para comparação podem ser utilizados nesse tipo.
- Charstring - Esse tipo abrange um conjunto de caracteres, ou mesmo apenas um caracter, um exemplo desse tipo é a palavra *Ola*, ou simplemente a letra *O*. Os operadores desse tipo são *first*, *last*, // (concatenação) e *substring*.

- PId - Esse tipo caracteriza a identidade de uma instância de um processo. Possui os operadores (:=), (=) e (/=).

O SDL também permite que sejam definados tipos de estruturas como no exemplo abaixo:

newtype Pessoa **struct**
nm Charstring;
rg Integer;
fm Boolean;
endnewtype Pessoa;

Esse exemplo define uma estrutura que define uma *Pessoa* com os campos *nm* (nome), *rg* (registro geral), e *fm* (feminino é *true* e masculino é *false*).

Um tipo matriz também pode ser definido usando o tipo predefinido *array*, como no exemplo abaixo:

newtype Caixa
Array (Character, Integer)
endnewtype Caixa;

A tipo definido como *Caixa*, teve sua indexação feita via um tipo *Character*. Os valores do tipo *Caixa* são do tipo *Integer* e são acessados, como em uma programação de linguagem, escrevendo *b(z)*, onde **b** é uma variável do tipo *Caixa*, **z** é um tipo *Character* e *b(z)* é do tipo *Integer*.

Algumas vezes é útil usar um tipo em diferentes contextos. Por motivos de clareza, um tipo pode ter que dê mais informação sobre o uso dele. A construção **syntype** permite o uso de um tipo já existente com um nome diferente, por exemplo:

syntype Int = Integer; endsyntype;

A construção **syntype** também pode ser utilizado para restringir a escala de um tipo já existente, por exemplo:

syntype Smallint = Int **constants** 0:1000;
endsystype;

Esse apêndice apresenta na seção A.2 todas as estruturas definidas para o sistema WATM. Cada estrutura é apresentada em uma subseção específica.

A seção A.3 contem todas as mensagens que trafegam no sistema WATM especificado. Os sinais são mostrados em subseções cujo título é o nome do bloco onde o sinal foi definido.

A seção A.4 mostra os novos tipos que foram definidos para o sistema WATM. Esses tipos não fazem parte da linguagem SDL padrão, contudo foram criados para suprir uma necessidade do sistema.

A seção A.5 apresenta os tipos de matrizes difinidos para o sistema WATM. Cada tipo de matriz é mostrado en uma subseção diferente na seção A.5.

A última seção, seção A.6, contem todas as variáveis utilizadas pelos processos e procedimentos do sistema. Cada variável é apresentada dentro de uma subseção cujo título é o nome do processo ou procedimento onde ela foi definida.

A.2 Estruturas

A.2.1 Estrutura *cell*

Essa estrutura é global ao sistema, isso implica que ela pode ser utilizada em qualquer *Process Type* do bloco. é composta das variáveis abaixo:

- Seq - Nesse campo a estação base grava a potência com que ela recebeu o pedido de registro. Ele é do tipo *Natural*.
- BId_S - Esse campo armazena o identificador do processo (PId) da estação base, ou seja, identifica a estação base. Ele é do tipo *PId*. Ele é preenchido durante o registro do móvel e durante a conexão. Durante esse último evento, ele armazena o PId da estação do móvel que fez o pedido de conexão (*Source*).
- DId_S - Esse campo armazena o identificador do processo (PId) do domínio. Ele é do tipo *PId*. Ele é preenchido durante o registro do móvel e durante a conexão. Durante esse último evento, ele armazena o PId do domínio do móvel que fez o pedido de conexão (*Source*). Durante o registro, ele identifica o domínio ao qual o móvel pertence.
- MId_S - Esse campo armazena o identificador do processo (PId) do móvel. Ele é do tipo *PId*. Ele é preenchido durante o registro do móvel e durante a conexão. Durante esse último evento, ele armazena o PId .do móvel que fez o pedido de conexão (*Source*). Durante a conexão, ele identifica o móvel que pediu o registro.
- BId_T - Esse campo armazena o identificador do processo (PId) da estação base. Ele é do tipo *PId*. Ele é preenchido apenas durante a conexão. Ele armazena o PId da estação do móvel que recebeu o pedido de conexão (*Target*).
- DId_T - Esse campo armazena o identificador do processo (PId) do domínio. Ele é do tipo *PId*. Ele é preenchido apenas durante a conexão. Ele armazena o PId do domínio do móvel que recebeu o pedido de conexão (*Target*).
- MId_T - Esse campo armazena o identificador do processo (PId) do móvel. Ele é do tipo *PId*. Ele é preenchido apenas durante a conexão. Ele armazena o PId do móvel que recebeu o pedido de conexão (*Target*).
- resource - Esse campo armazena a classe de serviço necessária para a conexão do terminal. Ele é do tipo *cell*

Essa estrutura foi criada para transportar os dados do móvel. Tanto para o registro como para a conexão.

A.2.2 Estrutura *Home_struct*

Essa é uma estrutura global ao sistema. É composta das variáveis abaixo:

- MId - Esse campo armazena o identificador do processo (PId) do móvel. Ele é do tipo *PId.*
- BId - Esse campo armazena o identificador do processo (PId) da estação base. Ele é do tipo *PId.*
- DId - Esse campo armazena o identificador do processo (PId) do domínio. Ele é do tipo *PId.*
- Signal_Power - Esse campo armazena a potência com que a estação base recebeu o pedido de registro. Ele é do tipo *Natural.*

Essa estrutura foi criada para o registro do móvel.

A.2.3 Estrutura *PIds_struct*

Essa é uma estrutura global para o sistema. É composta das variáveis abaixo:

- Index - Esse campo é do tipo *Natural* e tem como função criar um número para indexar o segundo campo dessa estrutura.
- PIds - Esse campo tem como objetivo armazenar PIds. É do tipo *PId.*

A.2.4 Estrutura *Table_Struct*

Essa estrutura é definida no **Process Base_Id_Table**. Ele tem como objetivo fazer parte de uma tabela cujo o índice é um número natural e o elemento é essa estrutura. Para isso ela possui os dois campos abaixo do tipo *PId*:

- Base - Esse campo é designado para armazenar o PId de uma instância do processo BaseStation.
- MSP - Esse campo é designado para armazenar o PId de uma instância do processo MobileSwitchingPoint.

Dessa forma, cada instância do processo BaseStation, possuirá uma instância do **Process TypeMobileSwitchingPoint** correspondente.

A.3 Mensagens

A.3.1 Mensagens do System WATM

Nessa subseção serão apresentadas todas as mensagens definidas no **System WATM**. Por estarem definidas no sistema, essas mensagens têm caráter global, isto é, podem trafegar por qualquer dos processos que compõem o sistema.

- Verify_blocking - Essa mensagem é enviada da estação base destino para o terminal móvel destino da conexão. Ao recebê-la, o terminal verifica se o terminal que originou a chamada consta na sua lista de bloqueio. Para simplificar o sistema, foi desconsiderado lista de bloqueio, entretanto a existência da mensagem possibilita a criação desta lista. A mensagem *Verify_blocking* é do tipo *cell*.

- Connection_request - Essa mensagem, que transporta a estrutura *cell*, faz o pedido de conexão do móvel para a estação base.

- Connection_Setup - Essa mensagem, que transporta a estrutura *cell*, inicia o processo de verificação de recursos, para a conexão, no dispositivo que a recebe.

- Connection_refused - Essa mensagem informa que, a conexão pedida, foi recusada. Ela transporta a estrutura *cell*.

- Connection_location - Essa mensagem é enviada do MSP para o DLS. Seu objetivo é pedir a localização do terminal móvel destino. Ela é do tipo *cell*.

- Connection_located - Essa mensagem informa ao MSP a localização do terminal móvel destino. Ela é do tipo *cell*.

- Connection_complete - Essa mensagem informa que, a conexão pedida, foi completada. Ela transporta a estrutura *cell*.

- RegisterMobile - Essa mensagem é enviada do **ambiente** para o **Process Signal_Manager**. Ela informa qual a instância do **Process Type Mobile** irá se registra, e qual a instância do **Process Type MobileSwitchingPoint** será o domínio. Ela transporta dois PId's, sendo então do tipo *PId, PId*.

- Register_request - Essa mensagem, que transporta a estrutura *cell*, tem como objetivo requer o pedido de registro do móvel. é enviada do móvel para a estação base.

- Register_request_ack - Essa mensagem tem como objetivo informar ao móvel o recebimento, pela estação base, do pedido de registro. Ela transporta a estrutura *cell*.

- Register_Base - Essa mensagem é enviada da estação base para o MSP, e do MSP para o DLS . Ela requisita o registro de um terminal móvel na rede WATM. Ela transporta a estrutura *cell*.

- Register - Essa mensagem é enviada do DLS para o MSP informando que o registro do terminal móvel foi efetuado. Ela transporta a estrutura *cell.*

- Register_complete - Essa mensagem informa, que o pedido de registro, foi aceito. Ela transporta a estrutura *cell.*

- Target_not_found - Essa mensagem informa que o terminal móvel destino não foi localizado. Ela trafega por todos os componentes da rede WATM. Ela transporta a estrutura *cell.*

- InitializeSystem - Essa mensagem é utilizada para montar a rede WATM. Ela indica o número de terminais móveis, estações base, e MSP's que a rede terá. Ela transporta, então, três números do conjunto dos naturais, por isso ela é do tipo *Natural, Natural, Natural.*

- Initialize_DLS - Essa mensagem é enviada do MSP para o DLS sempre que uma instância do **Process Type MobileSwitchingPoint** é criada. Ela determina a criação de uma instâncias dos **Process Type Home_Register** e **Process Type Visiting_Register** que estará associada a instância do **Process Type MobileSwitchingPoint** criada, cujo PId é enviado na mensagem. No DLS é criado uma tabela que associa essas três instâncias criadas. A mensagem *Initialize_DLS* é do tipo *PId.*

- Ask_Bases - Essa mensagem requisita o PId de três estações móveis para o móvel se registrar.

- Bases - Essa mensagem transporta o PId de cada uma das três estações base candidatas a serem servidoras do móvel que pediu registro, devido a isso ela é do tipo *PId, PId, PId.*

- Callup - Essa mensagem é enviada do **ambiente** para o **Process Signal_Manager.** Ela informa quais instâncias do **Process Type Mobile** representarão a origem e o destino da conexão, e qual a classe de QoS desejada para a conexão. Ela transporta dois PId's e um número natural, sendo então do tipo *PId, PId, Natural.*

- Reject - Essa mensagem, enviado do DLS para o MSP, informa ao MSP que o pedido de registro foi rejeitado.

- MSP_Begin - Essa mensagem informa ao processo **Process Base_Id_Table** que terá início a criação das instâncias do **Process Type MobileSwitchingPoint.**

- MSP_End - Essa mensagem informa ao processo **Process Base_Id_Table** que foi concluída a criação das instâncias do **Process Type MobileSwitchingPoint.**

- MSP_PId - Essa mensagem informa ao processo **Process Base_Id_Table** o PId de cada instância do **Process Type MobileSwitchingPoint** criada. Com os dados dessa mensagem, o processo **Base_Id_Table** monta uma tabela onde cada instância

do **Process Type BaseStation** criada é associada a uma instância do **Process Type MobileSwitchingPoint** recebida nesta mensagem. A mensagem *MSP_PId* é do tipo *PId*.

- Updated - Essa mensagem é enviada do DLS para o MSP informando quais os dados antigos e novos de registro do terminal móvel registrado. Cada dado de registro está contido na estrutura *Home_struct*, como essa mensagem transporta dois dados de registro, o velho e o atual, ela é do tipo *Home_struct, Home_struct*.

A.3.2 Mensagens do Block Type Mobile

Nessa subseção serão apresentadas todas as mensagens definidas no **Block Type Mobile**. Por estarem definidas no bloco, essas mensagens têm caráter local, isto é, só podem trafegar pelos processos que compõem esse bloco.

- Which_Base - No momento do registro, o **Process Mobile_Id_Table** associa cada instância do **Process Type Mobile** registrado a instância do **Process Type BaseStation** com a qual ele deve enviar as mensagens. A mensagem *Which_Base* verifica qual instância do **Process Type BaseStation**, o **Process Type Mobile** deve mandar suas mensagens. O conteúdo dessa mensagem é o PId de uma instância do **Process Type Mobile**, por isso ela é do tipo *PId*.

- Base - Essa mensagem transporta o PId do **Process Type BaseStation** ao qual um determinado **Process Type Mobile** está associado. É a resposta da mensagem *Which_Base*.

- Data_written - Essa mensagem informa a uma instância do **Process Type Mobile** que os dados de registro dessa instância foram armazenados em uma tabela do **Process Mobile_Id_Table**.

- Register_data - Essa mensagem envia para o **Process Mobile_Id_Table** os PId dos **Process Type Mobile** e **Process Type BaseStation** que foram registrados. Ela é do tipo *PId, PId*.

A.3.3 Mensagens do Block Type BaseStation

Nessa subseção serão apresentadas todas as mensagens definidas no **Block Type BaseStation**. Por estarem definidas no bloco, essas mensagens têm caráter local, isto é, só podem trafegar pelos processos que compõem esse bloco.

- Which_MSP - No momento da inicialização do **System WATM**, o **Process Base_Id_Table** associa cada instância do **Process Type BaseStation** com uma do **Process Type MobileSwitchingPoint** para o qual ele deve enviar as mensagens.

A mensagem *Which_ MSP* verifica qual instância do **Process Type MobileSwitchingPoint**, o **Process Type BaseStation** deve mandar suas mensagens. O conteúdo dessa mensagem é o PId de uma instância do **Process Type BaseStation**, por isso ela é do tipo *PId*.

- The_MSP - Essa mensagem transporta o PId do **Process Type MobileSwitchingPoint** ao qual um determinado **Process Type BaseStation** pode se comunicar. É a resposta da mensagem *Which_MSP*.

- Base_Id - Essa mensagem transporta a estrutura *PIds_struct* do procedimento InitializaSystemII para o processo Base_Id_Table. É usada no instante da criação das instâncias do **Process Type BaseStation**, e carrega o número da estação base criada e PId da instância.

A.3.4 Mensagens do Block Type DomainLocationService

Nessa subseção serão apresentadas todas as mensagens definidas no **Block Type DomainLocationService**. Por estarem definidas no bloco, essas mensagens têm caráter local, isto é, só podem trafegar pelos processos que compõem esse bloco.

- data_not_found - Essa mensagem é enviada de uma instância do **Process Type Home_Register** para o **Process Type DomainLocationService** informando que os dados do terminal móvel destino não encontram-se naquela instância. Essa mensagem é do tipo *cell*.

- data_inserted - Essa mensagem é enviada ou de uma instância do **Process Type Home_Register**, ou de uma instância do **Process Type Visiting_Register**, para o **Process Type DomainLocationService** informando que os dados do terminal móvel que pediu registro foram armazenados. Essa mensagem é do tipo *Home_struct*.

- write_data - Essa mensagem é enviada do **Process Type DomainLocationService** para a instância ou do **Process Type Home_Register**, ou do **Process Type Visiting_Register**, informado que os dados de registro do terminal móvel devem ser armazenados. Essa mensagem é do tipo *Home_struct*.

- read_data - Essa mensagem é enviada do **Process Type DomainLocationService** para a instância ou do **Process Type Home_Register**, ou do **Process Type Visiting_Register**, para obter os dados de localização do terminal destino. Essa mensagem é do tipo *Home_struct*.

- checking - Quando um terminal móvel tenta registrar-se com visitante, o **Process Type DomainLocationService** envia a mensagem *checking* para a instância do **Process Type Home_Register** que atua como domínio do terminal para que seja aprovado, ou rejeitado, o registro desse terminal móvel como visitante. Essa mensagem é do tipo *Home_struct*.

- visiting - Essa mensagem é enviada do **Process Type Home_Register** para o **Process Type DomainLocationService** quando os dados de registro de um terminal móvel poderem ser inseridos como visitante. Essa mensagem é do tipo *Home_struct.*

- rejected - Essa mensagem é enviada do **Process Type Home_Register** para o **Process Type DomainLocationService** quando os dados de registro de um terminal móvel são rejeitados. Essa mensagem é do tipo *Home_struct.*

- Update - Essa mensagem é enviada do **Process Type DomainLocationService** para uma instância do **Process Type Visiting_Register** informando quais os dados antigos e novos de registro do terminal móvel registrado para que eles possam ser atualizados. Cada dado de registro está contido na estrutura *Home_struct*, como essa mensagem transporta dois dados de registro, o velho e o atual, ela é do tipo *Home_struct, Home_struct.*

A.4 Definição de Novos Tipo

O tipo definido como *SmallNat*[1], é um subconjunto do conjunto de números naturais. Esse subconjunto está definido em dois procedimentos do sistema: Register_request, do processo BaseStation; e no processo Base_id_Table.

Em cada uma dessas ocasiões, esse subconjunto tem a quantidade de elementos diferentes. No primeiro caso ele possui 10 elementos, do número 1 ao 10. No segundo, ele tem 4 elementos, do número 2 ao 5.

O objetivo desse subconjunto é atribuir a variáveis do tipo *Natural* números aleatórios pertencentes a esse subconjunto.

A.5 Matrizes

Todas as matrizes definidas para a especificação das sinalizações de Registro e Conexão receberam o nome de *Table*, sendo diferenciadas pelo número que segue o nome. Todas as matrizes com o mesmo nome são iguais, independente do local onde foram definidas. Foram definidas quatro matrizes que serão comentadas nas subseções abaixo.

A.5.1 Matriz *Table*

A matriz *Table* possui o índice *PId* e o conteúdo *PId.* Ela foi criada para associar o PId de duas instâncias de processo, por exemplo associar uma instância do *Process Type Mobile* com uma instância do *Process Type BaseStation.* Essa matriz é utilizada nos **Process Mobile_Id_Table** e **Process Type DomainLocationService.**

[1]Small Natural

A.5.2 Matriz *Table1*

A matriz *Table1* possui o índice *Natural* e o conteúdo *Table_struct*. A estrutura *Table_struct*foi definida na seção A.2.4. A matriz *Table1* foi criada armazenar os dados da estrutura *Table_struct* segundo uma ordem numérica, dessa forma a estrutura pode ser consultada utilizando um número natural. Essa matriz é utilizada no **Process Base_Id_Table**.

A.5.3 Matriz *Table2*

A matriz *Table2* possui o índice *Natural* e o conteúdo *PId*. Essa matriz é utilizada no **Process Type DomainLocationService**.

A.5.4 Matriz *Table3*

A matriz *Table3* possui o índice *PId* e o conteúdo *Home_struct*. A estrutura *Home_struct*foi definida na seção A.2.2. A matriz *Table2* foi criada armazenar os dados da estrutura *Table_struct* associada ao PId de uma instância de processo específica.. Essa matriz é utilizada nos **Process Type Home_Register** e **Process Type Visiting_Register**.

A.6 Variáveis

A.6.1 Variáveis do Process Signal_Manager

Abaixo estão descritas as variáveis utilizadas no **Process Signal_Manager**.

- V1, V2 e V3 - Essas três varáveis são do tipo *Natural*. Elas têm como função receber, da mensagem *InitializeSystem*, e enviam posteriormente na mesma mensagem, o número de terminal móveis, estações base, e MSP's, respectivamente, do **System WATM**. A variável *V1*, quando usada na mensagem *CallUp*, recebe/envia o valor da classe de QoS desejada na conexão.

- PId1 e PId2 - Essas duas variáveis são do tipo *PId*. Na mensagem *RegisterMobile* elas recebem/enviam a instância do **Process Type Mobile** irá se registra, e a instância do **Process Type MobileSwitchingPoint** que será o domínio, respectivamente. Quando usadas na mensagem *CallUp* elas recebem/enviam as instância do **Process Type Mobile** que representarão, respectivamente, a origem e o destino da conexão.

A.6.2 Variáveis do Process Mobile_Id_Table

Abaixo estão descritas as variáveis utilizadas no **Process Mobile_Id_Table**.

- MId_Table - Essa variável é uma matriz do tipo *Table*. A definição de uma matriz do tipo *Table* será apresentada na seção A.5.1.

- Mobile - Essa variável é do tipo *PId* . Ela recebe, em qualquer mensagem que ela apareça, o PId de uma instância do **Process Type Mobile**.

- Bases - Essa variável é do tipo *PId* . Ela recebe, em qualquer mensagem que ela apareça, o PId de uma instância do **Process BaseStation**.

A.6.3 Variáveis do Process Type Mobile

As variáveis definidas no **Process Type Mobile** podem ter seus valores utilizados pelos procedimentos definidos dentro desse processo.

- Cell_WATM - Essa variável é do tipo *cell*, seção A.2.1. A variável *Cell_WATM* como objetivo receber o conteúdo das mensagens: *Register_complete*; *Verify_blocking* e *Register_request_ack*.

- Source - Essa variável, declarada em qualquer dos processos, é do tipo *PId*. Ela, no procedimento de registro do móvel, é usada para receber o PId do processo do móvel que deseja de registrar. No procedimento de conexão, ela armazena o PId do processo do móvel que requer a chamada. Contudo, ela pode aparecer em algum procedimento como uma variável auxiliar para receber um PId de alguma mensagem, sendo aproveitada para evitar a declaração de mais uma variável.

- Domain - Essa variável, declarada em qualquer dos processos, é do tipo *PId*. Ela recebe o PId do domínio de um determinado móvel.

- Target - Essa variável, declarada em qualquer dos processos, é do tipo *PId*. Ela é usada para receber o PId do processo do móvel destino de uma conexão.

- V1 e V2 - Essas duas varáveis são do tipo *Natural*. Elas têm como função receber, da mensagem *InitializeSystem*, e enviam posteriormente na mesma mensagem, o númeor de estações base e MSP's, respectivamente, do **System WATM**. A variável *V1*, quando usada na mensagem *CallUp*, recebe o valor da classe de QoS desejada na conexão.

- New_Mobile - Essa variável identifica o número de móveis registrados no sistema.

- No_of_Mobile - Essa variável recebe da mensagem **InitializeSystem** o número de móveis que o sistema terá. Ela é do tipo *Natural*.

A.6.3.1 Procedure Callup

- Cell_WATM - Essa variável, do tipo *cell*, tem a função de armazenar os dados, do terminal móvel, necessários para a conexão. Esses dados são enviados na mensagem *Connection_request*. A variável *Cell_WATM* também é utilizada para receber os valores das mensagens *Connection_refused*, *Connectio_complete* e *Target_not_found*.

A.6.3.2 Procedure RegisterMobile

- BS1, BS2 e BS3 - Essas três varáveis são do tipo *PId*. Cada uma delas tem a função receber o PId de uma instância do **Process Type BaseStation** enviada pela mensagem *Bases*.

A.6.4 Process Base_Id_Table

- BaseId - Essa variável, do tipo *Table_struct*, é utilizada para armazenar o PId de uma instância do **Process Type BaseStation** e sua instância, do **Process Type MobileSwitchingPoint**, associada que são recebidos na mensagem *Base_Id*.

- BSId_Table - Essa variável, do tipo *Table1*, seção A.5.2, é uma matriz que guarda os valores das instâncias do **Process Type BaseStation** e suas instâncias, do **Process Type MobileSwitchingPoint**, associadas, indexadas por um número inteiro positivo.

- BS1 - Essa variável é do tipo *PId*. Ela tem a função receber o PId de uma instância do **Process Type BaseStation** enviada pela mensagem *Which_MSP*.

A.6.4.1 Procedure Which_MSP

- Index - Essa variável, do tipo *Natural*, é utilizada para varrer as posições da tabela **BSId_Table**.

- MSP - A variável **MSP** recebe o resultado da busca com a variável **Index**. A variável **MSP** é do tipo *PId*.

A.6.4.2 Procedure MSP_Begin

- MSP - Essa variável recebe um PId da mensagem *MSP_PId* e o armazena na matriz **BSId_Table**. A variável **MSP** é do tipo *PId*.

- Index - Essa variável, do tipo *Natural*, é um contador; sempre que a mensagem *MSP_PId* é recebida ele é acrescido de uma unidade. A variável **Index** é usada para determinar a posição da matriz **BSId_Table**, seção A.6.4, que armazenará a variável **MSP**

- Index1 - Essa variável, do tipo *Natural*, é utilizada na função de associar instâncias do **Process Type MobileSwitchingPoint** a instâncias do **Process Type BaseStation**.

A.6.4.3 Procedure Ask_Bases

- Index - Essa variável é do tipo *Natural*. Ela tem o objetivo de receber um valor aleatório do tipo *SmallNet*, seção A.4. Esse valor aleatório é utilizado para determinar as três estações base do registro do terminal móvel.

- BS1, BS2 e BS3 - Essas três varáveis são do tipo *PId*. Cada uma delas tem a função receber o PId da instância do **Process Type BaseStation** que o terminal móvel enviará o pedido de registro.

A.6.5 Process Type BaseStation

- Cell_WATM - Essa variável, do tipo *Cell*, é utilizadas para receber os valores das mensagens *Register_request*, *Register_complete*, *Connection_Setup*, *Connection_complete*, *Target_not_found*, *Connection_refused* e *Connection_request*.

- V1 e V2 - Essas duas varáveis são do tipo *Natural*. Elas têm como função receber da mensagem *InitializeSystem* o número de terminais móveis e MSP's, respectivamente, do **System WATM**.

- No_of_BS - Essa variável recebe da mensagem **InitializeSystem** o número de estações base que o sistema terá. Ela é do tipo *Natural*.

A.6.5.1 Procedure Register_request

- Aleatorio - Essa variável recebe um valor aleatório do tipo *SmallNet*, seção A.4. Esse valor aleatório é utilizado para determinar a estação base que registrará o terminal móvel. A variável **Aleatorio** é do tipo *Natural*.

- MSP - Essa variável, do tipo *PId*, recebe o PId da instância do **Process Type MobileSwitchingPoint**, pela mensagem *The_MSP*, e o envia pela mensagem *Register_Base*.

A.6.5.2 Procedure Register_request

- BaseStation - Essa variável é definida do tipo *PIds_struct*, seção A.2.3. A variável **BaseStation** recebe no campo **Index** o número da instância do **Process Type BaseStation** criada, e no campo **PIds** o PId dessa instância. A variável **BaseStation** é utilizada para enviar essas duas informações em uma mesma mensagem, *Base_Id*.

A.6.5.3 Procedure Connection_request

- MSP - Essa variável tem a função de receber um PId do **Process Type MobileSwitchingPoint**, pela mensagem *The_MSP*, e armazená-lo na em **Cell_WATM**, seção A.6.5. A variável **MSP** é do tipo *PId*.

A.6.6 Process Type MobileSwitchingPoint

- OldDomain e writedata - Essas variáveis são do tipo *Home_struct*, seção A.2.2. Elas têm a função de receber os valores da mensagem *Updated*.

- Cell_ATM - Essa variável, do tipo *Cell*, recebe o conteúdo das mensagens *Register_Base*, *Register*, *Connection_refused*, *Connection_request*, *Connection_located*, *Target_not_found*, *Connection_complete* e *Connection_Setup*.

- No_of_Mobile - Essa variável recebe da mensagem **InitializeSystem** o número de MSP's que o sistema terá. Ela é do tipo *Natural*.

A.6.6.1 Procedure Updated

- Cell_ATM - Essa variável do tipo *cell* tem o objetivo de receber os valores da variável **writedata** e enviá-los por meio da mensagem *Register_Complete*.

A.6.6.2 Procedure InitializeSystemI

- MSPPId - Essa variável recebe o PId da instância do **Process Type MobileSwitchingPoint** que foi criada. A variável **MSPPId** é do tipo *PId*.

- New_MSP - Essa variável, do tipo *Natural*, conta o número de instâncias do **Process Type MobileSwitchingPoint** que foram criadas.

A.6.7 Process Type Home_Register

- routing_table - Essa variável, do tipo *Table3*, seção A.5.4, é uma matriz que indexa um PId a estrutura *Home_struct*, seção ssec:EHS. A variável **routing_table** foi definida no **Process Type Home_Register** para que podesse ser alterada, ou consultada, pelos: **Procedure read_dataI**; **Procedure write_data**; **Procedure Checking**.

- Cell_ATM - Essa variável, do tipo *Cell*, é utilizada para receber os dados da mensagem *read_data*.

- writedata - Essa variável é do tipo *Home_struct*. Elas têm a função de receber os valores das mensagens *write_data* e *Checking*.

A.6.7.1 Procedure write_data

- OldDomain - Essa variável é do tipo *Home_struct*. Quando existe a necessidade do uso da mensagem *Updated*, é nessa variável que os dados de registro, do terminal móvel, que foram substituídos são armazenados.

A.6.8 Process Type Visiting_Register

- routing_table - Essa variável, do tipo *Table3*, seção A.5.4, é uma matriz que indexa um PId a estrutura *Home_struct*, seção ssec:EHS. A variável **routing_table** foi definida no **Process Type Visiting_Register** para que podesse ser alterada, ou

consultada, pelos: **Procedure write_dataI**; **Procedure Update**; **Procedure read_dataII**.

- Cell_ATM - Essa variável, do tipo *Cell*, é utilizada para receber os dados da mensagem *read_data*.

- OldDomain e writedata - Essas variáveis são do tipo *Home_struct*, seção A.2.2. Elas têm a função de receber os valores da mensagem *Updated*.

A.6.9 Process Type DomainLocationService

- HR_Table - Essa variável, do tipo *Table*, seção A.5.1, é uma matriz que associa cada instância do **Process Type MobileSwitchingPoint** a uma instância do **Process Type Home_Register**. A variável **HR_Table** foi definida no **Process Type DomainLocationService** para que podesse ser alterada, ou consultada, pelos: **Initialize_DLS**; **Procedure data_not_found** e **Procedure Register_Base**.

- VR_Table - Essa variável, do tipo *Table*, seção A.5.1, é uma matriz que associa cada instância do **Process Type MobileSwitchingPoint** a uma instância do **Process Type Visiting_Register**. A variável **VR_Table** foi definida no **Process Type DomainLocationService** para que podesse ser alterada, ou consultada, pelos: **Initialize_DLS** e **Procedure read_data**.

- MSPs_Table - Essa variável, do tipo *Table2*, seção A.5.3, é uma matriz que associa cada instância do **Process Type MobileSwitchingPoint** a número inteiro positivo. A variável **MSPs _Table** foi definida no **Process Type DomainLocationService** para que podesse ser alterada, ou consultada, pelos: **Initialize_DLS** e **Procedure read_data**.

- Index - Essa variável é do tipo *Natural*. Ela é utilizada somente no **Procedure Initizalize_DLS**. A variável *Index* foi definida no **Process Type DomainLocationService**, para que as alterações no seu valor, realizadas pelo **Procedure Initizalize_DLS**, fiquem inalteradas após a execução do procedimento.

- IdxSearch - Essa variável é do tipo *Natural*. Ela é utilizada somente no **Procedure data_not_found**. A variável *Index* foi definida no **Process Type DomainLocationService**, para que as alterações no seu valor, realizadas pelo **Procedure Initizalize_DLS**, fiquem inalteradas após a execução do procedimento.

- Cell_ATM - Essa variável, do tipo *Cell*, é utilizada para receber os dados das mensagens *Connection_location*; *read_data*; *data_not_found*; *Register_Base*.

- get_sender - Essa variável recebe o conteúdo da mensagem *Initialize_DLS*. A variável **get_sender** é do tipo *PId*.

- OldDomain e writedata - Essas variáveis são do tipo *Home_struct*, seção A.2.2. Elas têm a função de receber os valores da mensagem *Update*. Além do valor da mensagem *Update*, a variável **writedata** também recebe os valores das mensagens *rejected* e *visiting*.

- datainserted - Essa variável é do tipo *Home_struct*. Ela tem a função de receber o conteúdo da mensagem *data_inserted*.

A.6.9.1 Procedure data_inseted

- Cell_ATM - Essa variável, do tipo *cell*, é utilizada para receber em seus campos **MId_S**, **BId_S** e **DId_S** os respectivos valores dos campos **MId**, **BId** e **DId** da variável **datainserted**, seção A.6.9, para enviá-los na mensagem *Register*.

A.6.9.2 Procedure Register_Base

- writedata - Essa variável é do tipo *Home_struct*. Ela armazena nos seus campos **MId**, **BId** e **Signal_Power** os respectivos campos **MId_S**, **BId_S** e **Seq** da variável **Cell_ATM**, seção A.6.9. Ao campo **DId**, da variável **writedata**, é atribuído o PId da instância de processo que enviou a mensagem *Register_Base*. O conteúdo dessa variável é enviado pelas mensagens *checking* e *write_data*.

Apêndice B

Processos e Procedimentos Auxiliares da Especificação

Durante a especificação das sinalizações de Registro e Conexão, capítulo 4, alguns processos e procedimentos foram criados para possibilitar a simulação de uma rede WATM apresentada nesse trabalho. Esses processos e procedimentos não foram mostrados no capítulo 4 por não fazerem parte da especificação das sinalizações propriamente ditas, portanto são apresentadas aqui. Na figura B.1 é apresentado uma árvore com todos os blocos, processos e procedimentos do sistema WATM; os processos e procedimentos que compõem este apêndice estão indicados com uma seta.

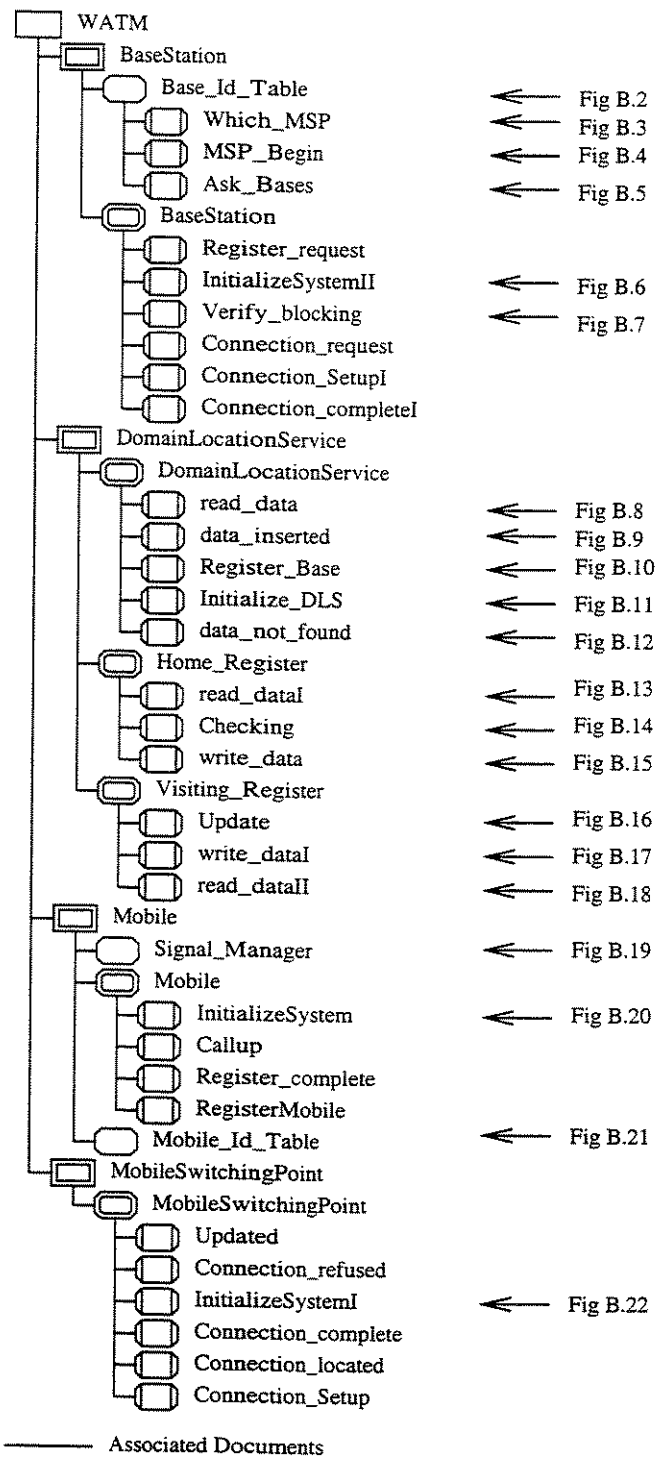

Figura B.1: Árvore do System WATM - Organizer

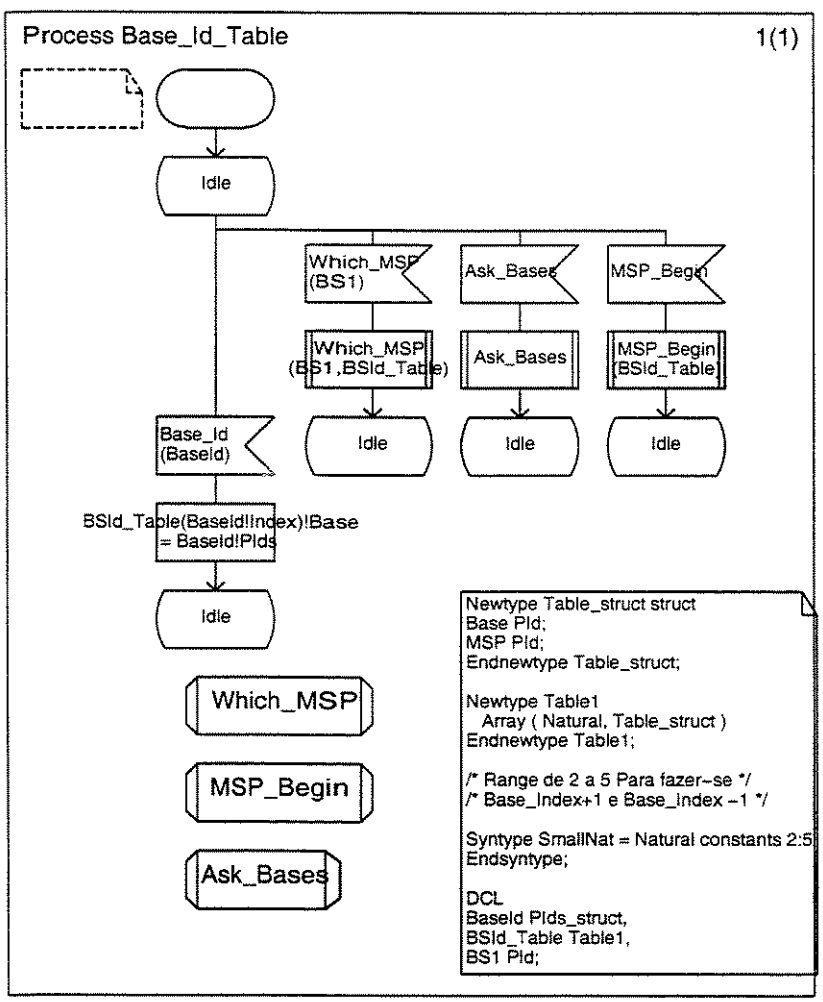

Figura B.2: Processo Base_Id_table/Bloco BaseStation

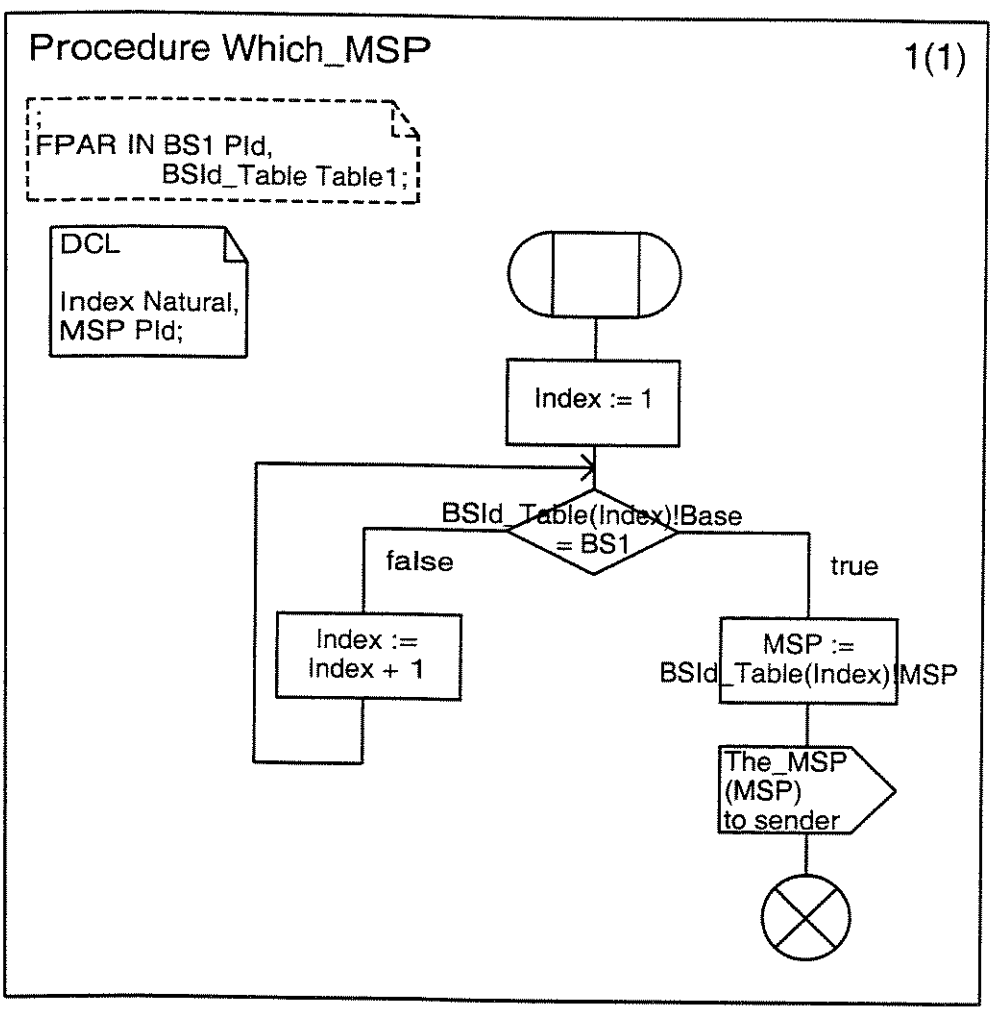

Figura B.3: Procedimento Which_MSP/Processo Base_Id_table/Bloco BaseStation

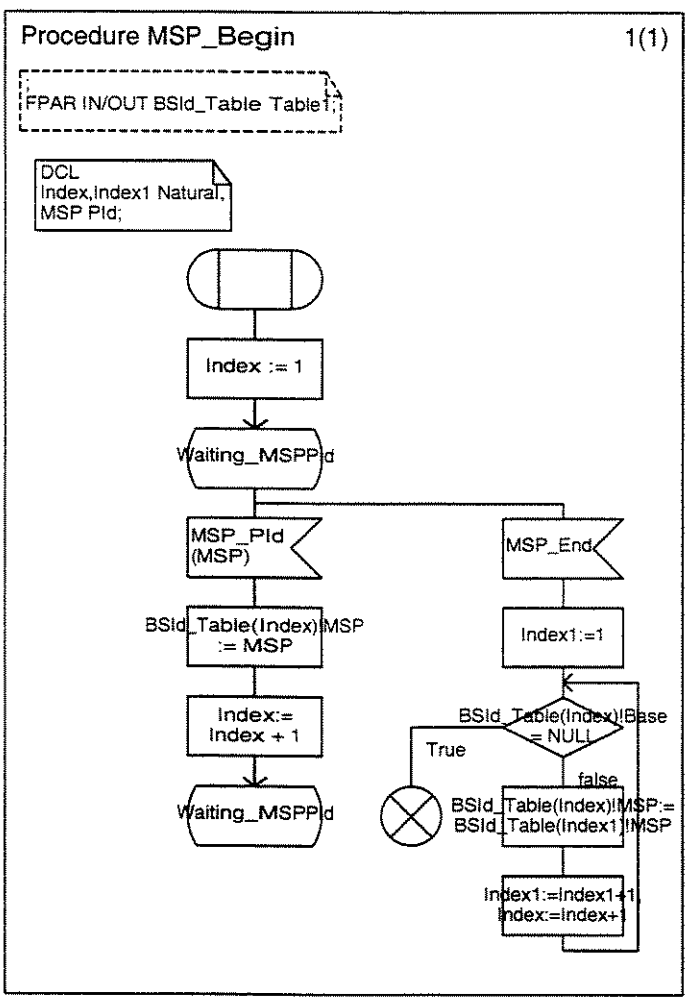

Figura B.4: Procedimento MSP_Begin/Processo Base_Id_table/Bloco BaseStation

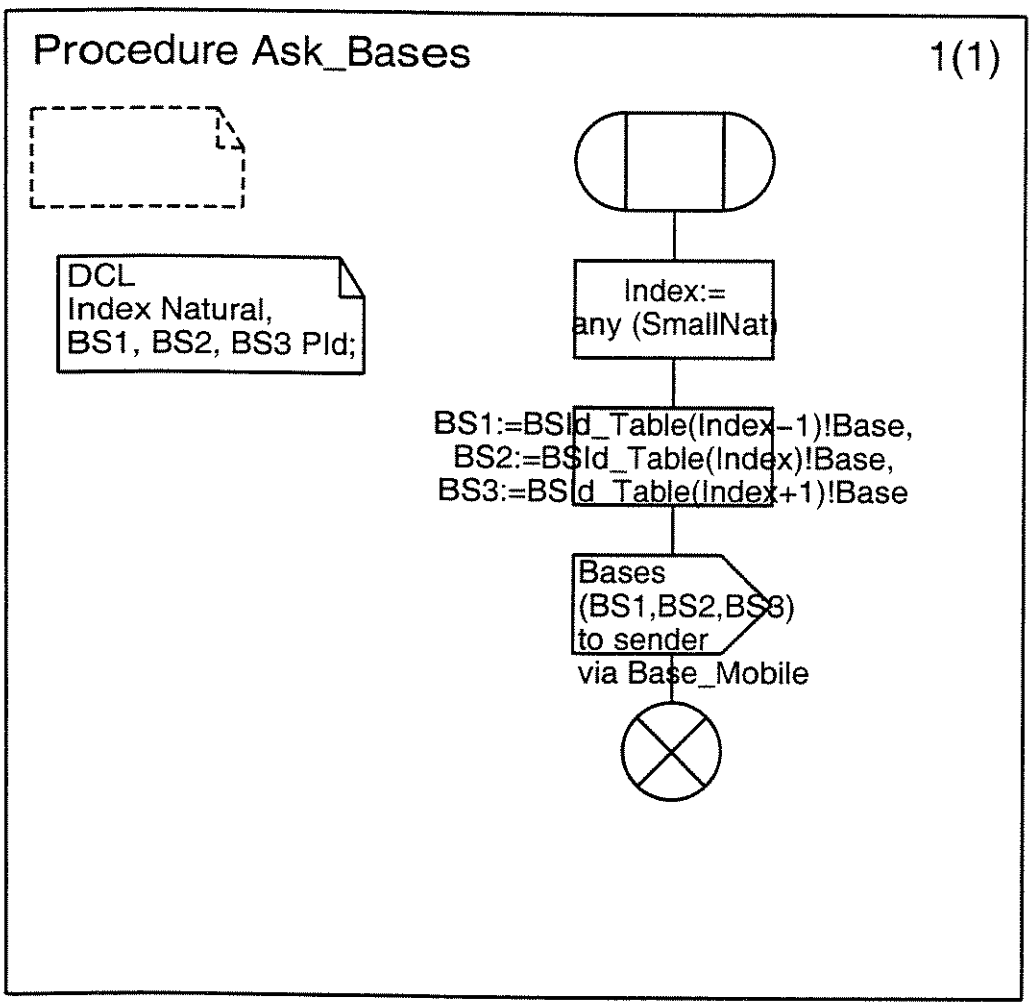

Figura B.5: Procedimento Ask_Bases/Processo Base_Id_table/Bloco BaseStation

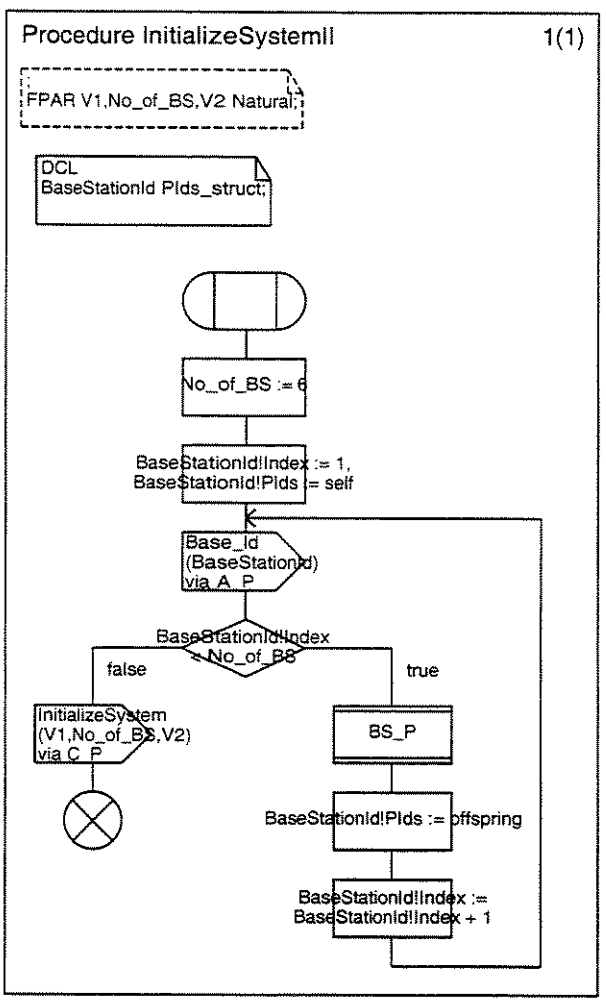

Figura B.6: Procedimento InitializeSystemII/Processo BaseStation/Bloco BaseStation

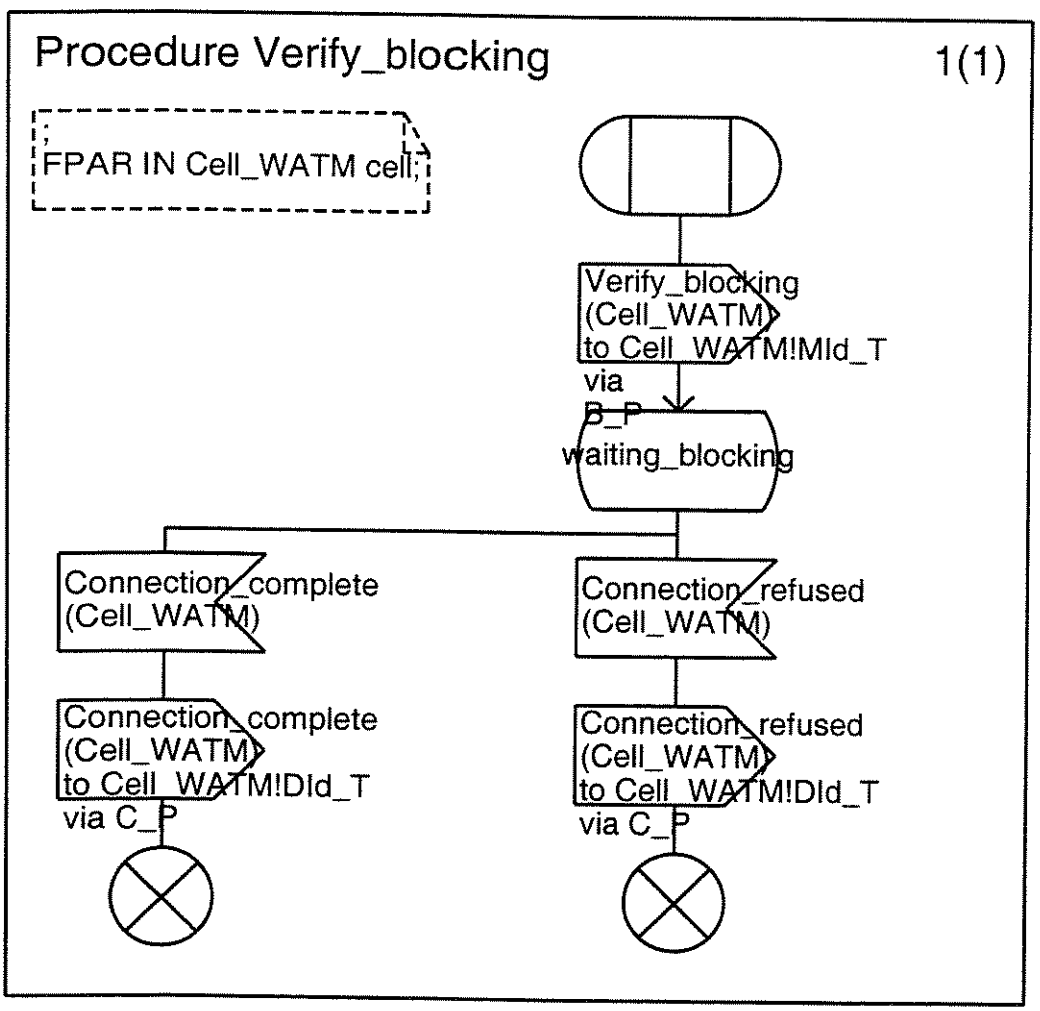

Figura B.7: Procedimento Verify_blocking/Processo BaseStation/Bloco BaseStation

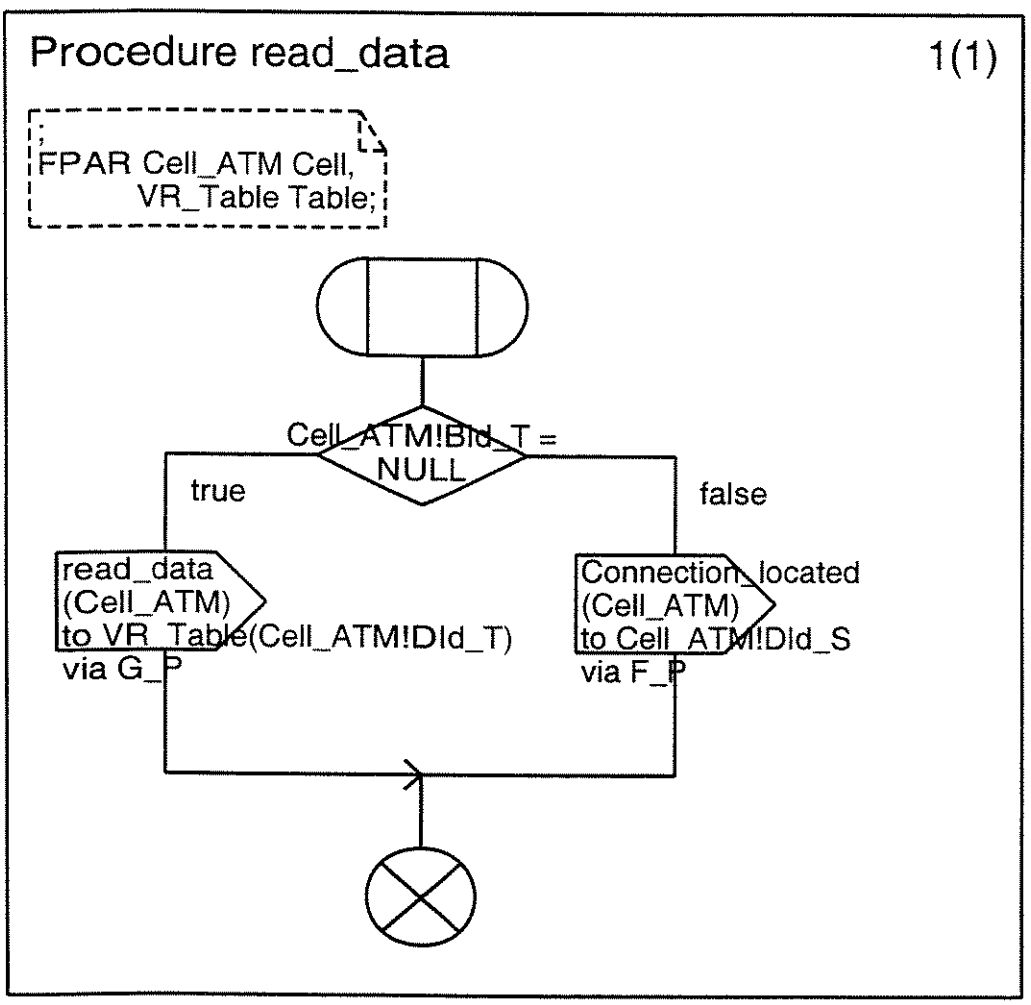

Figura B.8: Procedimento read_data/Processo DomainLocationService/Bloco DomainLocationService

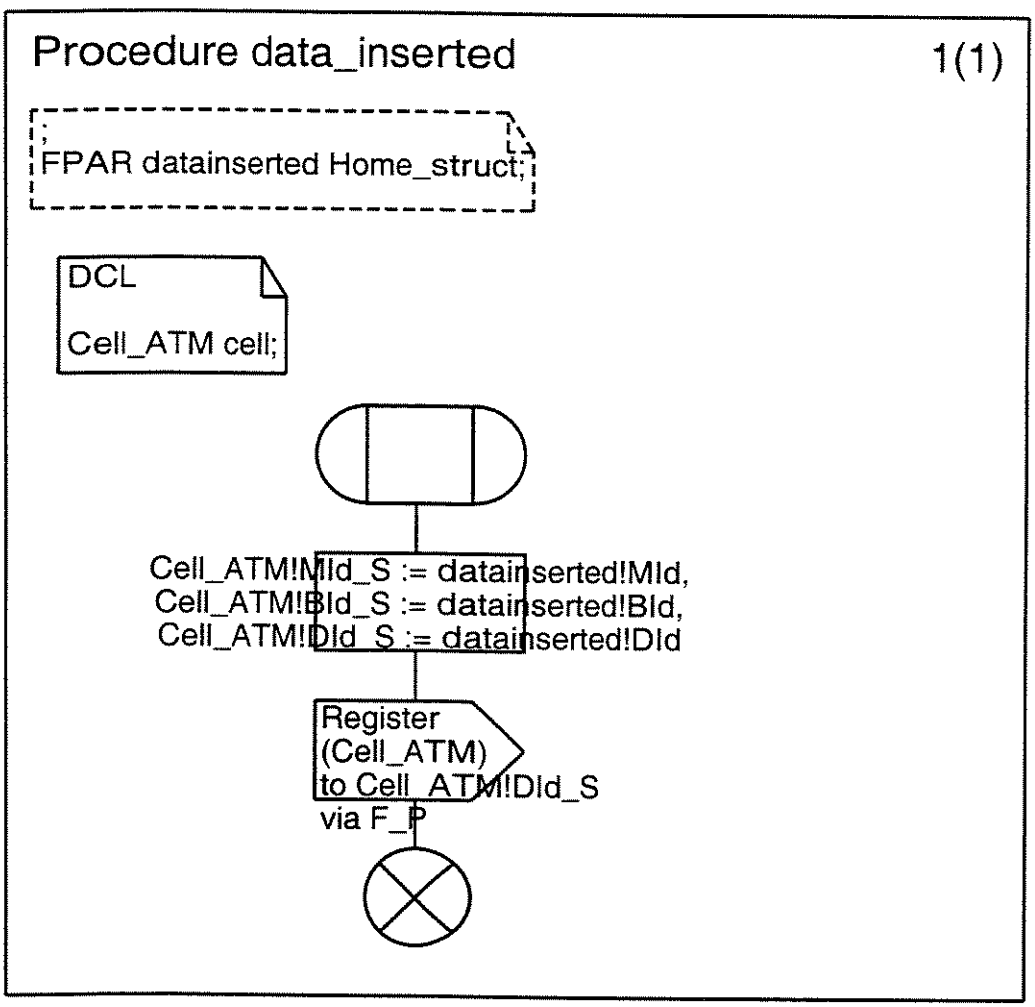

Figura B.9: Procedimento data_inserted/Processo DomainLocationService/Bloco DomainLocationService

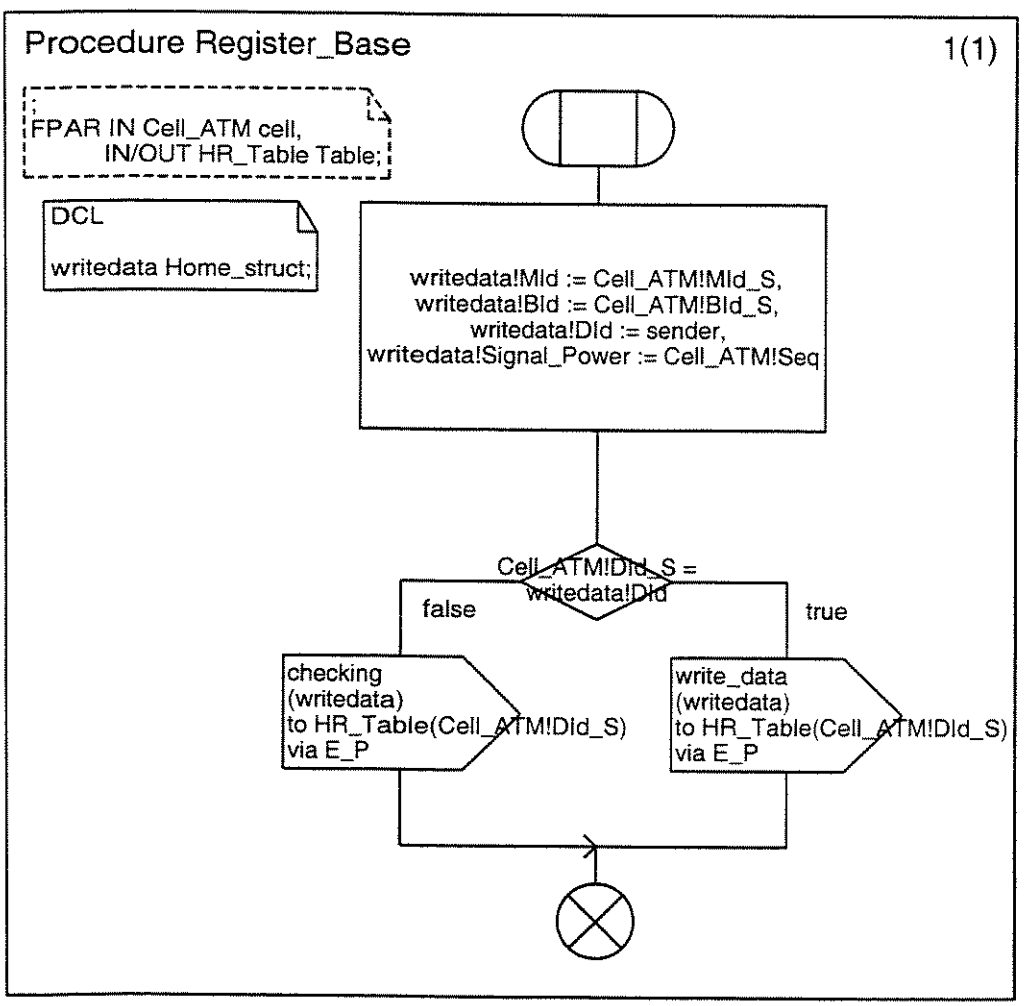

Figura B.10: Procedimento Register_Base/Processo DomainLocationService/Bloco DomainLocationService

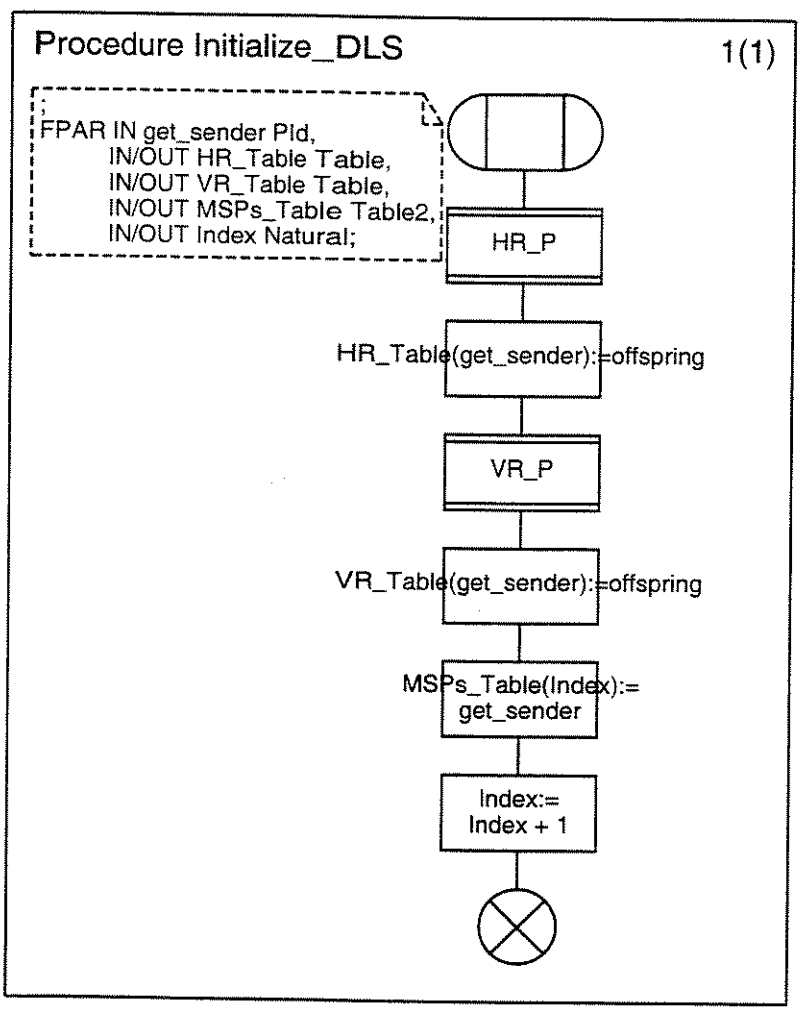

Figura B.11: Procedimento Initialize_DLS/Processo DomainLocationService/Bloco DomainLocationService

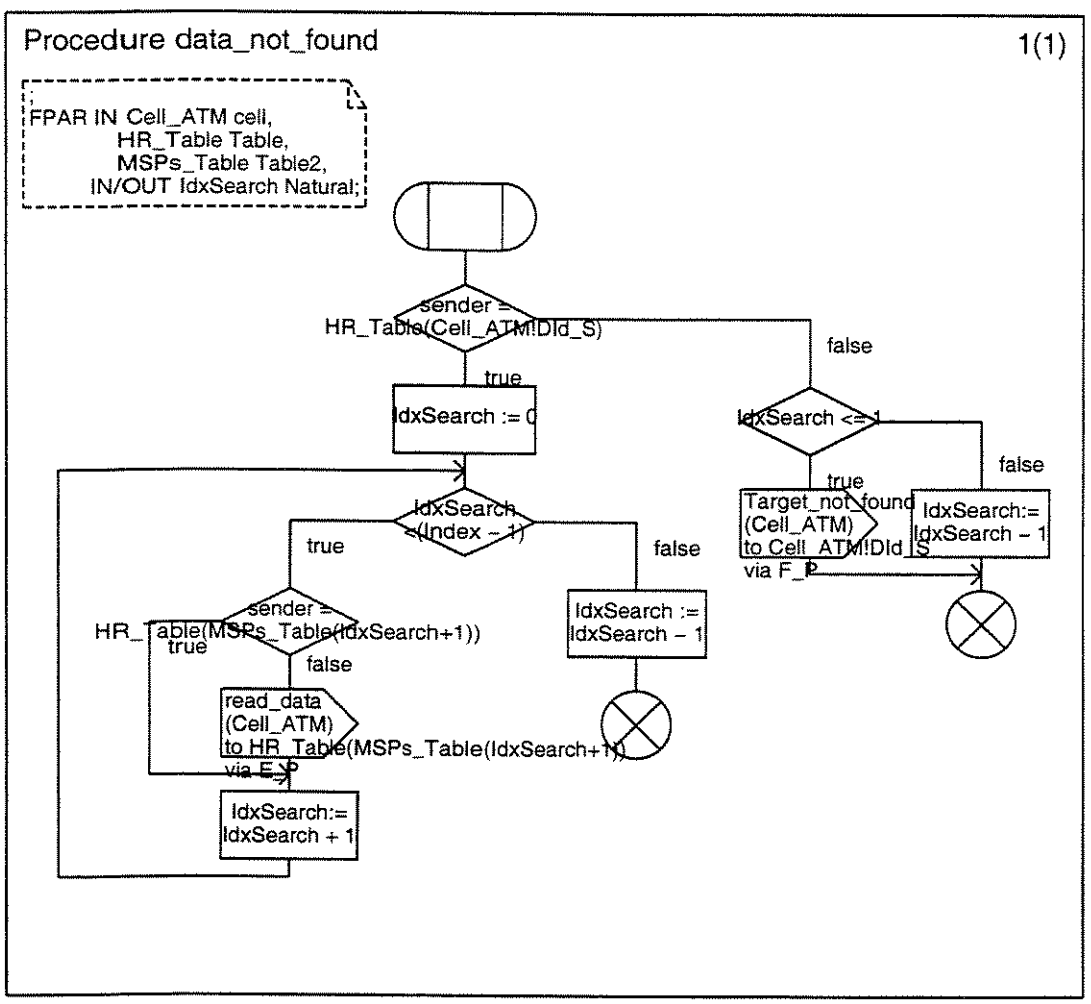

Figura B.12: Procedimento data_not_found/Processo DomainLocationService/Bloco DomainLocationService

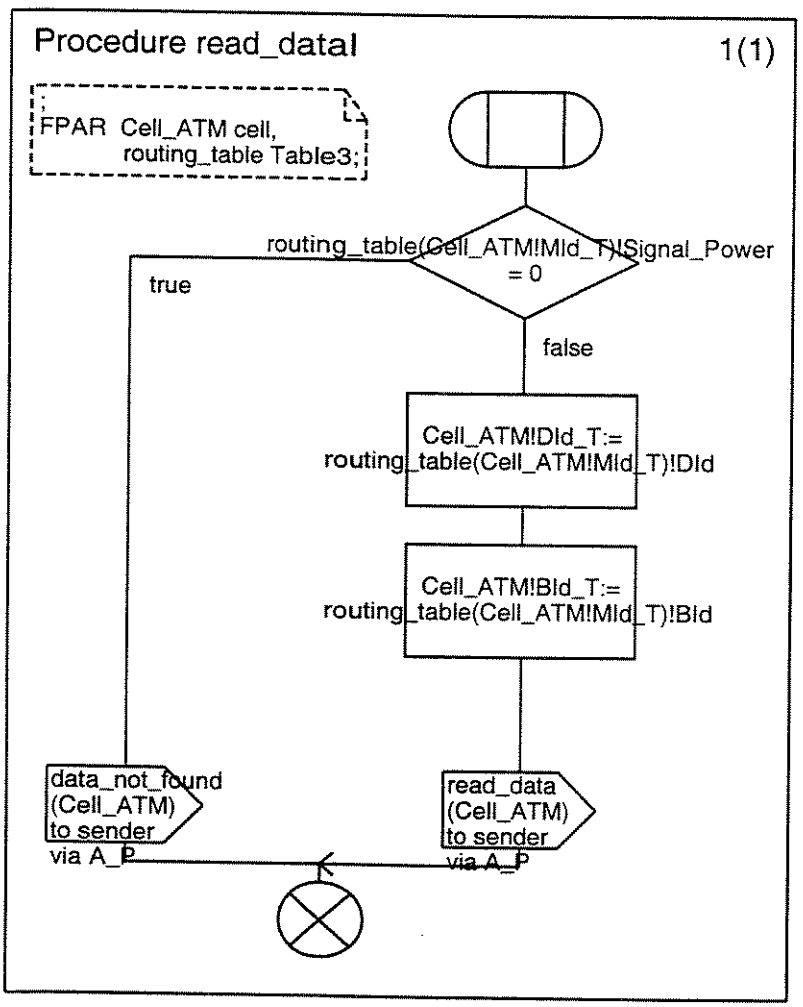

Figura B.13: Procedimento read_dataI/Processo Home_Register/Bloco DomainLocation-Service

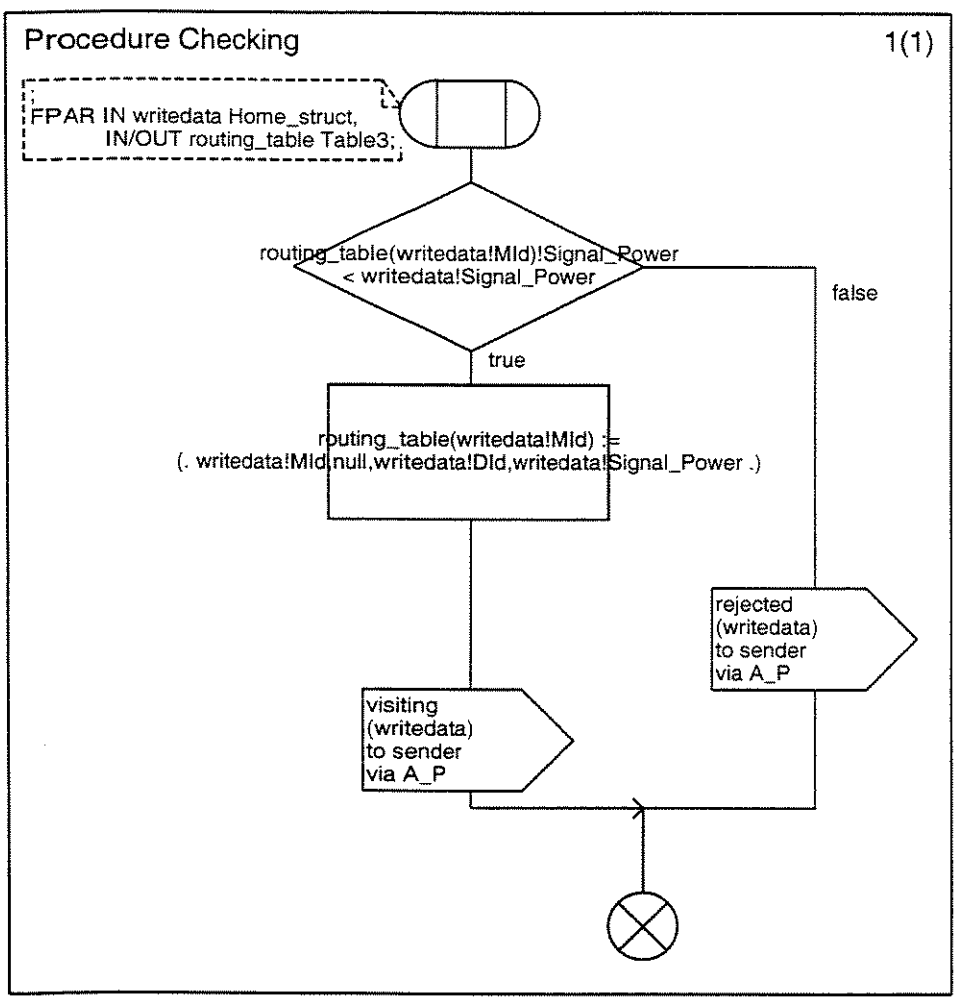

Figura B.14: Procedimento Checking/Processo Home_Register/Bloco DomainLocation-Service

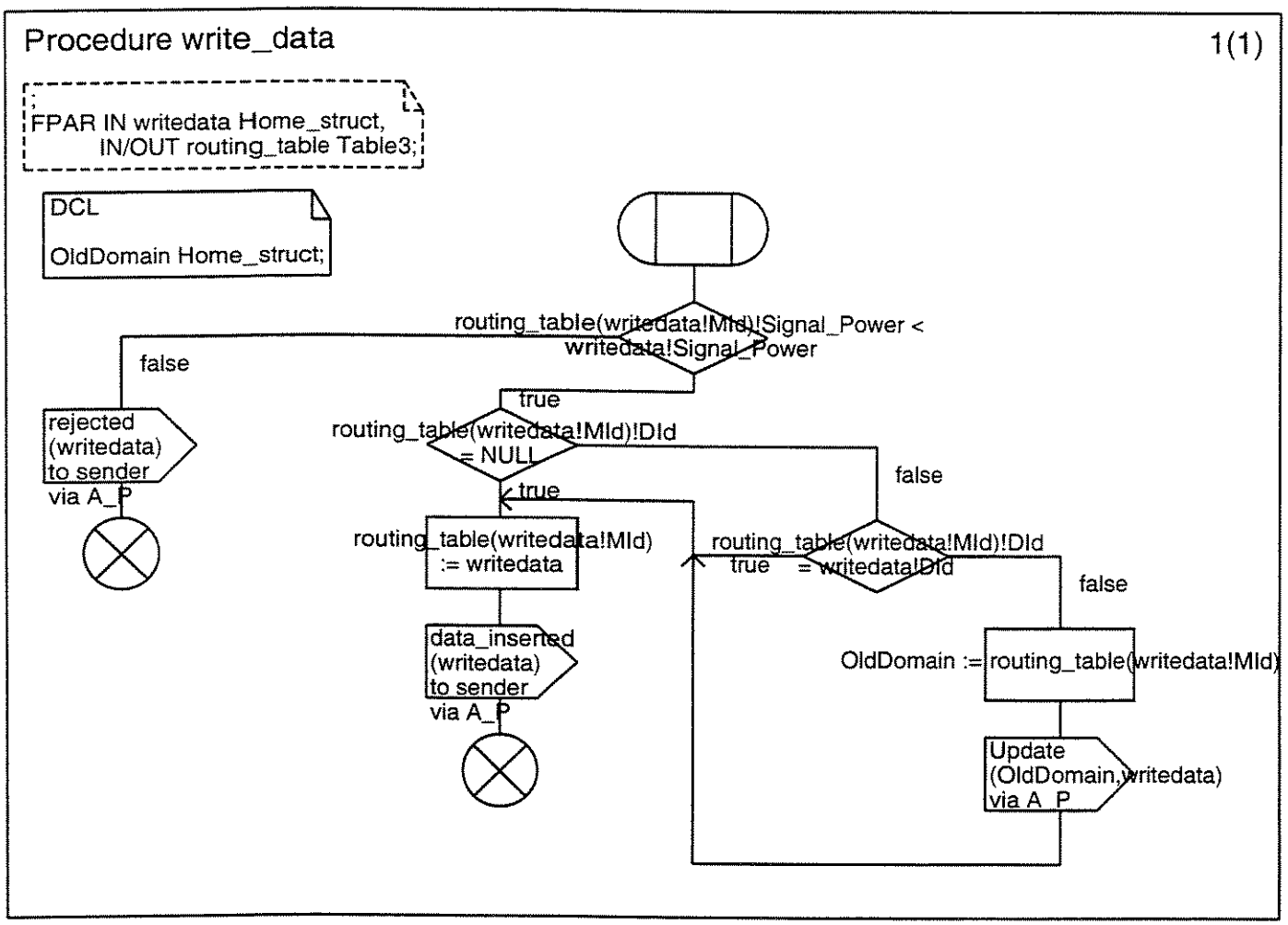

Figura B.15: Procedimento write_data/Processo Home_Register/Bloco DomainLocation-Service

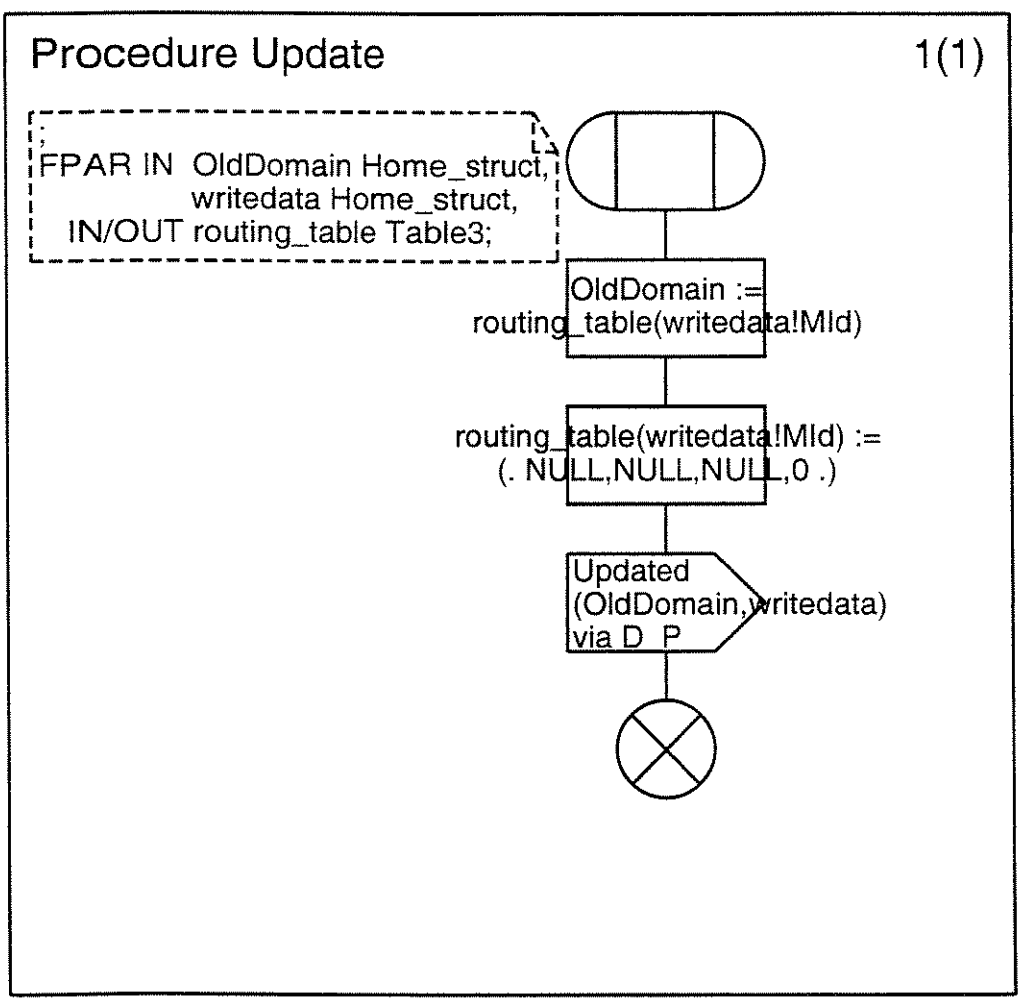

Figura B.16: Procedimento Update/Processo Visiting_Register/Bloco DomainLocation-Service

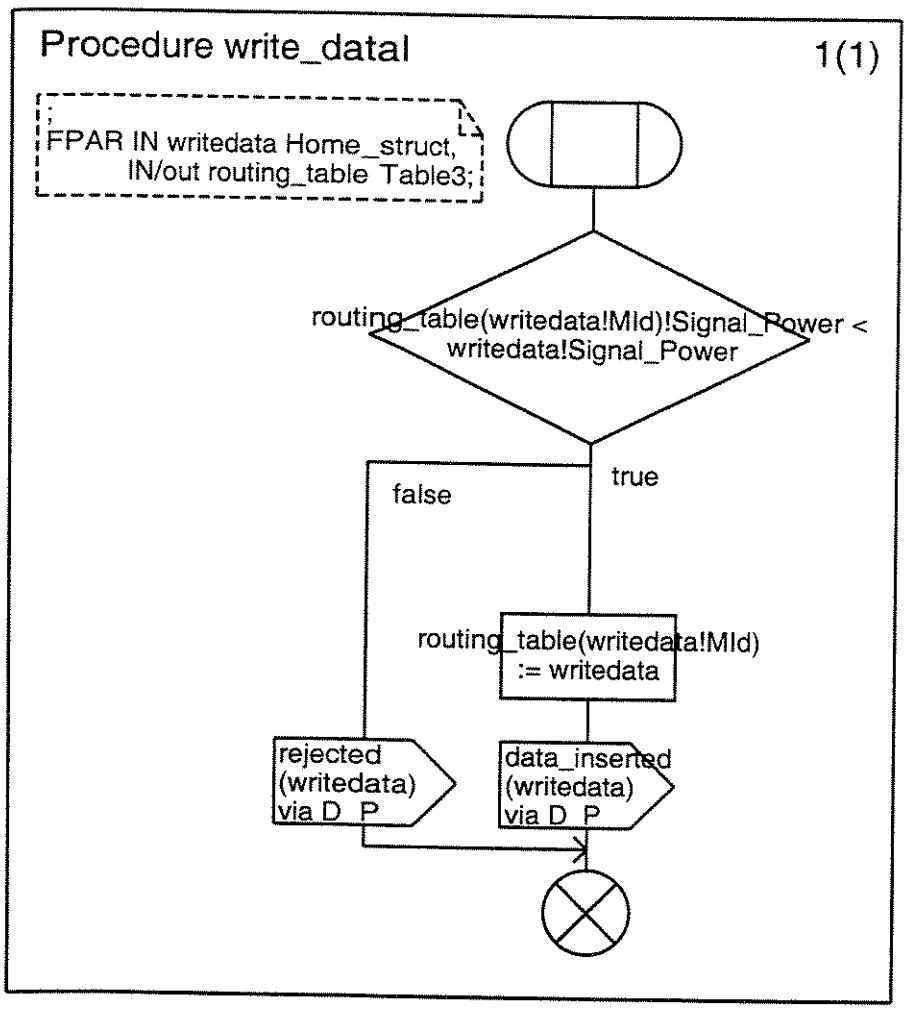

Figura B.17: Procedimento write_dataI/Processo Visiting_Register/Bloco DomainLocationService

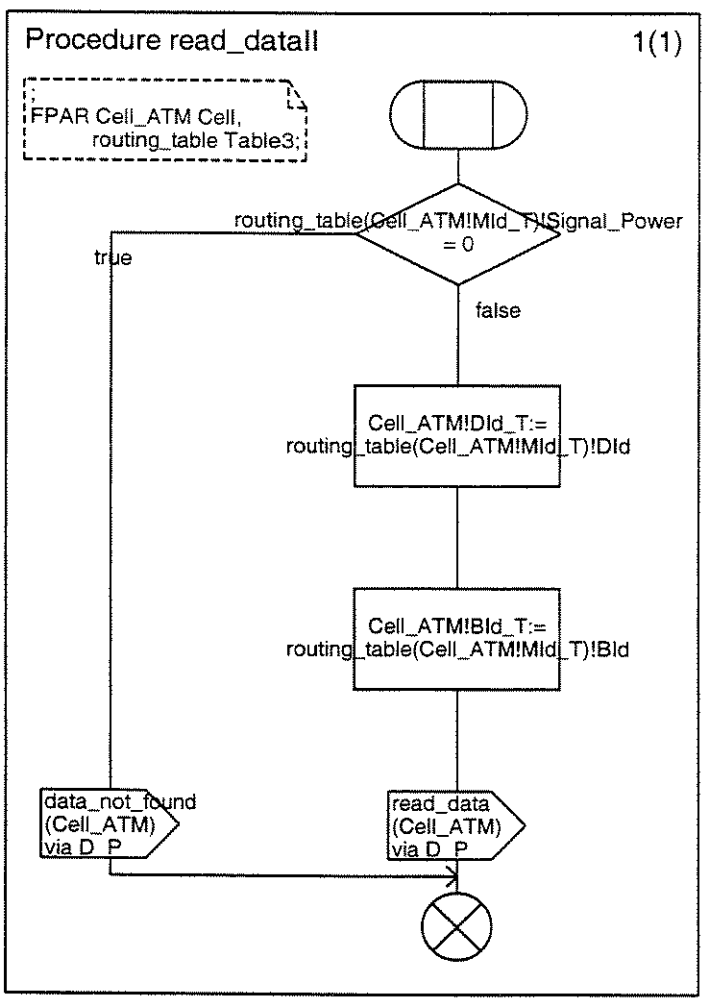

Figura B.18: Procedimento read_dataII/Processo Visiting_Register/Bloco DomainLocationService

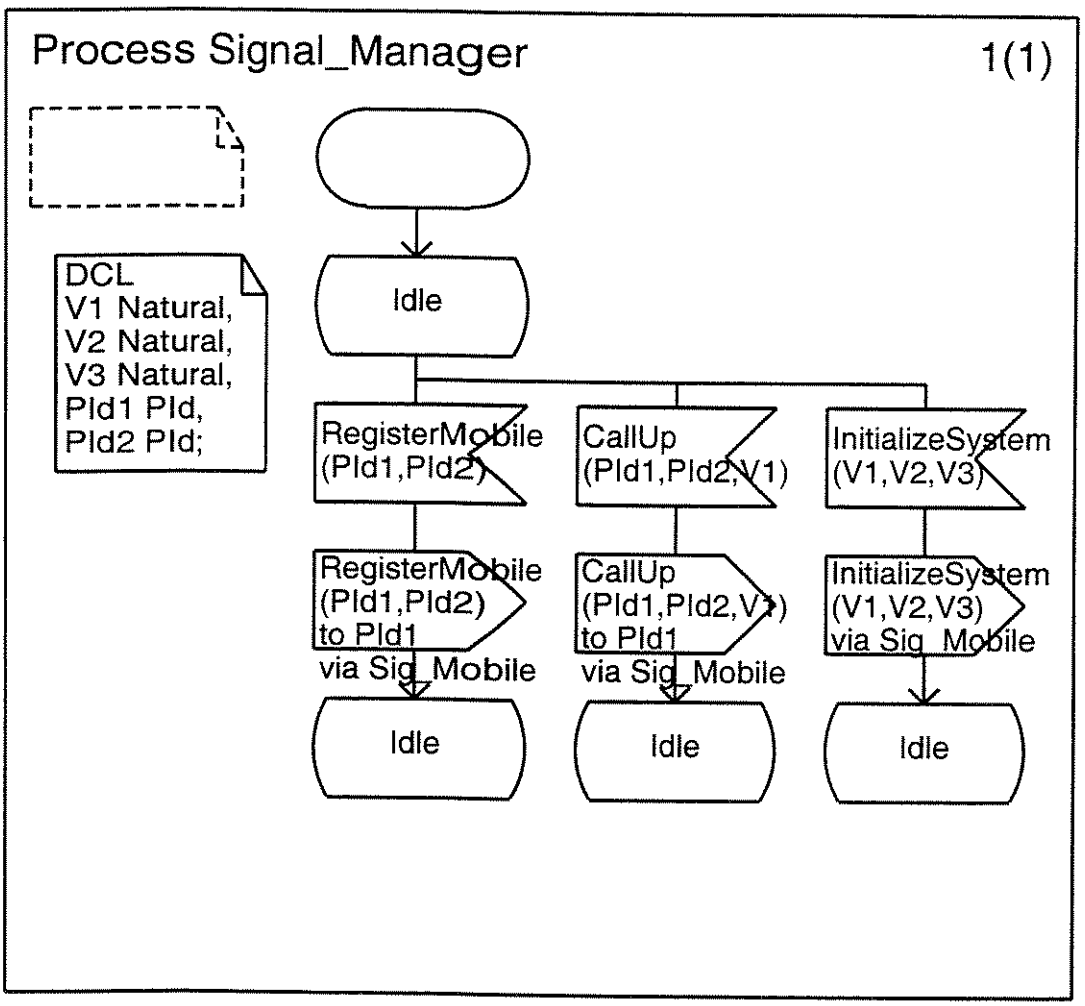

Figura B.19: Processo Signal_Mobile/Bloco Mobile

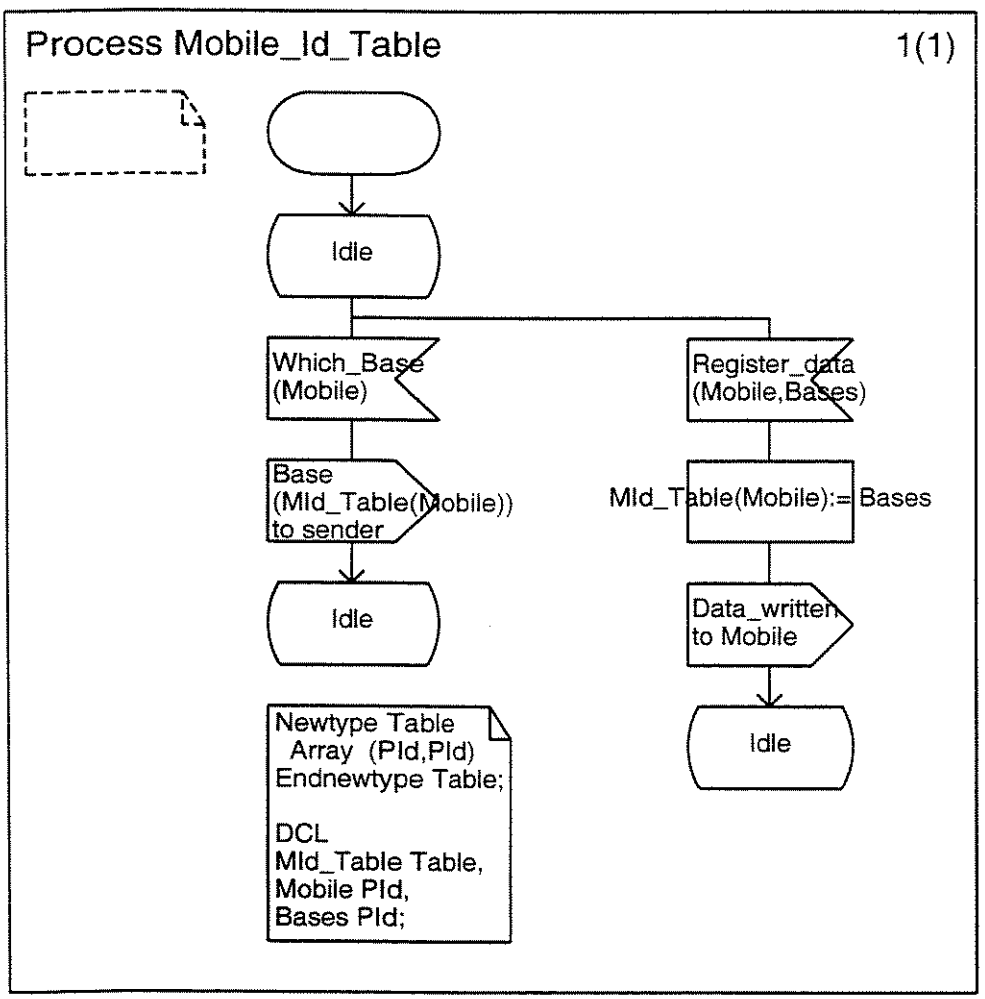

Figura B.20: Processo Mobile_Id_Table/Bloco Mobile

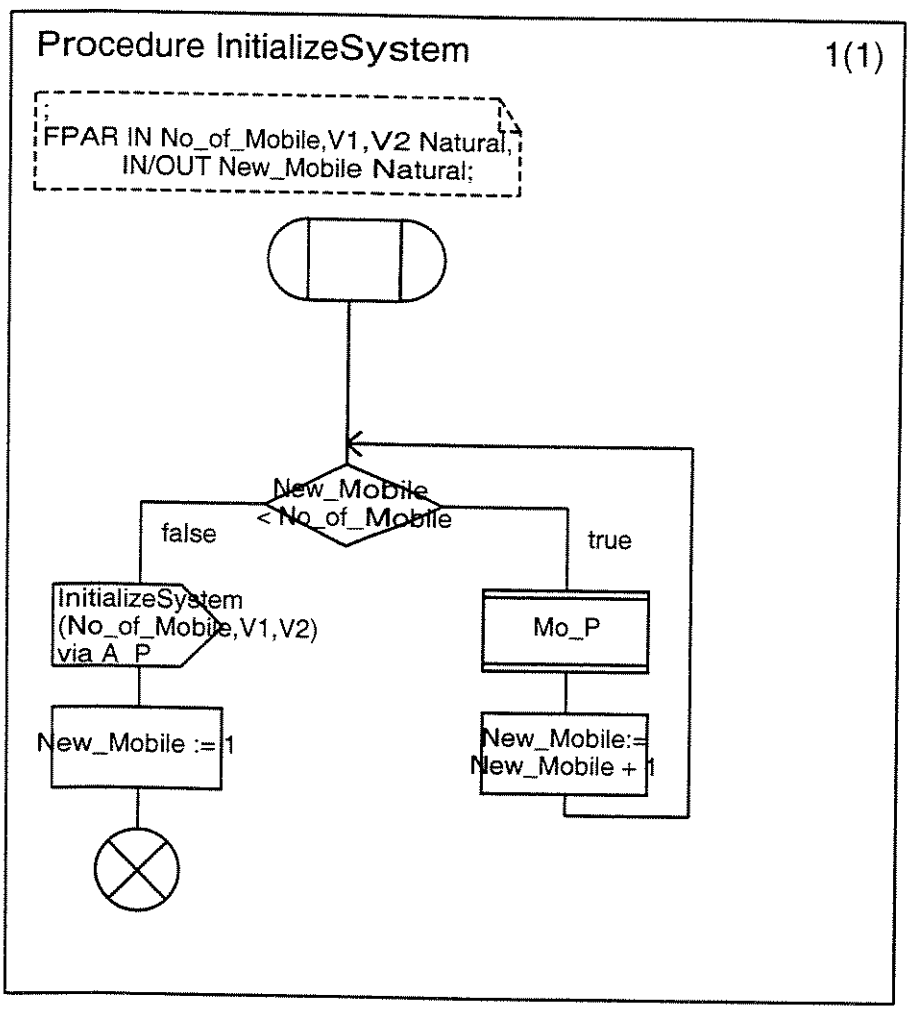

Figura B.21: Procedimento InitializeSystem/Processo Mobile/Bloco Mobile

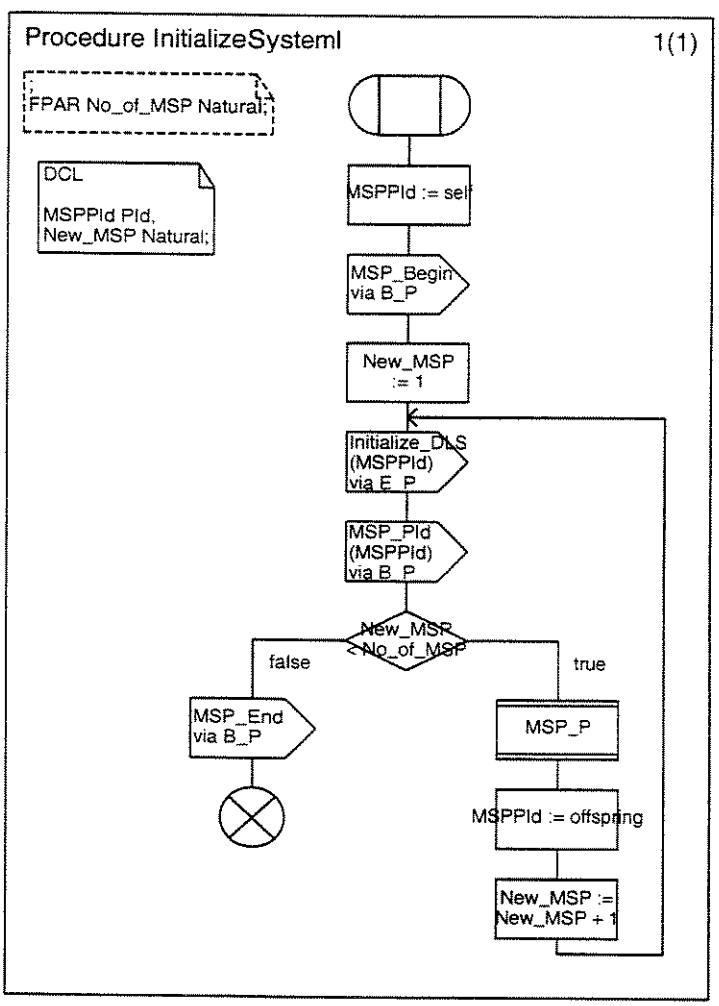

Figura B.22: Procedimento InitializeSystemI/Processo MobileSwitchingPoint/Bloco MobileSwitchingPoint

Bibliografia

[1] B. A. Akyol and D. C. Cox. Signaling alternatives in a wireless atm network. *IEEE Journal on Selected Areas in Communications*, 15(1):35–49, Jan 1997.

[2] Anthony Alles. ATM Internetworking. Technical report, Sisco Systems, Inc., 1995.

[3] M. C Barreto and W. C. Borelli. Especificação em SDL'92 das Sinalizações de *Registration* e *Call Setup* de uma Rede Sem Fio. *CLEI (Conferencia Latinoamericana de Informatica)*, pages 337–346, Set 1997. Valparaíso/Chile.

[4] Frenc Belina, Dieter Sarma, and Amardeo Sarma. *SDL with Applications from Protocol Specification.* Prentice Hall International, 1991.

[5] ATM Forum. Technical Specifications (Spec Watch) - Feb., 1999. http://www.atmforum.com/atmforum/specs/specwatch.html.

[6] Harri Hansén. Connection management functions of a private wireless ATM network. Master's thesis, Helsink University of Tecnology, March 1996.

[7] Douglas Hill, Glen Gabriel Ben-Yosef, and Tom Nolle. *The Switching Book.* Xylan publisher, 1996.

[8] ITU-T, Geneve. *CCITT Specification and Description Language (SDL): programming languages*, 1993. Recommendation Z.100.

[9] ITU-T, Geneve. *Messagem Sequence Chart (MSC)*, 1993. Recommendation Z.120.

[10] ITU-T. *B-ISDN Application Protocols for Access Signalling*, Fev/95. Recommendation Q.2931.

[11] R. Jain. Wireless ATM.
http://www.cis.ohio-state/ jain/cis788/wireless_atm/index.html.

[12] I. Katzela and M. Veeraraghavan. Virtual Tree Routing Protocol for a Wireless ATM LAN. Technical report, Bell Labs, Feb 1996.

[13] William C. Y. Lee. *Mobile Communications Engineering.* McGraw-Hill, second edition, 1997.

[14] N. H. Loukas, N. I. Passas, and L. F. Merakos. Design of Call Control Signaling in Wireless ATM Networks. *IEEE/ICC (International Conference on Communications)*, 1997. Montreal/Canada.

[15] Carla G. N. Macário. O uso combinado da técnica de modelagem baseada em objetos OMT com a linguagem de especificação formal SDL como metodologia alternativa para o desenvolvimento do ambiente de software AIDA. Master's thesis, Universidade Estadual de Campinas, Dezembro 1997.

[16] S. Mauw and M. A. Reniers. An algebraic semantics of Basic Message Sequence Charts. *The computer journal*, 37(4):269–277, 1994.

[17] K. Pahlavan and A. H. Levesque. Wireless Data Communications. In *Proceedings of IEEE*, pages 1398–1430, 1994. Vol. 82, No. 9.

[18] Kaveh Pahlavan, Ali Zahedi, and Prashant Keishamurthy. Wideband Local Access: Wireless LAN and Wireless ATM. *IEEE Communications Magazine*, 15(1):34–40, Nov 1997.

[19] J. Porter .and A. Hopper. An ATM based protocol for Wireless LANs. Technical report, Olivetti Research Limited, 1994. ftp://ftp.orl.co.uk/pub/docs/ORL/tr.94.2.ps.Z.

[20] J. Porter, A. Hopper, D. Gilmurray, O. Mason, J. Naylon, and A. Jones. The ORL Radio ATM System, Architecture and Implementation. Technical report, Olivetti Research Limited, Jan 1996. ftp://ftp.orl.co.uk/pub/docs/ORL/tr.96.5.ps.Z.

[21] Martin de Prycker. *Asynchronous Transfer Mode Solution for Broadband ISDN.* Prentice Hall International, third edition, 1995.

[22] D. Raychaudhuri, L. J. French, R. J. Siracusa, S. K. Biswas, R. Yuan, P. Narasimhan, and C. A. Johnson. WATMnet: A Prototype Wireless ATM System for Multimidia Personal Communication. *IEEE Journal on Selected Areas in Communications*, 15(1):83–94, Jan 1997.

[23] D. Raychaudhuri and N. Wilson. Multimedia personal communication network: System design issues. *3rd Workshop Third Generation Wireless Inform. Networks*, pages 259–288, 1992. New Brunswick, NJ/USA.

[24] D. Raychaudhuri and N. D. Wilson. ATM-Based Transport Architecture for Multiservices Wireless Personal Communication Networks. *IEEE Journal on Selected Areas in Communications*, 12(8):1401–1414, Oct 1994.

[25] E. Rudolph, P. Graubmann, and J. Grabowski. Tutorial on Message Sequence Charts. ftp.win.tue.nl - pub/tecreports/sjouke/msctutorial.ps.Z.

[26] Robert Saraco, J. R. W. Smith, and Rick Reed. *Telecommunications Systems Engineering using SDL.* Elsevier Science Publishers, 1989.

[27] L. F. G. Soares, G. Lemos, and S. Colcher. *Redes de Computadores: Das LANs, MANs e WANs às Redes ATM.* Campus, 1995.

[28] William Stallings. *ISDN and Broadband ISDN with Frame Relay and ATM.* Prentice Hall International, third edition, 1995.

[29] Telebrás, Brasil. *Básico de Telefonia Móvel Celular*, 1994. 3a. Edição.

[30] Telelogic AB, Malmö, Sweden. *SDT 2.3 User's Guide*, 1993.

[31] M. Umehira, M. Nakura, H. Sato, and A. Hashimoto. ATM Wireless Access for Mobile Multimidia: Concept and Architecture. *IEEE Personal Communications*, pages 39–48, Oct 1996.

[32] U. Varshney. Supporting Mobility with Wireless ATM. *Network Magazine*, pages 131–133, Jan 1997.

[33] M. Veeraraghavan, M. J. Karol, and Y. K. Eng. Mobility and Connection Management in a Wireless ATM LAN. *IEEE Journal on Selected Areas in Communications*, 15(1):50–69, Jan 1997.

[34] M. D. Yacoub. *Foundations of Mobile Radio Engineering.* CRC Press, 1993.

[35] R. Yuan, S. K. Biswas, and D. Raychaudhuri. A Signaling and Control Architecture for Mobilit Support in Wireless ATM Networks. *IEEE/ICC (International Conference on Communications)*, pages 478–484, 1996. Dallas/USA.