PROTOCOLO PARA GERENCIAMENTO HIERÁRQUICO DE REDES DE TELECOMUNICAÇÕES

Marcial Porto Fernandez

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO EM ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

____

Prof. Aloysio de Castro Pinto Pedroza, Dr.

____

Prof. Michael Anthony Stanton, Ph.D.

____

Prof. Otto Carlos Muniz Bandeira Duarte, Dr.Ing.

RIO DE JANEIRO, RJ - BRASIL

JANEIRO DE 1998

FERNANDEZ, MARCIAL PORTO

Protocolo para Gerenciamento Hierárquico de Redes de Telecomunicações [Rio de Janeiro] 1998

XII, 147 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia Elétrica, 1998)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Protocolo
2. Gerenciamento de Redes
3. Especificação Formal

I.COPPE/UFRJ II.Título (série)

A luz, o sol, o ar livre
envolvem o sonho do engenheiro.
O engenheiro sonha coisas claras:
superfícies, tênis, um copo de água.

O lápis, o esquadro, o papel;
o desenho, o projeto, o número:
o engenheiro pensa o mundo justo,
mundo que nenhum véu encobre.

(Em certas tardes nós subíamos
ao edifício. A cidade diária,
como um jornal que todos liam,
ganhava um pulmão de cimento e vidro.)

A água, o vento, a claridade
de um lado o rio, no alto as nuvens,
situavam na natureza o edifício
crescendo de suas formas simples.

(João Cabral de Melo Neto - *O Engenheiro*)

À minha querida Irenísia

AGRADECIMENTOS

À minha mãe (*in memoriam*), pelo primeiro e definitivo apoio.

À querida Irenísia, pelo amor, pelo apoio diário na realização deste trabalho e pelas revisões cuidadosas de ortografia e linguagem nos artigos e no texto final da tese.

Ao meu pai, minha tia Emília e meu tio Jorge, pelo apoio, incentivo e afeto, que me proporcionaram a tranqüilidade necessária à realização de qualquer trabalho intelectual.

Ao meu orientador, Prof. Aloysio, pelo apoio, pelos conselhos, pela presença e pelo estímulo durante o desenvolvimento deste trabalho.

Aos membros da banca examinadora pela presença e importante contribuição.

Ao amigo Romário, pelo incentivo e solidariedade.

Ao Gardel, aluno da Iniciação Científica, pela dedicação e grande ajuda no desenvolvimento do programa e nos testes com o simulador.

Aos amigos do GTA: Adriana, Andrea, Antônio, Belém, Bill, Daniel, Emerson, Hermes, Jonny, Luis, Marcelo (Baiano), Marcelo Spindola, Marcos Tadeu, Ronaldo e Rubi.

Aos colegas do laboratório, Emerson, Belém e Leandro, pela grande ajuda para fazer o simulador funcionar.

Ao pessoal da secretaria do PEE, Solange, Dayse e Patrícia

À CAPES, pelo financiamento da minha bolsa de estudo.

Além das pessoas aqui mencionadas, devo ter deixado de citar muitas outras que contribuíram mesmo indiretamente na realização deste trabalho. Peço desculpas a estas e agradeço sinceramente a todos.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências (M.Sc.).

PROTOCOLO PARA GERENCIAMENTO HIERÁRQUICO DE REDES DE TELECOMUNICAÇÕES

Marcial Porto Fernandez

Janeiro de 1998

Orientador: Aloysio de Castro Pinto Pedroza

Programa: Engenharia Elétrica

RESUMO

Os sistemas de gerenciamento de redes de computadores existentes apresentam uma topologia plana, isto é, são constituídos por um único gerente e vários agentes. A gerência de redes de telecomunicações, pela sua dimensão e complexidade, exige maiores recursos computacionais e capacidades do sistema de gerenciamento, impossíveis de se realizar com apenas um gerente. O TMN, norma que se propõe a padronizar sistemas oferecendo estes recursos, já prevê o uso de vários gerentes organizados em uma arquitetura hierárquica. Este trabalho apresenta um estudo dos problemas de gerenciamento de grandes redes de telecomunicações e faz uma proposta de protocolo para um Sistema de Gerenciamento Hierárquico (SGH), apropriado para gerenciar redes desse porte. Este protocolo oferece recursos não definidos na norma TMN, como gerência de domínio, gerência de diretrizes e delegação de gerenciamento, que se mostraram ferramentas valiosas para o gerenciamento de grandes redes. O protocolo SGH é especificado através da linguagem de descrição formal Estelle e suas funcionalidades são verificadas utilizando-se um modelo de testes.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for degree of Master of Science (M.Sc.).

PROTOCOL FOR HIERARCHICAL MANAGEMENT OF TELECOMMUNICATION NETWORKS

Marcial Porto Fernandez

January, 1998

Advisor: Aloysio de Castro Pinto Pedroza

Department: Electrical Engineering

ABSTRACT

The computers network management systems nowadays present a flat topology, that is, a system is represented by one manager and several agents. The management of telecommunications networks, because of its size and complexity, demands larger computer resources and management system capacities, impossible with just one manager. The TMN standard, that intends to standardize systems offering those resources, already foresees the use of several managers organized in a hierarchical architecture. This work presents a study of the problems in managing a large telecommunications network and proposes a protocol for a Hierarchical Management System (SGH), adapted for managing large networks. This protocol offers resources which are not defined in the TMN standards, such as domain management, policy management and delegation management, that are valuable tools for managing a large network. The SGH protocol is specified using the formal description language Estelle and its functionalities are verified using a model of tests.

SUMÁRIO

Índice de Figuras

Índice de Tabelas

Lista de Siglas

ACSE	Association Control Service Element
ASN.1	Abstract Syntax Notation 1
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
CMOT	CMIP Over TCP/IP
DME	Distributed Management Environment
ISO	International Organization for Standardization
ITU	International Telecommunication Union
MIB	Management Information Base
NMF	Network Management Forum
OSI	Open Systems Interconnetion
PDU	Protocol Data Unit
ROSE	Remote Operations Service Element
SDU	Service Data Unit
SGH	Sistema de Gerenciamento Hierárquico
SNMP	Simple Network Management Protocol
TMN	Telecommunications Management Network

Capítulo I

Introdução

Os sistemas de gerenciamento de redes de comunicação utilizam uma mesma filosofia: um elemento principal deve reunir informações e realizar comandos de controle do sistema, enquanto os demais elementos adquirem estas informações e executam as ordens. O elemento principal é denominado *gerente* e os elementos que executam suas determinações são chamados de *agentes*.

A arquitetura é simples e atende à maioria dos sistemas existentes, mas começa a apresentar deficiências quando há um aumento da quantidade de elementos a serem gerenciados. A adição de elementos agentes aumenta a necessidade de processamento no gerente, bem como o tráfego de dados nos seus canais de comunicação. Estas são situações que dizem respeito ao gerenciamento de redes de telecomunicações, nas quais a quantidade de dispositivos gerenciados pode chegar a vários milhões.

As primeiras propostas de gerenciamento hierárquico surgiram para atender à demanda de gerenciamento decorrente do crescimento da Internet. Até então, os sistemas gerente-agente utilizados atendiam às necessidades das redes existentes, cujas extensões não justificavam o uso de sistemas mais complexos. O Manager-to-Manager MIB [1], que descreveremos neste trabalho, foi um dos primeiros sistemas que implementavam uma hierarquia entre gerentes.

Seguindo o exemplo do gerenciamento de rede na Internet, a norma TMN (Telecommunications Management Network) [2], publicada em 1992, propôs uma especificação de sistema que permitisse a gerência integrada de redes de telecomunicações, já que até este momento os sistemas existentes eram proprietários.

O ITU (International Telecommunication Union) incluiu o gerenciamento hierárquico no escopo da norma TMN, reconhecendo a necessidade da concepção de um sistema com vários gerentes organizados de forma hierárquica, mas não definiu de como este tipo de gerência seria implementado. Por isso, a maioria das implementações existentes acabaram utilizando protocolos proprietários. Como exemplo, temos os sistemas TMN comerciais ISM da Bull [3] e TeMIP da DEC [4] que utilizam protocolos proprietários para realizar o gerenciamento hierárquico.

O presente trabalho propõe a definição de um protocolo que implementa um sistema de gerenciamento hierárquico aberto, em conformidade com os objetivos da norma TMN[5].

A arquitetura de gerenciamento hierárquico proposta é adequada para o controle de grandes redes de telecomunicações. Ela restringe o gerenciamento a pequenos grupos, mas permite a sincronização das ações de gerenciamento em todo o sistema. A arquitetura TMN é utilizada para concentrar todas as informações de gerenciamento e realizar o controle nos elementos gerenciados.

Apresentamos neste trabalho algumas técnicas que implementam um sistema hierárquico no ambiente Internet como Manager-to-Manager MIB [1], SubManager [6], e Meta MIB [7]. Mostramos também diversos conceitos de sistemas distribuídos para a organização de um sistema com vários gerentes, tais como gerência de domínio, gerência de diretrizes e a delegação de gerenciamento. Estes conceitos permitem uma arquitetura flexível e escalonável, adequada portanto à gerência de grandes redes de telecomunicações.

O Capítulo II traz uma visão dos sistemas de gerenciamento existentes, como o SNMP (Simple Network Management Protocol), o CMIP (Common Management Information Protocol) e o TMN; mostra os problemas para gerenciamento de grandes redes de comunicação e algumas idéias para tentar resolvê-los. No Capítulo III, encontram-se a especificação do Sistema de Gerenciamento Hierárquico (SGH), suas primitivas e o protocolo correspondente. O Capítulo IV apresenta o projeto do sistema SGH, sua metodologia, modelagem, especificação formal, simulação, teste e comentários sobre o uso do ambiente de projeto. Finalmente, o Capítulo V apresenta as conclusões do trabalho e sugere temas para trabalhos futuros.

Capítulo II

Arquitetura de Gerenciamento

II-1 Introdução

O Capítulo II apresenta resumidamente os sistemas para gerenciamento existentes e mostra os problemas para gerenciar uma grande rede de telecomunicações. Apresentaremos também algumas técnicas de gerenciamento hierárquico e conceitos de gerenciamento distribuído utilizados em nosso trabalho.

A seção II-2 apresenta um resumo dos sistemas existentes para gerenciamento de redes de computadores e telecomunicações, como SNMP, CMIP e TMN. Na seção II-3, mostramos os problemas para se gerenciar uma grande rede de telecomunicações e descrevemos como seria a estrutura de um sistema com vários níveis hierárquicos. A seção II-4 apresenta várias técnicas encontradas na literatura que visam à implementação de um sistema de gerenciamento hierárquico no ambiente Internet e mostra os conceitos de gerenciamento distribuído que forneceram subsídios teóricos formais ao nosso trabalho.

II-2 Sistemas de Gerenciamento de Redes de Computadores e Telecomunicações

Os sistemas de gerenciamento surgiram a partir do momento em que vários equipamentos de comunicação, instalados em locais distantes, necessitaram ser supervisionados e receber comandos de configuração. Estes sistemas, entretanto, eram específicos e proprietários para um certo equipamento ou fabricante. Com a heterogeneidade de equipamentos, era necessário implantar um sistema específico para cada grupo de dispositivos ou fabricante. No final dos anos 80, estabeleceu-se um esforço no âmbito da ISO (International Organization for Standardization) para padronizar os protocolos de gerenciamento, com o objetivo de que um único sistema pudesse gerenciar todos os equipamentos de uma rede.

A normatização dos protocolos de gerenciamento para redes de comunicação de dados seguiram duas linhas principais: o gerenciamento Internet (SNMP) e o gerenciamento OSI (CMIP). A fim de estabelecer um padrão para as redes de telecomunicações, o International Telecommunication Union (ITU) utilizou o sistema OSI, fazendo surgir o TMN.

II-2.1 Gerenciamento Internet (SNMP)

A Rede Internet quando atingiu uma dimensão mundial, obrigou a especificação de um sistema para gerenciar dispositivos diversos. Como a normalização OSI (Open Systems Interconnection) ainda estava no início, e em 1988 foi proposto o sistema de gerenciamento SNMP [8], que deveria funcionar temporariamente até estar disponível o protocolo CMIP [9]. O SNMP, entretanto, alcançou uma abrangência muito grande e acabou se tornando um padrão na indústria. Por isso, foi necessário incluir melhorias, fazendo surgir em 1993 o SNMPv2 [10]. Este já apresentava recursos como o sistema de segurança (criptografia e autenticação), a sincronização entre vários gerentes (não permitir que dois gerentes escrevessem em um agente) e a implementação do informe confirmado (os *traps* do SNMPv1 não têm confirmação) [11].

II-2.2 Gerenciamento OSI (CMIP)

A ISO, observando a dificuldade para gerenciar diversos dispositivos como computadores, comutadores, multiplexadores e PABX, iniciou os estudos para especificar um protocolo de gerenciamento padrão. Este deveria prover informações de alarme, medidas de desempenho, estatísticas de tráfego, relatório de eventos, além da configuração dos elementos de rede. Vários fabricantes, dentre eles, AT&T, Digital e British Telecom, contribuíram, a partir de 1979, para definição da norma do CMIP. Esse esforço foi concluído no período de 1993 a 1995, com a publicação da maioria das normas.

A especificação do CMIP utilizou os conceitos de Orientação a Objeto, adaptando-se facilmente aos sistemas mais modernos. Este ambiente dispõe de uma ótima ferramenta para gerenciar dispositivos de vários fornecedores, além de permitir flexibilidade aos projetistas para organizar um ambiente mutável.

Os sistemas de gerenciamento para redes de computadores SNMP e CMIP usam uma arquitetura plana (Figura 1), por isso a interação se restringe à troca de mensagem entre dois dispositivos, gerente e agente. Nesta arquitetura, o elemento chamado gerente centraliza todas as informações, processa, armazena, interpreta e fornece a ordem para execução de ações corretivas. O outro elemento, chamado agente, realiza uma função localizada e que, dependendo do sistema, pode executar a coleta de informações, processamento e armazenamento. As informações coletadas são armazenadas em estruturas de dados chamadas MIB (Management Information Base). Como se vê, apesar dos vários graus de funcionalidade, toda tomada de decisão é realizada no gerente [12].

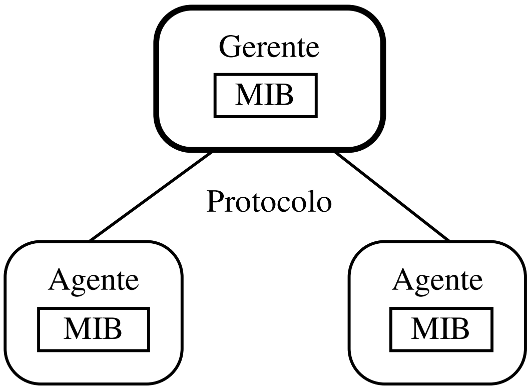

Figura 1 - Arquitetura de um sistema de gerenciamento

Os **Agentes** estão geralmente localizados nos equipamentos remotos, como concentradores de comunicação ou o próprio equipamento usuário, onde podem coletar facilmente as informações de monitoração. Essas informações podem ou não sofrer algum processamento e são armazenadas em uma base de dados local chamada **MIB** - Management Information Base [11].

O **Gerente** é um elemento centralizador, tendendo portanto a ser um elemento único. Ele tem a responsabilidade de solicitar as informações aos agentes, armazenar em uma base de dados própria e apresentar para o usuário responsável pela rede todas as informações de gerenciamento, geralmente de uma forma gráfica. O gerente também dispõe de uma tabela MIB, mas neste caso ela apenas consolida as informações coletadas nos diversos agentes.

O elemento de ligação entre o gerente e os agentes é o **Protocolo**, que deve prover uma comunicação confiável com uma eficiência alta, já que o tráfego de gerenciamento não deve ocupar muito espaço na banda para não diminuir a capacidade útil, isto é, a transmissão de dados para o usuário.

O esquema do protocolo CMIP utilizado entre um gerente e um agente é esquematizado na Figura 2. O gerente pode solicitar ao agente informações relativas a um objeto gerenciado (mensagem solicitação) e o agente responde uma lista de todas as informação desse objeto (mensagem resposta). Para permitir uma diminuição do tráfego entre gerente-agente, o gerente não precisa solicitar constantemente ao agente as informações de situação do objeto gerenciado, o agente pode enviar apenas as alterações ocorridas (mensagem notificação) [13].

Outro tipo de relacionamento, além da troca de mensagens de informações, é a necessidade de determinação e verificação dos atributos de gerenciamento do agente (mensagens escreve atributos e lê atributos, respectivamente). Um exemplo para este tipo de relacionamento são os limites de alarmes para disparar uma mensagem de notificação. Como essa mensagem requer um nível de autoridade maior que a simples consulta, ela será representada separadamente.

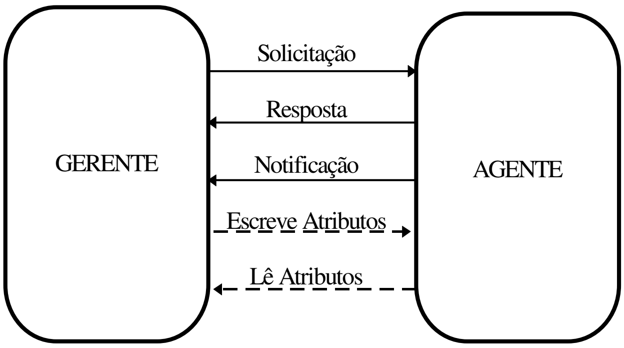

Figura 2 - Mensagens de interação entre Gerente e Agente

II-2.3 Gerenciamento de Telecomunicações ITU (TMN)

Em 1992, foram publicadas pelo ITU (International Telecommunication Union) as normas do TMN, que propuseram um sistema de gerenciamento para redes de comunicação. O TMN foi estruturado sobre os conceitos OSI e de técnicas de programação orientada a objetos. Suas funções de gerenciamento foram organizadas em cinco grupos, conforme as áreas funcionais da OSI [2]:

1. Gerenciamento de Desempenho: o gerenciamento de desempenho trata da monitoração da eficiência dos equipamentos de comunicação, como tráfego e parâmetros de qualidade de serviço (QoS). Suas funções são verificar o estado das tentativas de conexões (sucesso ou falha), monitorar o tráfego de dados e detectar

possíveis congestionamentos, sem atingir os limites definidos no gerenciamento de falhas.

2. Gerenciamento de Falhas (Alarmes): o gerenciamento de falhas trata apenas do estado dos dispositivos de comunicação. Determina os parâmetros e dá informações de alarmes, corrige falhas e executa rotinas de teste.

3. Gerenciamento de Configuração: o gerenciamento de configuração trata de todos os procedimentos necessários para colocar o equipamento em operação, exceto a sua instalação física. Determina a operação ou não de dispositivos, acerta o relógio interno, inicia, atualiza e faz *back-up* dos dados armazenados.

4. Gerenciamento de Contabilidade: o gerenciamento de contabilidade trata de quantificar a utilização dos usuários com o objetivo de cobrança dos serviços. Faz a contabilidade de uso e seu armazenamento e determina as classes de tarifação.

5. Gerenciamento de Segurança: o gerenciamento de segurança trata do controle de acesso de usuários à rede de comunicação. Identifica usuários (para permitir o acesso e efetuar contabilidade), informa acessos, detecta intrusos e tentativas de acesso não autorizado.

O sistema TMN prevê, na sua especificação básica, uma arquitetura hierárquica que possibilita o controle de uma grande rede de telecomunicações.

II-3 Problemas de Gerenciamento de uma Rede de Grandes Dimensões

O principal problema de uma grande rede é não poder ser gerenciada apenas por um gerente. Mesmo que seja admitido o aumento da capacidade de processamento dos novos equipamentos, sempre teremos um limite, ora por causa do processamento, ora por causa da velocidade do canal de comunicação. Assim o sistema deve ser flexível, escalonável, podendo a sua capacidade ser aumentada de acordo com as necessidades.

A solução seria implementar um sistema distribuído, com vários gerentes, e que cada um deles pudesse gerenciar um grupo de agentes. Mas é importante atentar para o fato de que a existência de vários gerentes pode provocar um problema de autoridade. Por isso, cada um deve saber o que pode controlar e a quem deve se reportar, de forma que o sistema fique coerente e não haja conflito de instruções entre gerentes diferentes.

Em um sistema de grande porte não se pode também vincular rigidamente um agente a um gerente, pois no caso de um gerente ficar inoperante, deve-se ter a possibilidade de outro gerente assumir o controle.

Uma arquitetura hierárquica é a mais adequada neste caso, mas ela não deve ser rígida no que concerne às relações entre agentes e gerentes, nem provocar muito retardo na transmissão de informações entre eles. O sistema deve ser flexível também quanto à possibilidade de acrescentar elementos gerenciados dinamicamente, à medida que sejam ativados, assim como retirá-los caso sejam desligados.

Uma informação importante para o gerenciamento é a hora local, para registrar, por exemplo, o momento em que o equipamento sofreu alguma ocorrência. Por isso, é necessário uma sincronização dos relógios no sistema e, dependendo da precisão requerida para ele, a conferência da hora local deve ser mais ou menos freqüente.

Um sistema hierárquico deve também respeitar a responsabilidade e autoridade das empresas que operam os sistemas de telecomunicação. Como é comum haver diversas empresas operando as telecomunicações em uma região, uma para ligações dentro do mesmo estado, outra para ligações dentro do país e outra para ligações internacionais, é necessário respeitar a autoridade sobre a área de atuação. Por exemplo, a empresa nacional não pode controlar um canal dentro do estado, pois este é de responsabilidade da empresa estadual. Porém a empresa nacional pode definir regras para os canais de comunicação da empresa estadual que são utilizados para comunicação nacional, de forma a garantir a qualidade do serviço[14].

II-3.1 Estrutura Geral para um Grande Sistema de Gerenciamento

O modelo de arquitetura hierárquica é apresentado a seguir com três níveis hierárquicos. Não há, entretanto, qualquer restrição ou diferenças significativas em relação a modelos com mais níveis. Consideremos que a rede seja constituída por comutadores, elementos típicos de uma rede de telecomunicação, e linhas de comunicação.

O primeiro nível hierárquico é representado na Figura 3, onde três comutadores estão interligados e gerenciados por um gerente M1, que controla os parâmetros de comunicação do sistema. Para este gerente, toda a rede é vista desta forma e os controles referem-se exclusivamente à comunicação entre S1, S2 e S3.

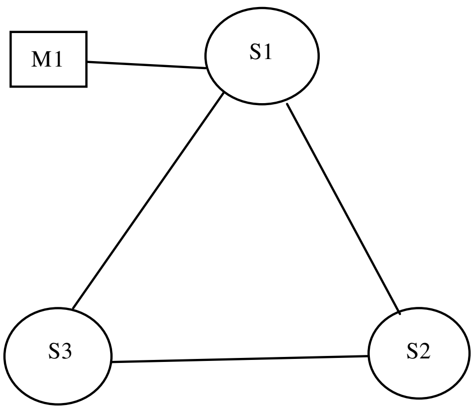

Figura 3 - Primeiro nível hierárquico do modelo de rede

Cada comutador mostrado acima pode ser na realidade uma sub-rede com seus diversos componentes. Um exemplo para a sub-rede S2 é mostrado na Figura 4. O gerente desta camada é MS2 e enxerga apenas essa rede da Figura 4. As conexões externas são vistas como uma conexão em aberto, mas que apresenta características reais, determinadas pelo nível hierárquico superior. Para permitir uma coerência de gerenciamento, são determinados parâmetros pelo gerente hierarquicamente superior, que deverão ser obedecidos pelo gerente MS2.

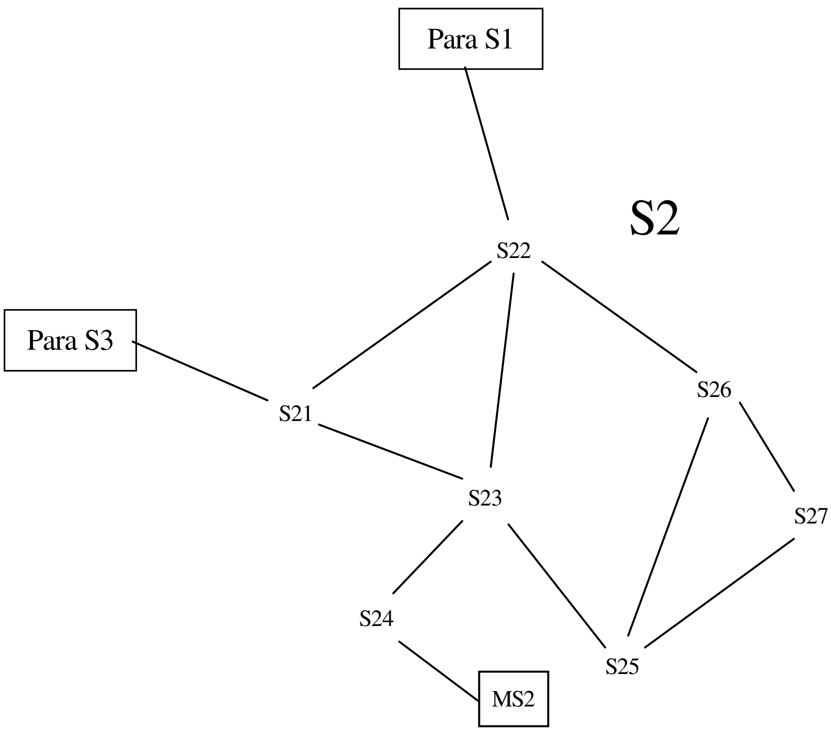

Figura 4 - Segundo nível hierárquico do modelo

Observe-se que a conexão entre S1 e S3 são, na realidade, duas conexões com S21 e S22. O gerente MS2 deve reservar uma banda nessas linhas de comunicação para o gerente M1 e garantir a comunicação entre S1 e S3, enquanto a banda restante pode ser utilizada por qualquer dispositivo pertencente a S2 e gerenciada por MS2. Havendo alguma falha de comunicação dentro deste nível, deverá ser solucionada pelo gerente MS2. Uma falha na rede hierarquicamente superior será resolvida pelo gerente de nível superior, que determinará ações de controle a serem obedecidas por MS2, assim como também por S21 e S22.

A Figura 5 mostra mais um nível abaixo da hierarquia, agora já com a representação de terminais do usuário. O controle é análogo ao anterior. Assim considera-se que, em um determinado nível, cada símbolo de comutador pode ser um equipamento fisicamente instalado ou uma abstração de comutador, na realidade uma sub-rede. O gerente enxerga apenas comutadores, sem se preocupar se é físico ou abstração, e efetua o controle como se ele fosse simplesmente um equipamento.

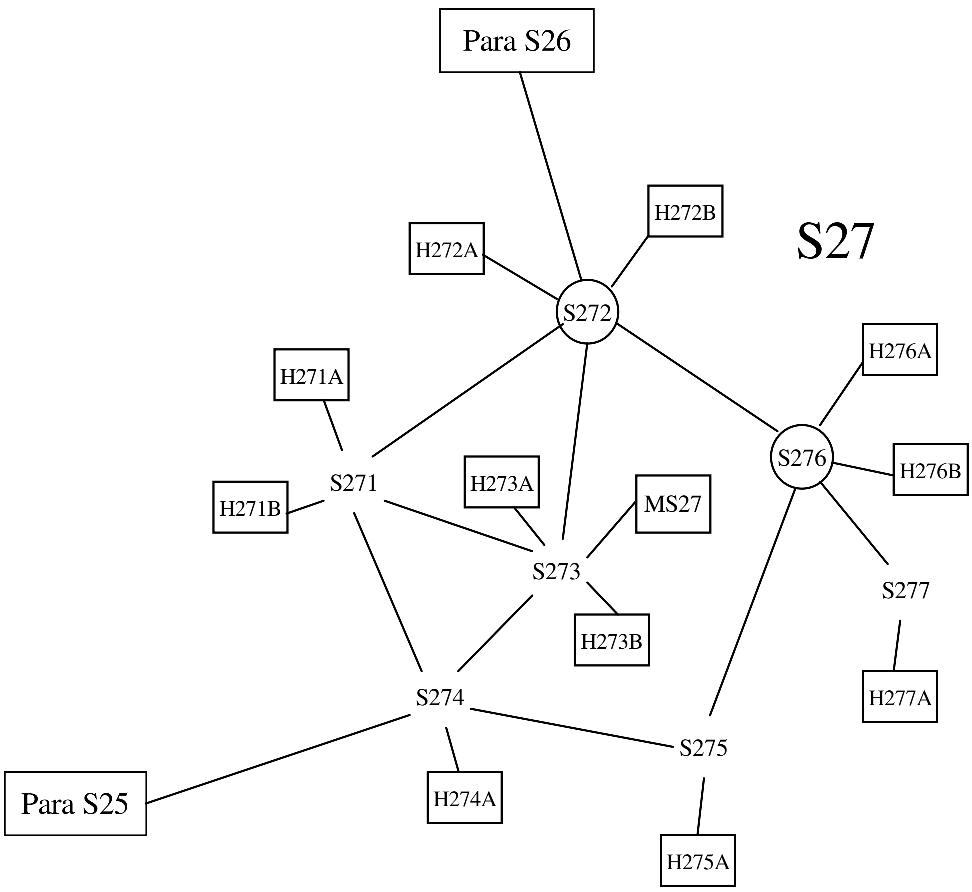

Figura 5 - Terceiro nível hierárquico do modelo

Um gerente de nível intermediário apresenta dois tipos de comportamento, um como gerente, controlando diversos equipamentos, e outro como agente, repassando informações de um comutador virtual para um gerente hierarquicamente superior. Um exemplo de implementação usando este conceito foi mostrado por SPIELMANN e KRAISER em [15].

II-4 Técnicas para Gerenciamento de Sistemas de Grande Porte

Nesta seção, reunimos várias técnicas encontradas na literatura que visam à implementação de um grande sistema de gerenciamento. Na seção II-4.1, apresentamos técnicas de gerenciamento hierárquico no ambiente Internet, comentamos as idéias utilizadas em nosso trabalho e os motivos porque outras foram preteridas.

Na seção II-4.2, apresentamos os vários conceitos de gerenciamento distribuído que forneceram subsídios teóricos formais ao nosso trabalho.

II-4.1 Técnicas de Sistemas de Gerenciamento Hierárquico

Analisamos aqui algumas técnicas de gerenciamento hierárquico no ambiente Internet. Apresentamos uma evolução histórica de três implementações de gerenciamento hierárquico usando o protocolo SNMP. Essas técnicas, entretanto, não são suficientes para a especificação de nosso sistema, pois apesar de implementarem um sistema hierárquico elas pressupõem uma arquitetura rígida, inadequada para o ambiente dinâmico de uma rede de telecomunicações. Em nosso trabalho, aproveitamos o conceito de Meta-MIB, técnica mais recente, da qual conservamos inclusive a denominação em nosso sistema.

II-4.1.1 Manager-to-Manager MIB

A primeira técnica de sistema com mais de um gerente foi implementada no padrão Internet e foi chamada *Manager-to-Manager MIB*, especificada por CASE, McCLOGHRIE, *et al.* [1]. Um gerente secundário concentra informações oriundas dos agentes e replica os dados de gerenciamento de sua MIB, como alarme, eventos e notificação, no gerente principal. Após a coleta de informações dos agentes o gerente secundário envia todas elas em apenas uma mensagem para o gerente principal. Além disso, este elemento principal realiza uma sincronização da base de dados evitando, por exemplo, que dois gerentes escrevam no mesmo agente informações diferentes. A Manager-to-Manager MIB está implementada apenas na versão 2 do SNMP.

O objetivo principal desta técnica é o de permitir que gerentes instalados remotamente, e com linhas de comunicação lenta, sejam gerenciados sem degradar a eficiência do canal de dados. O sistema *Manager-to-Manager* não é apropriado para gerenciar grandes redes pois não reduz o processamento de informações para o gerente principal, apenas diminui a freqüência de envio de dados para o gerente principal e conseqüentemente diminui também a quantidade de dados transmitidos. A função de monitoração do gerente secundário apenas testa valores dentro de certos limites inferiores e superiores. Nenhum cálculo mais complexo necessário para tomar decisão é realizado [6]. Finalmente, esta técnica não tem capacidade para realizar controle hierárquico completo, pois implementa, no máximo, mais um nível intermediário entre o gerente e agente.

II-4.1.2 Gerenciamento Hierárquico - Sub Manager

O *Sub Manager* é uma técnica proposta por SIEGL e TRAUSMUTH em [6]. Ela amplia o conceito da *Manager-to-Manager MIB*, aumentando a funcionalidade e permitindo mais níveis hierárquicos. Sua aplicação, no entanto, é restrita ao ambiente Internet, pois usa o SNMPv1 e SNMPv2.

A idéia básica é criar um dispositivo entre o gerente e agente chamado *Sub-Manager*. Esse dispositivo é constituído por um controle e três tabelas: uma chamada *SubMgrEntry*, que armazena os dados de entrada vindos dos agentes; uma outra chamada *SubMgrOps*, com a lista de procedimentos (como um *script* de funções de gerenciamento); e outra chamada *SubMgrValue*, com os resultados do cálculo de *SubMgrEntry* pelos procedimentos de *SubMgrOps*.

Os dados coletados nos agentes são armazenados em *SubMgrEntry* através dos mecanismos do SNMPv1 ou SNMPv2. Esses dados são processados conforme os procedimentos em *SubMgrOps* e armazenados em *SubMgrValue*. A comunicação entre o *SubManager* e o Gerente principal, assim como o padrão *Manager-to-Manager*, utiliza o protocolo SNMPv2, e os dados que são informados para o gerente principal estão armazenados em *SubMgrValue*.

Os procedimentos armazenados em *SubMgrOps* poderão ser alterados dinamicamente através de uma ferramenta chamada NMP (Network Management

Procedures). O sistema é bastante complexo, incluindo recursos de estampa de tempo e tempo de pesquisa (*polling*), para garantir a sincronização dos dados.

II-4.1.3 Gerenciamento Hierárquico - Meta MIB

Uma técnica para representar a configuração física no ambiente Internet, considerando conceitos de gerenciamento hierárquico e descentralizado, é o uso da Meta-MIB. Essa técnica foi proposta por SCHALLER em [7] e foi utilizada no Projeto WILMA. A idéia é definir três novas MIBs:

- Tabela de Componentes: especificando o tipo de equipamento em cada endereço IP, como computador, roteador, modem, etc.
- Tabela de Conexões: especificando todas as conexões de equipamentos, por exemplo, roteador1 com modem3, estação4 com Hub2.
- Tabela de Agentes: especificando o estado em um nível maior de abstração, tomando como base informações dos agentes. Por exemplo, o estado de uma linha de comunicação é a combinação do estado de dois roteadores e dois modems.

Dessas tabelas a que mais nos interessa é a Tabela de Agentes, pois permite a representação de vários níveis hierárquicos dentro de um sistema de gerenciamento. A implementação de uma Meta-MIB propicia a criação de uma arquitetura hierárquica, mas ainda rígida.

II-4.2 Conceitos de Sistemas de Gerenciamento Distribuído

Um sistema hierárquico rígido não atende às nossas necessidades. Para conferir flexibilidade à arquitetura de nosso sistema, utilizamos conceitos de Sistemas Distribuídos, tais como Gerência de Domínio, Diretriz de Gerenciamento e Delegação de Gerenciamento, os quais descrevemos a seguir.

II-4.2.1 Gerência de Domínios

A Gerência de Domínios é uma proposta de SLOMAN, MAGEE, *et al.* apresentada em [13]. Antes de defini-la, é importante ter em mente os conceitos de Objeto Gerenciado, Objeto Gerenciador e Objeto Composto.

O objeto gerenciado é definido como uma entidade que dispõe de recursos para gerenciamento . De uma forma geral, podemos representar o objeto gerenciado como mostrado abaixo, na Figura 6. O elemento principal é o núcleo de gerenciamento, que é composto pelo *software* de controle e as tabelas de dados (MIBs). Este núcleo interage com o meio exterior através das seguintes interfaces:

- Interface de comunicação com o dispositivo hierárquico superior, que será um gerente ao qual este objeto está relacionado.
- Interface de comunicação com o dispositivo hierárquico inferior, que será um gerente ou agente que este objeto gerencia.
- Interface com o recurso local, específica com o *hardware* e o *software* locais do equipamento, que realiza a coleta de informações de gerenciamento.

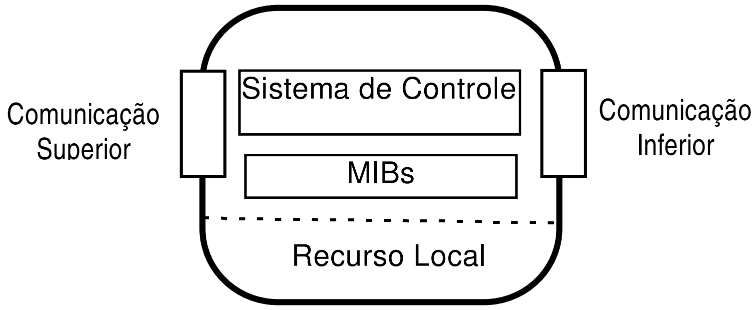

Figura 6 - Modelo de um objeto gerenciado

Em um grande sistema de gerenciamento, é difícil controlar e visualizar todos os elementos conectados à rede. Qualquer gerente, sistema ou operador humano, deve agrupar vários objetos gerenciados em um único objeto, para que ele possa ser tratado.

O objeto composto é definido como um conjunto de objetos gerenciados que possui um ou mais objetos gerenciadores (gerentes). Este conjunto é considerado como um único objeto gerenciado que encapsula todos os objetos internos. Um exemplo de objeto gerenciado composto é mostrado na Figura 7.

No exemplo, temos um objeto composto formado por um objeto gerenciador e dois objetos gerenciados. O objeto gerenciador centraliza a interface com os objetos gerenciados, que por sua vez poderão fazer interface com o nível hierárquico inferior (comunica-se com outro objeto gerenciador). A conexão de objetos gerenciados compostos forma uma árvore hierárquica com relações de responsabilidade e autoridade rígidas entre os objetos.

O objeto gerenciador consolida todos as informações colhidas pelos objetos gerenciados e representa todo esse conjunto de informações como um único objeto. Assim, a situação de todos os sub-objetos coletados pelo objeto gerenciado é um indicativo global (linhas cheias). Se um dispositivo hierarquicamente superior necessitar de alguma informação específica de um objeto gerenciado, solicita diretamente ao objeto que guarda as tabelas de estado (linha tracejada).

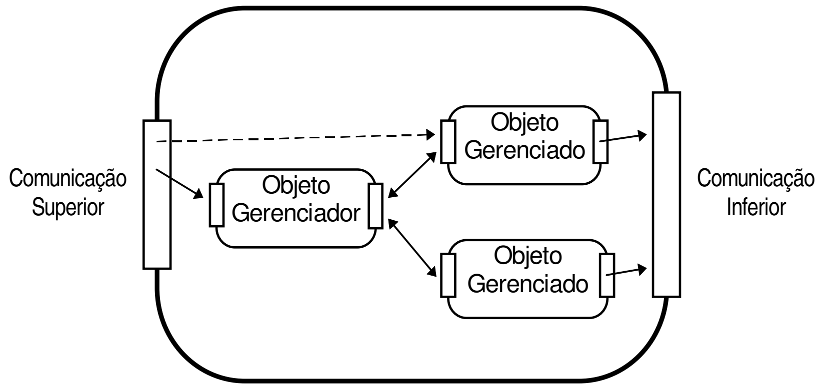

Figura 7 - Modelo de um domínio

O Domínio de Gerenciamento [13] é um grupo de objetos gerenciados com uma única interface. Um domínio é um objeto gerenciado que contém as informações de gerenciamento de todos os objetos pertencentes àquele grupo. Assim como um domínio pode ser por si só um objeto gerenciado, também pode ser membro de outro domínio.

O objeto gerenciado pode manter uma relação direta com um domínio. Neste caso, dizemos que ele é um membro direto e o domínio relativo a ele é chamado domínio pai. Quando existe um domínio ligado a um domínio pai chamamos subdomínio. Membros de um subdomínio são chamados de membros indiretos em relação ao domínio pai ou de filhos do domínio pai.

II-4.2.2 Diretriz de Gerenciamento

A Diretriz de Gerenciamento também é uma proposta de SLOMAN, MAGEE, *et al.* e é apresentada em [13]. O gerenciamento de sistemas distribuídos envolve as funções de monitorar as atividades, tomar decisões sobre gerenciamento e executar ações de controle para modificar o comportamento do sistema. Pela própria filosofia desse tipo de sistema, o gerente deve ter autonomia para decidir as ações que devem ser tomadas, sem precisar receber uma ordem explícita de um gerente superior. Estas decisões, no entanto, dentro do sistema distribuído, com vários objetos executando a função de gerência, precisam de uma diretriz que oriente sua realização.

Uma outra definição, semelhante a mostrada em [13] mas com enfoque mais generalista e abstrato foi apresentada por PUTTER, BISHOP, *et al.* em [16].

Diretriz é uma informação que influencia o comportamento de um objeto, isto é, que diz como deve ser a sua interação com os demais objetos do sistema. Toda diretriz tem um sujeito, no qual o objetivo deve atuar, e um alvo, que é o comportamento desejado. Em alguns casos o alvo ou o sujeito poderão ser implícitos, como, por exemplo, uma diretriz que determina que toda senha deve ter mais de oito caracteres, tem como sujeito implícito qualquer usuário que cria ou troca a senha [17].

Um sistema de gerenciamento, com um número muito grande de objetos, não pode ter uma diretriz para cada objeto individualmente. Assim, é necessário definir vários domínios agrupando diversos objetos com funções semelhantes aos quais uma mesma diretriz se aplica. Um objeto pode ser o sujeito ou o alvo de muitas diretrizes diferentes, o que pode provocar conflito entre diretrizes no objeto, e deve ser tratado por uma outra diretriz específica para tratar de conflito.

A diretriz de gerenciamento especifica também o relacionamento entre objetos gerenciadores e objetos gerenciados. O domínio pode especificar um conjunto de objetos gerenciadores e gerenciados no qual uma diretriz é aplicada.

As diretrizes podem ser divididas em dois grupos: Diretrizes de Autorização, que determinam aos objetos gerenciadores as funções que lhes são ou não permitidas realizar nos objetos gerenciados; e Diretrizes de Obrigação, que definem o que o objeto gerenciador é ou não obrigado a realizar. Uma diretriz de autorização pode ser positiva, permitindo executar uma ação, ou negativa, proibindo uma determinada ação. Já uma diretriz de obrigação determina simplesmente uma ação que deve ser realizada. Quando um objeto recebe uma diretriz de autorização negativa para uma determinada ação e recebe uma diretriz de obrigação para executar esta mesma ação, ocorre naturalmente um conflito de diretrizes que provocará problemas no sistema [18]. O gerenciador humano então julgará se a diretriz é adequada ou não e se deve implementá-la ou não, e poderá resolver eventuais conflitos entre diretrizes.

A especificação de diretrizes por um gerente humano inicia-se no nível mais elevado de abstração e vai sendo refinada a cada nível que desce. Como exemplo, podemos considerar um sistema de controle de uma caldeira. No nível mais elevado de abstração, o sistema deve manter a caldeira em estado seguro. Um nível abaixo, essa diretriz significa que a temperatura interna não deve ultrapassar, por exemplo, 750° C. Ao atingir essa temperatura, uma medida corretiva deve ser tomada, que pode ser fechar uma determinada válvula. Observe que uma diretriz não especifica necessariamente o que fazer para uma determinada situação, mas ela determina para o nível inferior os parâmetros de decisão [19].

A diretriz não é simplesmente um valor que deve ser atendido. Em um sistema heterogêneo, constituído por diversos equipamentos de diversos fabricantes, essas diretrizes poderão tomar valores distintos em função das características particulares de cada objeto gerenciado.

Uma questão preocupante é o conflito de diretrizes em um sistema distribuído. MOFFETT e SLOMAN propõe em [20] um equacionamento de autoridade e obrigações que possibilitam um sistema livre de conflitos mesmo sem a presença humana.

A aplicação de um sistema de gerenciamento baseado em diretrizes utilizado no projeto ESPRIT, assim como a implementação no gerenciamento de uma rede X.400 é mostrado por ALPERS e PLANSKY em [21].

Uma outra proposta de gerenciamento de diretrizes foi feita por WIES em [22], [23] e [24]. A idéia básica é semelhante, apenas complementa a proposta anterior, classificando as diretrizes segundo diversos critérios. Mostramos a seguir alguns deles:

- Tempo de vida: Especifica a duração de uma determinada diretriz, que pode ser curta, média ou longa.

- Modo de disparo: Como a diretriz é ativada, se disparada periodicamente, assincronamente ou se permanece constantemente ativa.

- Atividade: Determina o comportamento diante de um problema que pode ser apenas monitorá-lo, tentar contorná-lo ou executar uma ação corretiva no objeto problemático.
- Modo: Determina o tipo de diretriz, que pode ser de obrigação, permissão ou proibição.
- Geográfico: Se tem alcance apenas em uma sala, andar, prédio, rua, bairro, cidade, país, etc.
- Área funcional: É a função em que atua a diretriz, isto é, desempenho, falha, configuração, contabilidade e segurança.

II-4.2.3 Delegação de Gerenciamento

A delegação de gerenciamento foi conceituada por YEMINI, GOLDSZMIDT, *et al.* em [25] Ela especifica um modelo de sistema de gerenciamento flexível, isto é, um sistema adequado para controlar uma rede que se modifica ao longo do tempo. A idéia é que um gerente, ao detectar que está sobrecarregado para gerenciar um conjunto de objetos, envie um trecho de código para ser executado em outro gerente.

A outra possibilidade de delegação é quando um dispositivo gerente sofre paralisação e um outro dispositivo deve assumir o controle desse grupo de objetos gerenciados. Neste caso, um outro gerente de qualquer nível hierárquico superior assume o controle dos objetos e tem todos os poderes sobre eles, inclusive de delegá-los a outro gerente.

Uma proposta mais complexa foi proposta por MOFFETT e SLOMAN em [26], que, além da autoridade de poder, especifica também a autoridade de delegação. O sistema pode impor restrições semelhantes às organizações humanas, onde um chefe pode delegar a chefia a um subordinado, mas este não pode delegá-la para outra pessoa sem o consentimento do chefe, por exemplo.

A delegação de gerenciamento é importante para o caso de um centro comercial em uma cidade que demanda muitos serviços de telecomunicações durante o dia e poucos durante a noite. Parte dos gerentes responsáveis por gerenciar bairros residenciais próximos poderia auxiliar no controle deste centro comercial durante o período de maior necessidade.

II-5 Uma Proposta para um Sistema de Gerenciamento de Grande Porte

Nesta seção serão mostradas as deficiências das técnicas aqui apresentadas e como os conceitos de sistemas distribuídos poderão auxiliar na especificação de nosso sistema. A maioria das técnicas para gerenciamento hierárquico no ambiente Internet apresentadas são inadequadas para um sistema de gerenciamento de grande porte. Dentre elas, apenas a Meta-MIB mostrou-se viável para a especificação de nosso sistema.

O sistema Manager-to-Manager não é apropriado para gerenciar grandes redes pois não reduz o processamento de informações no gerente principal, apenas diminui a quantidade de dados transmitidos entre os gerentes. Além disso, não tem capacidade para realizar controle hierárquico completo, pois implementa, no máximo, apenas mais um nível intermediário entre o gerente e agente.

O uso do sistema SubManager foi descartado pela mesma razão do sistema Manager-to-Manager, no qual inclusive se baseia. Ele até chega a admitir vários níveis intermediários entre gerente e agente, mas continua a concentrar o processamento no gerente principal.

A única dessas técnicas utilizada parcialmente em nosso trabalho foi a Meta-MIB, que propicia a criação de uma verdadeira arquitetura hierárquica, mesmo que ainda rígida. A tabela da Meta-MIB mais interessante para o nosso sistema é a Tabela de Agentes, que permite a representação de vários níveis hierárquicos dentro de um sistema de gerenciamento. Em nossa proposta, utilizamos uma estrutura chamada Meta-MIB, que implementa a função da Tabela de Agentes, e da qual mantivemos o nome por representar exatamente o conceito apresentado pelos autores.

A Gerência de Domínio permite o controle da posição relativa dos gerentes e agentes na hierarquia, assim como criação e exclusão de elementos do sistema. A forma de implementação desta idéia em nosso projeto será detalhada nos capítulos seguintes.

A Diretriz de Gerenciamento permite organizar os graus de liberdade de comando no sistema, definindo limites de controle para que todos os gerentes possam ter autonomia sem interferir no desempenho global do sistema. As diretrizes também impõe restrições para evitar incoerências no sistema.

A Delegação de Gerenciamento utilizada em nosso projeto foi uma livre adaptação dos conceitos apresentados na seção II.4.2.3 de forma a oferecer recursos para que o próprio sistema efetue o equilíbrio de carga, alocando agentes para outros gerentes. Nossa proposta será discutida no capítulo III.

A proposta de nosso sistema é a combinação da técnica de Meta-MIB com os conceitos de Sistemas Distribuídos, que lhe proporcionam a flexibilidade desejada.

II-6 Comentários

Neste Capítulo, apresentamos um resumo dos sistemas existentes para gerenciamento de redes de computadores e telecomunicações, como SNMP, CMIP e TMN, relatamos os problemas para se gerenciar uma grande rede de telecomunicações e mostramos como seria a estrutura de um sistema com vários níveis hierárquicos. Finalmente apresentamos várias técnicas encontradas na literatura que visam à implementação de um grande sistema hierárquico de gerenciamento e os conceitos de gerenciamento distribuído que constituíram subsídios teóricos para o nosso trabalho.

No Capítulo III, a seguir, apresentaremos a especificação do Sistema de Gerenciamento Hierárquico (SGH), objeto deste trabalho, que utiliza a idéia de Meta-MIB e conceitos de sistemas distribuídos, como gerência de domínio, gerência de diretrizes e delegação de gerenciamento.

Capítulo III

Especificação de um Sistema de Gerenciamento Hierárquico (SGH)

III-1 Introdução

Neste Capítulo, descreveremos um sistema de gerenciamento capaz de controlar e monitorar uma grande quantidade de dispositivos, utilizando uma arquitetura hierárquica. Como base desta arquitetura está a concepção de dispositivos que admitem dois comportamentos distintos: um como gerente (em relação aos diversos agentes sob seu controle) e outro como agente (em relação ao gerente que o controla).

A seção III-2 apresenta o posicionamento do módulo que implementa o SGH com a perspectiva do modelo OSI. A seção III-3 apresenta a estrutura de dados utilizadas pelo Sistema. A seção III-4 mostra a especificação dos serviços oferecidos pelo módulo SGH e a seção III-5 apresenta a especificação de seu protocolo. Finalmente, a seção III-6 apresenta os comentários finais sobre especificação do sistema proposto.

III-2 Estrutura do Sistema de Gerenciamento Hierárquico (SGH)

O Sistema de Gerenciamento Hierárquico (SGH) é um módulo situado na camada Aplicação, conforme o modelo OSI, ocupando o mesmo nível do módulo TMN. O SGH implementa primitivas que possibilitam executar as funcionalidades descritas no capítulo anterior.

O modelo de organização do sistema proposto é apresentado na Figura 8, que mostra esquematicamente a disposição dos diversos módulos. Todos eles estão localizados na camada Aplicação do modelo OSI. As camadas inferiores, de Apresentação até Física, estão representadas genericamente na figura como Meio de Comunicação. Acima delas, está a camada que oferece o serviço CMIS (Common Management Information Service) e que se comunica com outros dispositivos através do protocolo CMIP.

O módulo TMN executa as funções de coleta de informações dos gerentes e as consolida em suas tabelas de dados de gerenciamento (MIBs). Este módulo implementa o comportamento de gerente e atende às necessidades para gerenciamento de uma rede de telecomunicações.

A camada de nível mais elevado foi chamada de Plataforma de Gerenciamento e consiste em um sistema de ambiente gráfico, que oferece as informações de gerenciamento para o operador de redes.

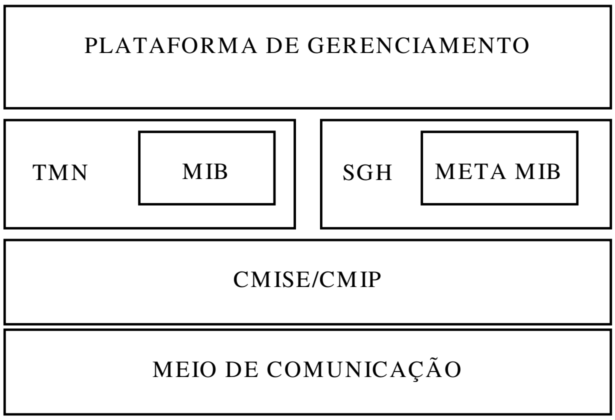

Figura 8 - Arquitetura do Sistema de Gerenciamento

O módulo SGH executa as funções de Gerência de Domínio, Gerência de Diretrizes e Gerência de Delegação, conceitos detalhados na seção II-4. Ele assume o comportamento de agente, fornecendo informações (*reports*) e recebendo informações de gerenciamento do gerente um nível acima, utilizando os dados consolidados na Meta-MIB.

A Meta-MIB apresenta para o gerente acima a imagem da rede gerenciada, segundo a visão do nível superior. O SGH é responsável também por ler as informações das MIBs TMN e convertê-las para a Meta-MIB. Esta conversão é bastante simples. Na Figura 4, por exemplo, se a conexão com a rede superior, representada pela porta 1 do comutador hipotético S2, é realizada pela porta 3 do comutador S21 da Figura 5, essa conversão é apenas a cópia dos parâmetros da porta 3 do comutador S21 para a posição 1 da Meta-MIB, que representa o comutador hipotético S2.

O uso do protocolo CMIP em detrimento do SNMP é justificado pelos seguintes motivos:

- O ITU recomenda esse protocolo para ser utilizado no sistema TMN. Como o SGH se propõe a ser um complemento deste sistema, a escolha do CMIP é natural.

- Um sistema flexível, permitindo vários níveis hierárquicos, é mais fácil de ser implementado com técnicas de programação orientada a objetos (OOD - Object Oriented Design) e o CMIP foi concebido tendo em vista essa técnica.

A forma de implementação dessa arquitetura hierárquica é a criação de um módulo SGH. Este possui primitivas de controle e uma estrutura de dados, uma delas chamada Meta-MIB, que consolida o modelo de rede para o nível superior. Esta Meta-MIB tem uma estrutura idêntica a uma MIB de Agente, o que permite ao gerente assumir o papel de agente em relação ao gerente um nível acima.

Para o módulo TMN, o SGH deve funcionar como um sistema de gerenciamento, solicitando informações das MIBs do TMN. Esta correlação deve ser fornecida pelo SGH. Em relação ao gerente do nível hierarquicamente acima, o SGH deve se comportar como um agente, fornecendo as informações consolidadas.

III-3 Estrutura de Dados do SGH

A estrutura do SGH, seguindo o modelo OSI, é mostrada a seguir na Figura 9. A relação com a camada inferior é realizada através das primitivas do serviço CMIS. Para a camada superior, são oferecidas as primitivas do serviço SGH.

Esta estrutura é composta por uma base de dados com todos os parâmetros necessários, como tabela de gerentes, tabela de agentes, tabela de diretrizes e a Meta-MIB, que serão descritas a seguir. A estrutura de controle realiza a manipulação de dados nas tabelas, oferece o serviço SGH especificado na seção III.4 e implementa o protocolo SGH, utilizando o serviço CMIS, o qual se encontra descrito na seção III.5.

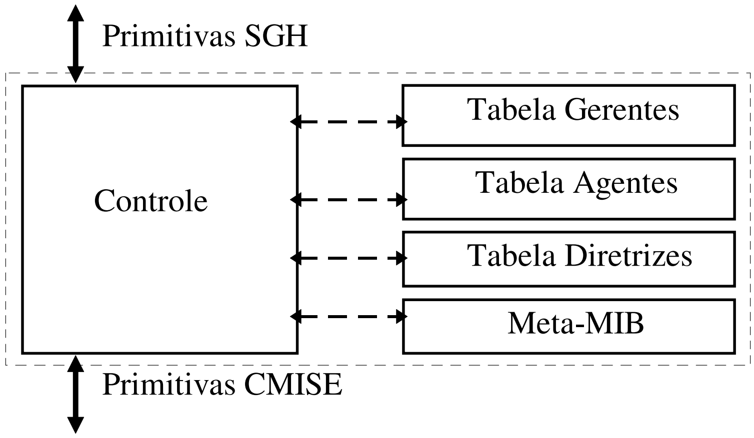

Figura 9 - Estrutura de dados do SGH

A **Tabela de Gerentes** (TableM) é uma lista com o endereço dos dispositivos gerentes de nível hierárquico superior, em ordem prioritária de conexão. Se o primeiro gerente não responder (estiver fora de operação, por exemplo) deve-se tentar contato com o segundo gerente da tabela.

A **Tabela de Agentes** (TableA) determina todos os objetos gerenciados por um determinado gerente. Através dessa lista, poderão ser realizadas as funções de delegação.

A **Tabela de Diretrizes** (TablePolicy) armazena os parâmetros do sistema. Usamos como referência alguns atributos propostos na norma TMN [27]. Nossa proposta de diretrizes é mostrada na Tabela 1.

A **Meta-MIB** deverá armazenar as informações de um dispositivo (comutador) hipotético para o nível hierárquico superior. A definição e tratamento desta MIB não é escopo do presente trabalho.

Tabela 1 - Diretrizes propostas para o SGH

Diretrizes	Descrição
AS	Administrative State. Determina os estados administrativos do objeto, como operativo, implantação, manutenção, etc.
AOL	Affected Object List. Determina a lista de instâncias que podem ser afetadas pela alteração de atributo de um determinado objeto.
ASAL	Alarm Severity Assignment List. Lista de valores de condições de alarme, que determina o seu limiar de disparo.
ASB	Alarm Status Behaviour. Atributos relacionados a um objeto, que especificam um sumário dos alarmes possíveis e sua severidade.
CTPC	Connect Termination Points Counter. Indica o número total de pontos de conexão que poderão ser utilizados.
DB	Directionality Behaviour. Especifica se o objeto gerenciado deve ser unidirecional ou bidirecional.
OS	Operational State. Determina os estados operacionais do objeto, como operacional, operação degradada, fora de operação, etc
PB	Protect Behaviour. Determina os estados de proteção de um determinado objeto, como se faz parte de circuito crítico (red line) ou se é substituível (replaceable).

III-4 Especificação do Serviço Oferecido pelas Primitivas SGH

Apresentamos a relação de primitivas para o sistema SGH, organizadas conforme as funções exercidas. Estas primitivas foram selecionadas a partir de sugestões encontradas na literatura sobre o assunto.

III-4.1. Serviço Oferecido pelas Primitivas de Gerência de Domínio

O serviço de Gerência de Domínio baseia-se nas especificações descritas na seção II-4.2.1. As primitivas de gerência de domínio foram definidas através das sugestões indicadas em [13] e estão relacionadas a seguir:

- Criar um novo domínio.
- Retirar um domínio.
- Inserir objetos em um domínio.
- Retirar objetos de um domínio.
- Relacionar os objetos pertencentes a um domínio.

Essas primitivas permitem a manipulação de gerentes e agentes pertencentes ao conjunto de objetos gerenciados de um determinado gerente. Permitem, por exemplo, a inclusão de um objeto novo ou a retirada de um objeto não mais utilizado.

A Tabela 2 apresenta as primitivas de gerência de domínio do SGH.

Tabela 2 - Descrição do serviço das primitivas de Gerência de Domínio

Nome da Primitiva	Especificação
MInitPrin	Nome: Inicia o gerente principal do sistema para iniciar a hierarquia. Parâmetros da SDU: LocalName, Authorization Função: Inicia o sistema com um gerente na estrutura hierárquica e o faz ser o gerente principal, aquele que vai iniciar todos os outros. Inicia todas as tabelas necessárias para o início de funcionamento do sistema.
ManagerInit	Nome: Inicia um gerente colocando-o na estrutura hierárquica do sistema Parâmetros da SDU: Manager, LocalName, Authorization Função: Inclui um gerente na estrutura hierárquica do sistema. Envia mensagem para o gerente Manager (gerente principal) que o inclui na hierarquia e recebe as tabelas necessárias para seu funcionamento na estrutura.
ManagerRead	Nome: Lê tabela de gerentes Parâmetro da SDU: Authorization Função: Lê toda a tabela de gerentes armazenada em TableM
AgentRead	Nome: Lê tabela de agentes Parâmetro da SDU: Authorization Função: Lê toda a tabela de agentes armazenada em TableA
AgentElimin	Nome: Retira Agente da estrutura Parâmetro da SDU: Agent, Authorization Função: Retira Agent da tabela de agentes (TableA) do gerente local e retira Manager na tabela de gerentes (TableM) do gerente remoto.
AgentCreate	Nome: Cria um Agente na estrutura Parâmetro da SDU: Agent, Authorization Função: Inclui Agent na tabela de agentes (TableA) do gerente local e inclui Manager na tabela de gerentes (TableM) do gerente remoto.

A SDU das primitivas ManagerInit, AgentElimin e AgentCreate é mostrada a seguir:

- Manager: Nome do Gerente a que se relaciona.
- Agent: Nome do Agente a que se relaciona.
- Local Name: Nome do gerente local (dele próprio).
- Authorization: Código de autorização
- Status: Situação de resultado da primitiva.

A SDU das primitivas ManagerRead e AgentRead é mostrada a seguir:

- List: Lista de parâmetros enviados ou recebidos.
- Pointer: Apontador da última posição de List.
- Authorization: Código de autorização
- Status: Situação de resultado da primitiva.

Este conjunto de primitivas oferece todos os recursos para que uma plataforma de gerenciamento possa estruturar um sistema hierárquico.

III-4.2 Serviço Oferecido pelas Primitivas de Gerência de Diretrizes

O serviço de Gerência de Diretrizes baseia-se nas especificações descritas na seção II-4.2.2. As primitivas de gerência de diretrizes foram definidas também através das sugestões indicadas em [13] e são as seguintes:

- Criar objeto de diretriz (atribuir diretriz a um domínio).
- Apagar objeto de diretriz (retirar diretriz de um domínio).
- Ler atributos (diretriz) de um domínio.
- Escrever atributos (diretriz) em um domínio.

Essas primitivas permitem o tratamento e manipulação de um determinado gerente. Por exemplo, permitem ler as diretrizes do gerente ou escrever diretrizes em um agente.

A Tabela 3 apresenta as primitivas de gerência de diretriz do SGH.

Tabela 3 - Descrição do serviço das primitivas de Gerência de Diretrizes

Nome da Primitiva	Especificação
PolicyMRead	Nome: Lê diretrizes locais Parâmetros: Authorization Função: Lê TablePolicy e informa todas as diretrizes estabelecidas pelo gerente superior.
PolicyARead	Nome: Lê Diretrizes do Agente Parâmetros: Agent, Authorization Função: Lê TablePolicy do Agent para verificar alguma incoerência.
PolicyAChange	Nome: Altera Diretriz do Agente Parâmetro: Agent, PolicyList, Authorization Função: Altera diretriz do Agent
PolicyMLoad	Nome: Atualiza Diretrizes com um determinado Gerente Parâmetros: Manager, Authorization Função: Pede a um gerente hierarquicamente superior para atualizar diretrizes locais TablePolicy.

III-4.3 Serviço Oferecido pelas Primitivas de Delegação

O serviço de Gerência de Diretrizes baseia-se nas especificações descritas na seção II-4.2.3. As primitivas de delegação foram definidas tomando como base uma livre interpretação dos conceitos apresentados em [25] e [26] e são as seguintes:

- Solicita autorização para delegar objetos gerenciados.
- Delega objetos para outro gerente.
- Elimina objetos que foram delegados para outro gerente.

Estas primitivas executam a função de delegação de objetos gerenciados de um gerente para outro.

A Tabela 4 apresenta as primitivas de gerência de domínio do SGH.

Tabela 4 - Descrição do serviço das primitivas de Delegação

Nome da Primitiva	Especificação
DelegAct	Nome: Solicita autorização de delegação e executa no caso positivo. Parâmetros: Manager, Agent, Authorization Função: Solicita ao Manager autorização para delegar Agent. Se o teste for afirmativo altera TableA local, TableA do gerente solicitado (Manager) e TableM do agente que vai ser delegado (Agent)

III-5 Especificação do Protocolo SGH

O protocolo SGH utiliza primitivas CMIS. Como ele se encontra em nível de aplicação, os únicos tratamentos de erro que implementa são erros do CMIP e erros de incoerência de dados do SGH.

As primitivas CMIS utilizadas são apresentadas na Tabela 5.

Tabela 5 - Relação de primitivas CMIS

Primitiva	Função
A-Associate	Estabelece a associação (conexão) com outra entidade
A-Release	Encerra a associação (conexão) com outra entidade
M-Get	Lê um bloco de informações de outra entidade
M-Set	Escreve um bloco de informações em outra entidade
M-Action	Executa uma ação em outra entidade

III-5.1 Protocolo de Gerência de Domínio

A entidade MInitPrin realiza apenas funções de leitura e escrita em tabelas locais, não existindo protocolo para ela. Ela permite iniciar o funcionamento do sistema, através da atribuição de valores às tabelas TableM e TablePolicy, além de atribuir o nome ao gerente que está sendo chamado.

A função da entidade ManagerInit é permitir que um gerente seja iniciado com base nas informações de um gerente que já faz parte da hierarquia. Esta entidade é implementada com dois ciclos de mensagem M-Get, uma para TableM e outra para TablePolicy. Após ser estabelecida a associação, o gerente respondedor devolve as tabelas TableM e TablePolicy utilizando a primitiva M-Get. O encerramento é realizado com a primitiva A-Release.

A entidade ManagerRead executa apenas funções de leitura e escrita em tabelas locais, e também não possui protocolo. Lê a tabela TableM, para que o gerente saiba qual a seqüência de gerentes aos quais deve se reportar.

A entidade AgentRead executa apenas funções de leitura e escrita em tabelas locais e não possui protocolo. Lê a tabela TableA, para que o gerente saiba quais os agentes que deve controlar.

A função da entidade AgentCreate é criar um relacionamento entre dois dispositivos na estrutura do sistema, alterando a tabela TableA do gerente iniciador e a tabela TableM do gerente respondedor. Após realizar a associação da comunicação com este outro elemento, o gerente iniciador envia informações para ele através da primitiva M-Set, alterando a tabela TableM. Altera então sua tabela TableA e realiza o encerramento da comunicação com a primitiva A-Release.

A função da entidade AgentElimin é eliminar um agente da estrutura alterando sua tabela TableA do gerente e a tabela TableM do agente. Inicialmente, o gerente iniciador realiza a associação da comunicação com o agente, enviando informações para ele através da primitiva M-Set e alterando sua tabela TableM. O gerente altera então sua própria tabela TableA e realiza o encerramento da comunicação através da primitiva A-Release.

III-5.2 Protocolo de Gerência de Diretrizes

A entidade PolicyMRead executa apenas funções de leitura e escrita em tabelas locais e não existe protocolo para ela. Lê a tabela TablePolicy para o gerente saber os valores de diretrizes armazenadas em sua tabela (que podem ser alterados por um gerente hierarquicamente superior).

A função da entidade PolicyARead é ler as diretrizes de um agente. Após efetuar a associação da comunicação com o agente, busca informações no agente respondedor através da primitiva M-Get, que lê a tabela TablePolicy. O encerramento é realizado com a primitiva A-Release.

A função da entidade PolicyAChange é alterar as diretrizes de um agente. Após ser estabelecida a associação, o agente respondedor altera a tabela TablePolicy através da primitiva M-Set. O encerramento é realizado com a primitiva A-Release.

A função da entidade PolicyMLoad é ler as diretrizes do gerente hierarquicamente superior, com o objetivo de atualizar as diretrizes do gerente iniciador. Inicialmente, esta entidade realiza a associação da comunicação com o gerente, depois busca informações no gerente respondedor, através da primitiva M-Get, que lê a tabela TablePolicy. O encerramento é realizado com a primitiva A-Release.

III-5.3 Protocolo de Delegação

A função da entidade DelegAct é perguntar a um gerente se pode receber a delegação de um agente, isto é, se pode controlar esse agente a partir daquele momento. Inicialmente, a entidade realiza a associação da comunicação com o gerente que receberá a delegação. O envio de informações para o gerente é realizado através da primitiva M-Action. Se a resposta for afirmativa para a delegação, ambos os gerentes (iniciador e respondedor) alteram suas tabelas TableA. Após isso, é realizado o encerramento com a primitiva A-Release.

Acertada a delegação entre os gerentes, é necessário informar ao agente que ele passa a responder para outro gerente. O gerente que fará a delegação estabelece a associação da comunicação com o agente que será delegado e a primitiva M-Action envia informações para o agente. Este altera então sua tabela TableM. Finalmente, o gerente iniciador altera a tabela TableA e realiza o encerramento da associação através da primitiva A-Release.

III-6 Comentários

Neste Capítulo, apresentamos o SGH, um sistema de gerenciamento capaz de controlar e monitorar uma grande quantidade de dispositivos, utilizando uma arquitetura hierárquica. Foram apresentados o posicionamento do Sistema segundo o modelo OSI, a estrutura de dados utilizada e a especificação dos serviços e do protocolo SGH.

O Capítulo IV apresentará o projeto do protocolo do SGH. Mostraremos a descrição do protótipo, a metodologia de projeto, a modelagem em Rede Predicado Ação e a especificação e simulação em linguagem Estelle.

Capítulo IV

Projeto do Sistema de Gerenciamento Hierárquico (SGH)

IV-1 Introdução

Neste Capítulo, apresentaremos o projeto do protocolo do SGH. A seção IV-2 mostra uma descrição do protótipo de simulação e teste, que visa demonstrar todas as funcionalidades do sistema proposto. Na seção IV-3, apresentamos a metodologia utilizada neste projeto. A seção IV-4 descreve a modelagem do protocolo utilizando a técnica de Rede Predicado Ação. A seção IV-5 mostra como foi implementado o protocolo, utilizando-se a linguagem de descrição formal Estelle.

IV-2 Descrição do Protótipo

O sistema de gerenciamento hierárquico proposto neste trabalho será testado com a simulação do caso apresentado na Figura 10. Utilizando este modelo, podem-se realizar os testes relacionados abaixo:

- Ativação de um novo gerente (de qualquer camada), assim como a atualização das suas tabelas.
- Mudança na Tabela de Diretrizes e a verificação de coerência.
- Delegação de gerenciamento, isto é, um gerente passa para outro o controle de alguns objetos gerenciados.
- Simulação de falha em um gerente, provocando que os gerentes um nível abaixo se reorganizem na nova rede, procurando um novo gerente e atualizando parâmetros.

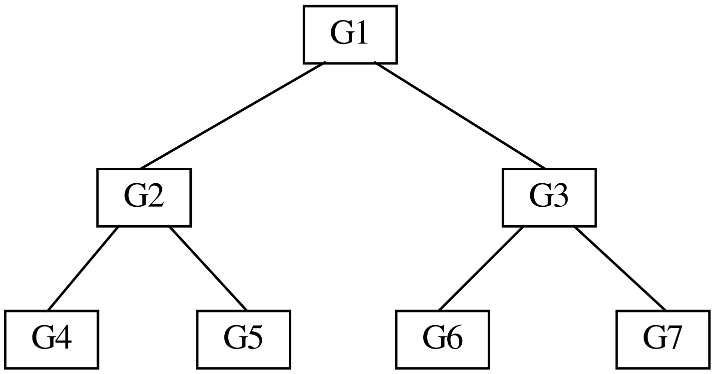

Figura 10 - Proposta de Estudo de Caso

O modelo considerado, mesmo sendo simples, permite destacar todas as interações possíveis entre gerentes, possibilitando uma verificação exaustiva do protocolo. Podemos observar que os gerentes G2 e G3 têm comportamento de gerente e agente.

IV-3 Metodologia de Projeto do SGH

O projeto do Sistema de Gerenciamento Hierárquico (SGH) utilizou a metodologia de especificação em linguagem de descrição formal. Utilizamos a linguagem Estelle em nosso projeto, em virtude de seus recursos e facilidade de uso e também pela disponibilidade de ferramentas de desenvolvimento e verificação.

Na primeira especificação, usamos o modelo de Rede Predicado Ação, pois sua representação gráfica é didática e permite uma análise completa da parte de controle do protocolo. Nesta fase, no entanto, não foi explicitada a manipulação dos dados, que são tratados mais fácil e apropriadamente com a linguagem Estelle. Esta especificação com Rede Predicado Ação permitiu uma verificação inicial e auxiliou na especificação em Estelle. Entretanto, os modelos em Rede Predicado Ação não foram verificados rigorosamente, verificação que foi realizada na simulação em Estelle.

Uma vez especificado o controle do protocolo com o modelo de Rede Predicado Ação, passamos para a manipulação de dados, utilizando a linguagem de descrição formal Estelle [28]. Esta linguagem é mais apropriada para a representação porque apresenta uma estrutura de máquina de estado, o que permitiu aproveitar o modelo em Rede Predicado Ação.

A etapa seguinte é a simulação para verificar o correto funcionamento do protocolo.

IV-4 Modelagem

O protocolo de comunicação do SGH implementa o serviço definido pelo conjunto de primitivas propostas para o SGH, utilizando o serviço CMIS. Apresentaremos abaixo o modelo gráfico de Rede Predicado Ação para cada entidade. Estes modelos, no entanto, não foram verificados com as ferramentas apropriadas, pois serviram apenas como guia para a especificação em Estelle. Esta, por sua vez, foi implementada e verificada. Nas figuras apresentadas não mostramos o estado de comunicação CMIS, que realiza a ligação entre o iniciador e respondedor, para deixar os desenhos mais limpos e claros. As transições de protocolo são todas indicadas.

As entidades MInitPrin, ManagerRead, AgentRead e PolicyMRead executam apenas funções de leitura e escrita em tabelas locais, não existindo comunicação nem protocolo entre as entidades. Por este motivo, o modelo correspondente é trivial e não será apresentado.

A entidade ManagerInit é mostrada na Figura 11. Sua função é iniciar um gerente, utilizando as informações de um gerente que já faça parte da hierarquia. Esta entidade é implementada com dois ciclos de mensagem M-Get, uma para TableM e outra para TablePolicy. Inicialmente, realiza-se a associação da comunicação, utilizando a primitiva A-Associate do ACSE. Depois as tabelas TableM e TablePolicy são recebidas através da primitiva M-Get. O encerramento é realizado com a primitiva A-Release.

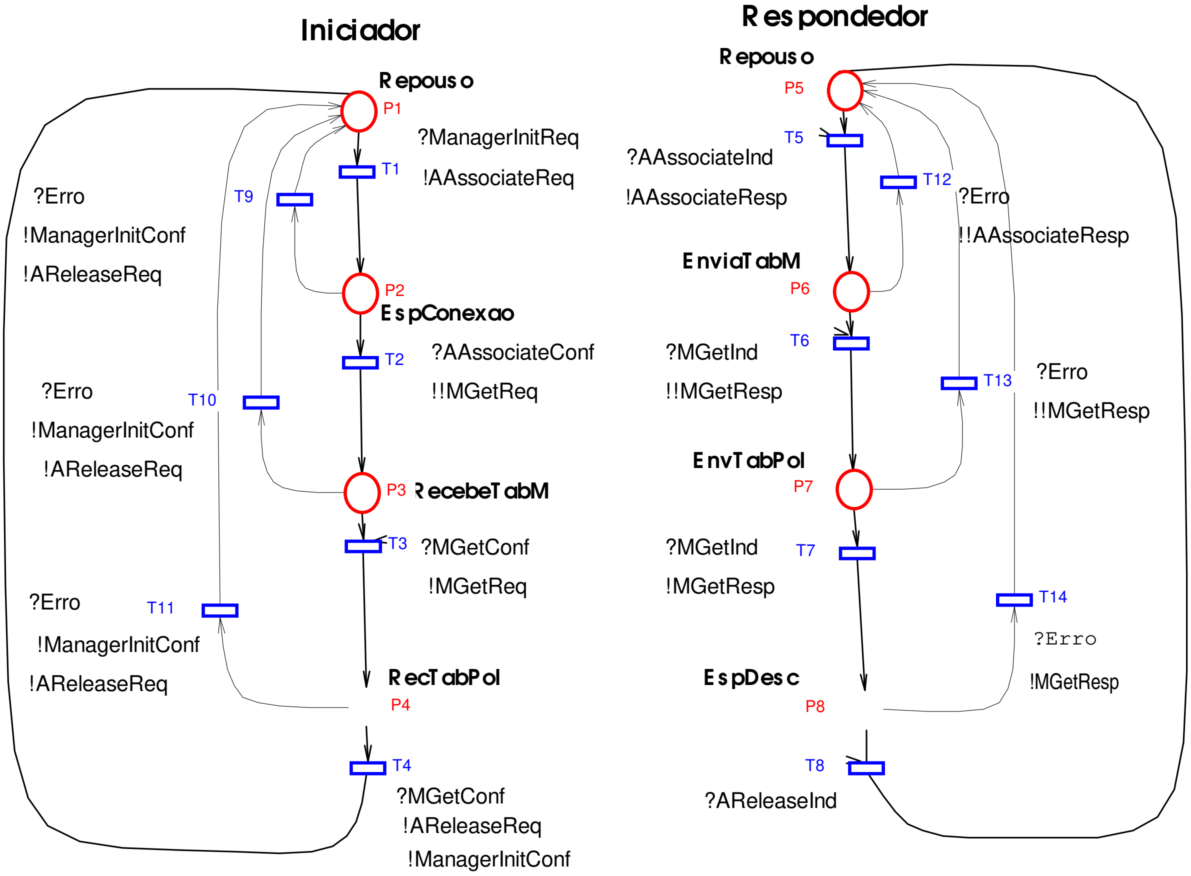

Figura 11 - Modelagem da entidade ManagerInit

A entidade AgentCreate está representada pela rede mostrada na Figura 12. Sua função é criar um relacionamento entre dispositivos na estrutura, alterando a tabela TableA do gerente iniciador e a tabela TableM do gerente respondedor. Inicialmente, o gerente realiza a associação da comunicação com o agente, utilizando a primitiva A-Associate do ACSE. O envio de informações para o agente respondedor é realizado através da primitiva M-Set, que altera a tabela TableM. O gerente iniciador altera então sua tabela TableA e realiza o encerramento através da primitiva A-Release.

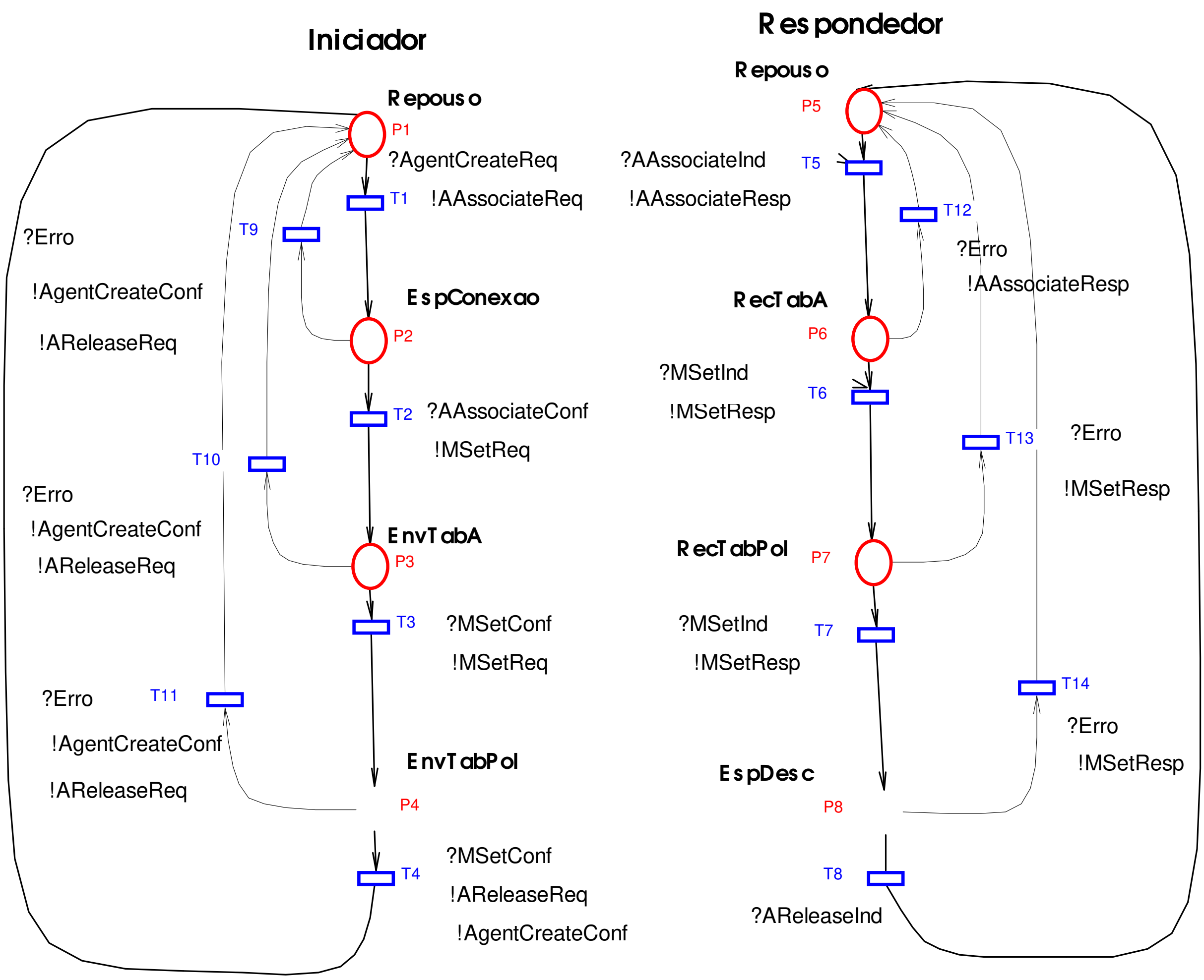

Figura 12 - Modelo da entidade AgentCreate

A entidade AgentElimin está representada pela rede mostrada na Figura 13. Sua função é eliminar um agente da estrutura alterando a tabela TableA do gerente e a tabela TableM do agente. Inicialmente, o gerente realiza a associação da comunicação com o agente, utilizando a primitiva A-Associate do ACSE. O envio de informações para o agente respondedor é realizado através da primitiva M-Set, que altera a tabela TableM. O gerente iniciador altera então sua tabela TableA e realiza o encerramento através da primitiva A-Release.

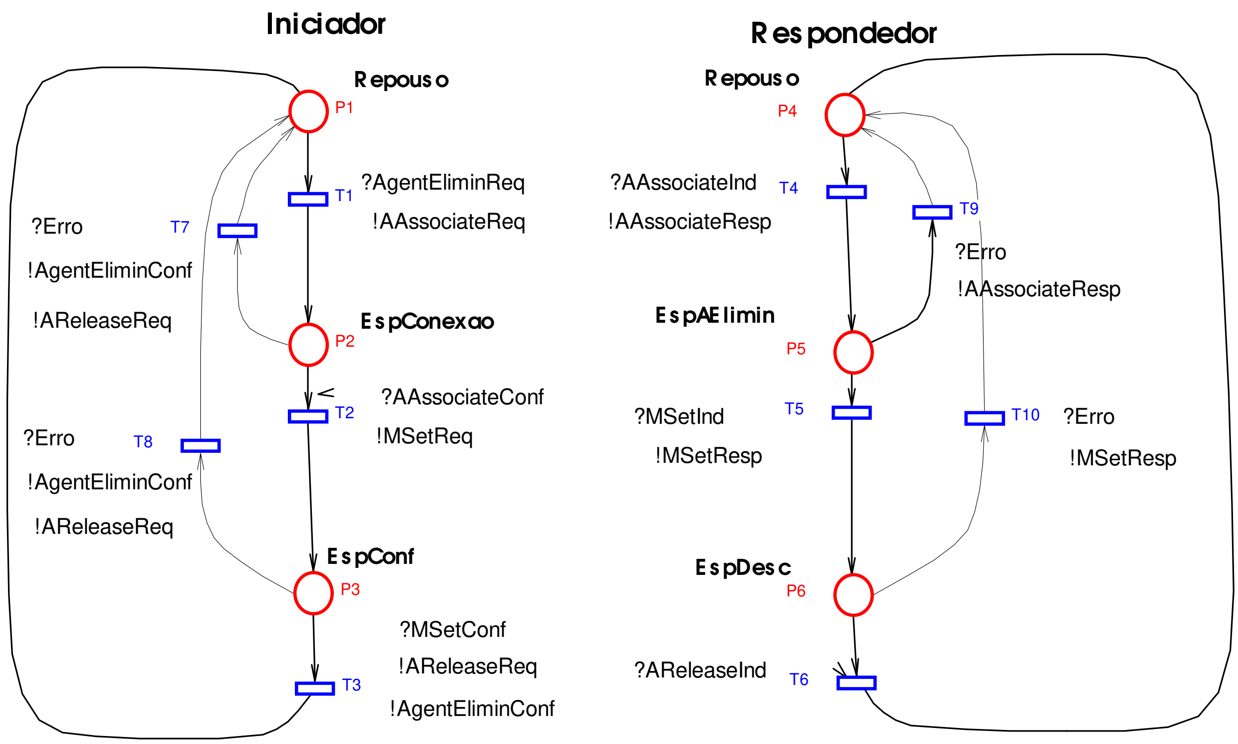

Figura 13 - Modelagem da entidade AgentElimin

A entidade PolicyAChange está representada pela rede mostrada na Figura 14. Sua função é alterar as diretrizes de um agente. Inicialmente, o gerente realiza a associação da comunicação com o agente, utilizando a primitiva A-Associate do ACSE. O envio de informações para o agente respondedor é realizado através da primitiva M-Set, que altera a tabela TablePolicy. O encerramento é realizado com a primitiva A-Release.

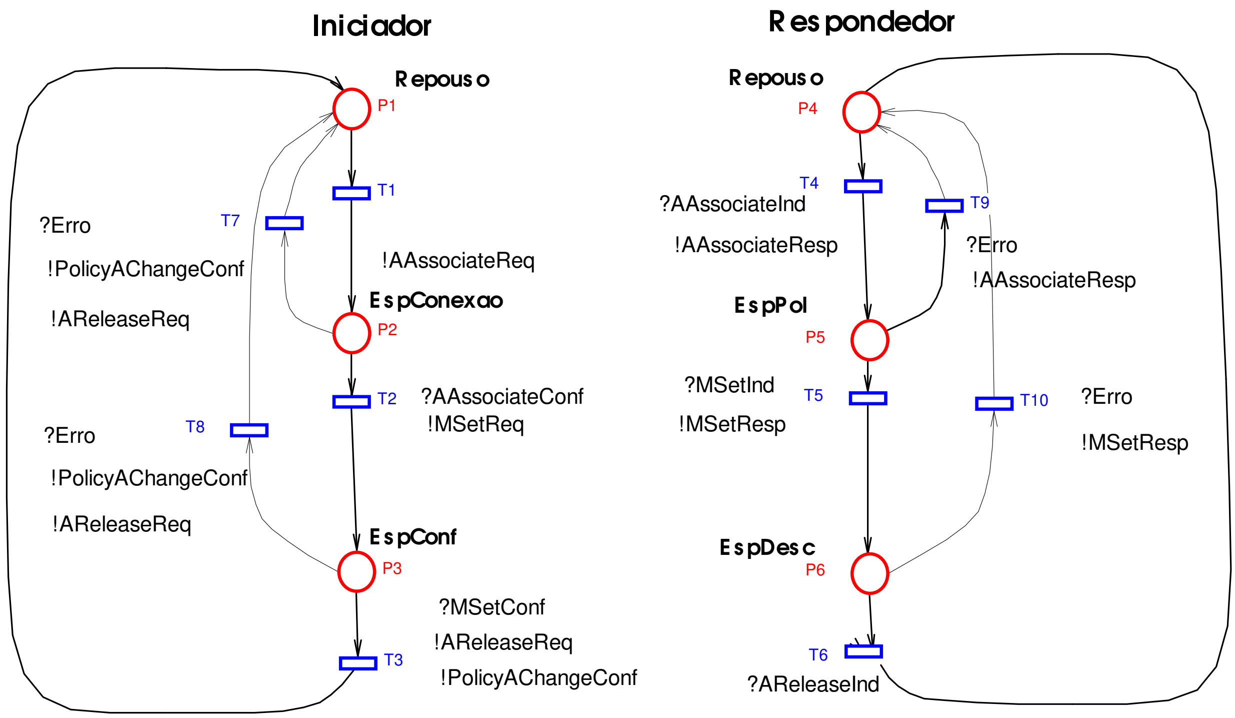

Figura 14 - Modelagem da entidade PolicyAChange

A entidade PolicyARead está representada pela rede mostrada na Figura 15. A sua função é ler as diretrizes de um agente. Inicialmente, o gerente realiza a associação da comunicação com o agente, utilizando a primitiva A-Associate do ACSE. A busca de informações no agente respondedor é realizada através da primitiva M-Get, que lê a tabela TablePolicy. O encerramento é realizado com a primitiva A-Release.

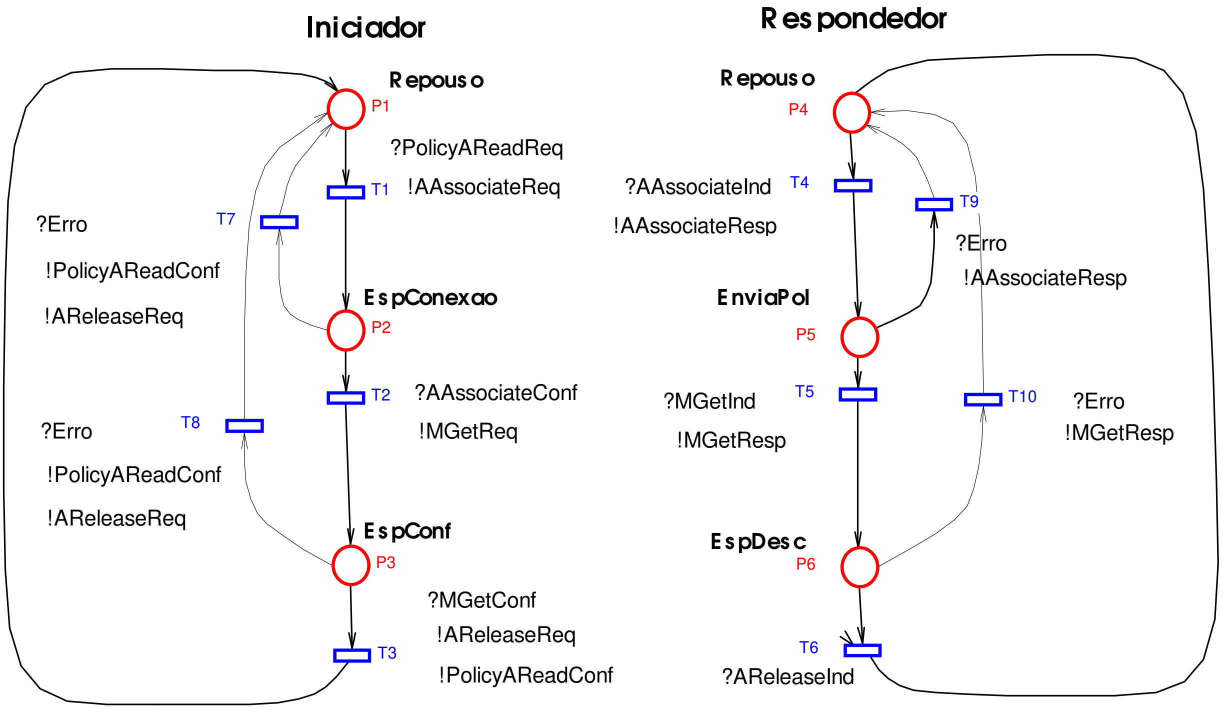

Figura 15 - Modelagem da entidade PolicyARead

A entidade PolicyMLoad está representada pela rede mostrada na Figura 16. Sua função é ler as diretrizes do gerente hierarquicamente superior, com o objetivo de atualizar as diretrizes do gerente iniciador. Inicialmente, o gerente iniciador realiza a associação da comunicação com o outro gerente, utilizando a primitiva A-Associate do ACSE. A busca de informações no gerente respondedor é realizada através da primitiva M-Get, que lê a tabela TablePolicy. O encerramento é realizado com a primitiva A-Release.

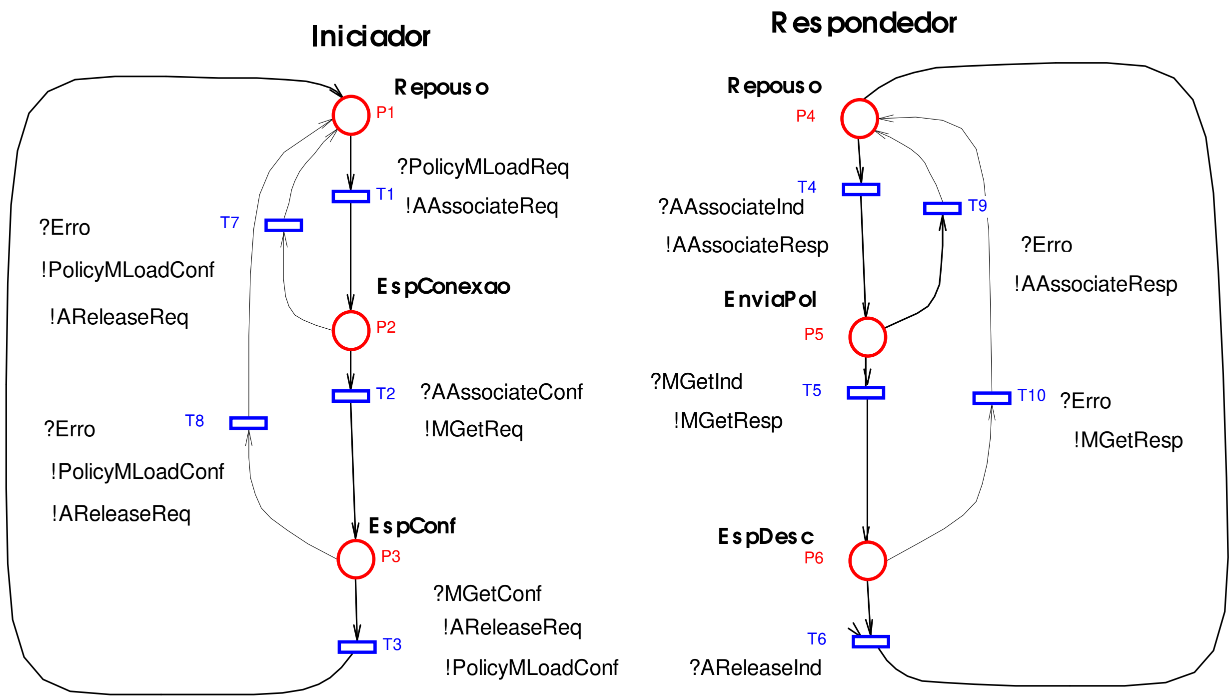

Figura 16 - Modelagem da entidade PolicyMLoad

A entidade DelegAct está representada pela rede mostrada na Figura 17. Sua função é perguntar a um gerente se pode receber a delegação de um agente e alterar as tabelas TableA dos gerentes e a tabela TableM do agente. Inicialmente, o gerente que fará a delegação realiza a associação da comunicação com o gerente respondedor, utilizando a primitiva A-Associate do ACSE. O envio de informações para o gerente é realizado através da primitiva M-Action, que altera a tabela TableA, se obtiver resposta positiva para a delegação. Após isso, é realizado o encerramento com a primitiva A-Release. Acertada a delegação, é necessário informá-la ao agente. O gerente iniciador estabelece a associação da comunicação com o agente que será delegado, utilizando a primitiva A-Associate do ACSE. O envio de informações para o agente é realizado através da primitiva M-Action, que altera a tabela TableM. O gerente iniciador altera a tabela TableA e realiza o encerramento através da primitiva A-Release.

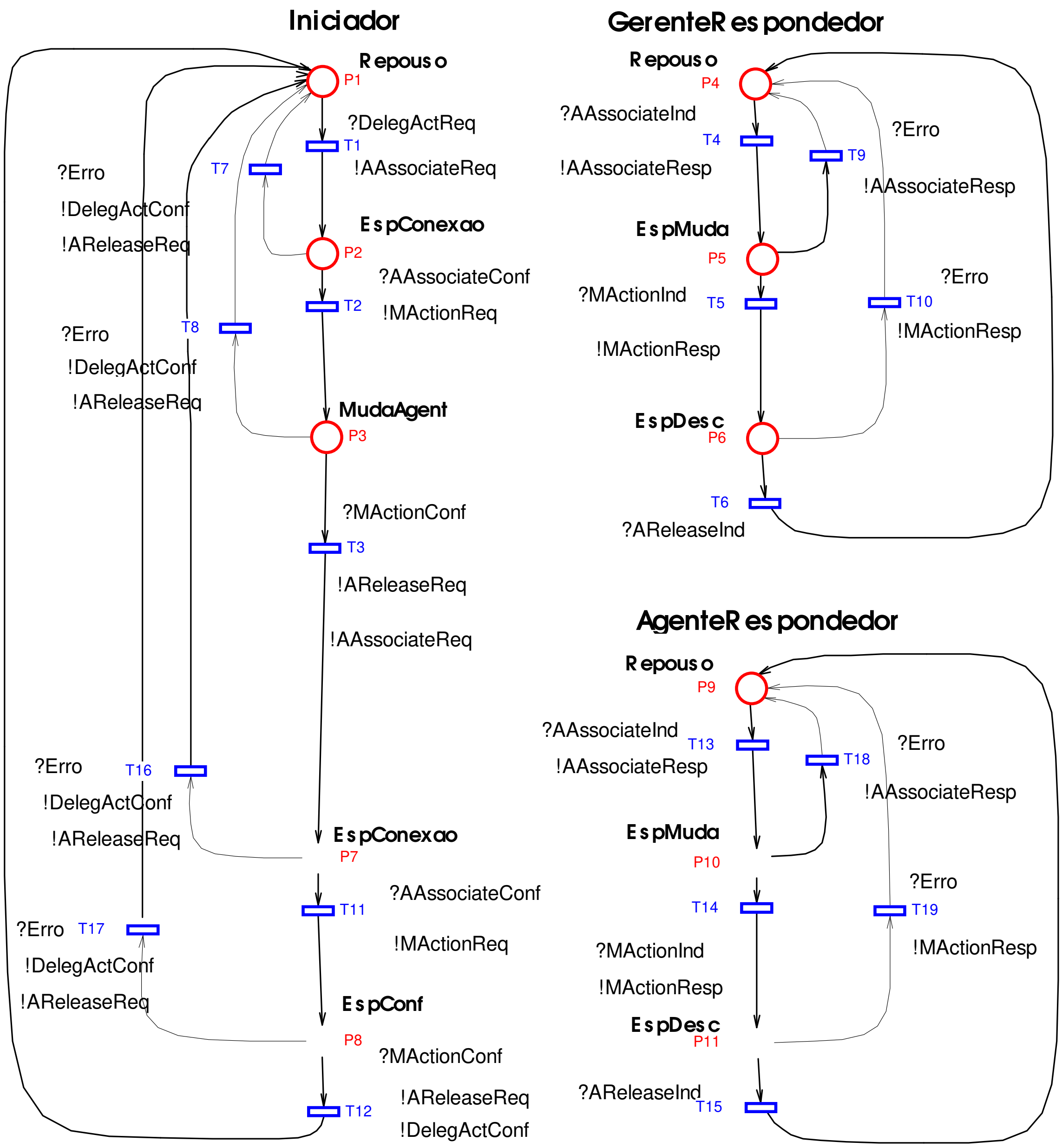

Figura 17 - Modelagem da entidade DelegAct

IV-5 Especificação Formal

O sistema de gerenciamento hierárquico proposto neste trabalho foi testado em simulação através de um estudo de caso apresentado na Figura 18. Com este modelo, realizamos os seguintes testes: ativação de um novo gerente, atualização das tabelas deste novo gerente, mudança na tabela de diretrizes e delegação de gerenciamento.

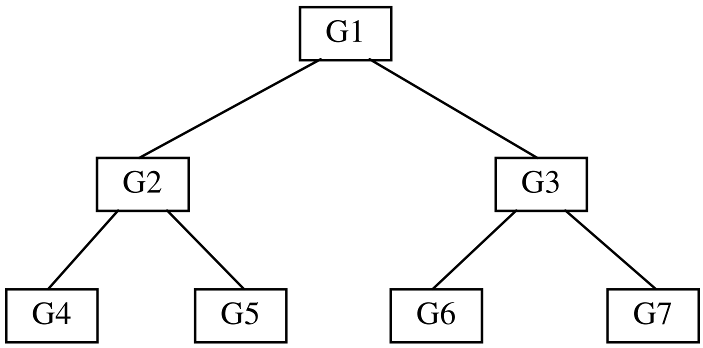

Figura 18 - Modelo para verificação do protocolo SGH

O modelo considerado, apesar de simples, permite destacar todas as interações possíveis entre gerentes, possibilitando uma verificação exaustiva do protocolo. Este modelo, implementado em linguagem Estelle, é mostrado na Figura 19. O módulo SGH pode ser instanciado quantas vezes forem necessárias, neste caso, sete vezes. O módulo Usuário permite a interface com o operador, possibilitando a montagem das SDUs conforme a especificação do serviço. O módulo CMIS emula o funcionamento de uma camada que oferece o serviço de gerenciamento, utilizando o protocolo CMIP. Este módulo apenas estabelece a comunicação entre os diversos módulos SGH, copiando as mensagens do módulo remetente para o módulo destinatário.

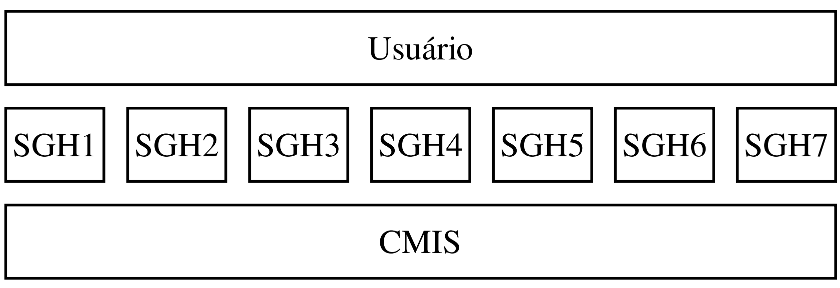

Figura 19 - Modelo de simulação em Estelle

Os nomes dos canais e módulos do programa em Estelle são apresentados na Figura 20. Esta padronização é usada em toda a especificação e permitiu a escrita de código de forma organizada. Observe-se que acima do módulo de usuário (Module User) existe um canal em aberto (USYS.SYS_Interface), necessário para permitir a entrada de dados pelo simulador. Os canais UCEP e MCEP são do tipo array para possibilitar a instanciação dos diversos módulos SGH.

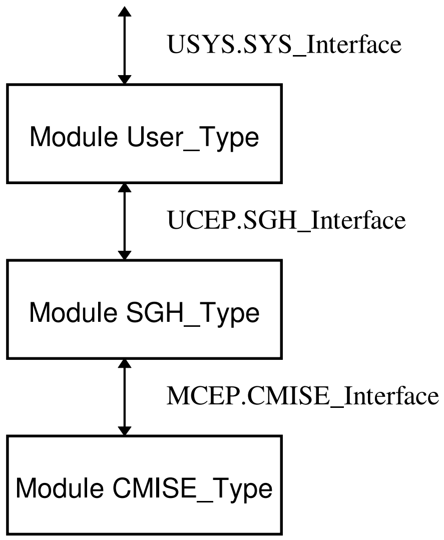

Figura 20 - Estrutura do Programa Estelle

A listagem completa do código Estelle do programa utilizado em nosso protótipo é apresentado no Anexo 1, que segue o modelo e a estrutura apresentados acima.

IV-5.1 Descrição de Variáveis

A relação de variáveis utilizadas no sistema SGH são apresentadas na Tabela 6. Todas as variáveis são do tipo Integer, porque o simulador tem problemas para imprimir string variável.

Tabela 6 - Variáveis utilizadas no SGH

Variável	Descrição da Variável
Manager	Nome do gerente, a quem deve se reportar
Agent	Nome do agente que é controlado ou o gerente um nível abaixo
TableM	Tabela de gerentes, em ordem seqüencial de submissão
TableA	Tabela de agentes
TablePolicy	Tabela de diretrizes
PolicyType	Tipo de diretriz

IV-5.2 Descrição das SDUs SGH

O formato das SDUs do serviço SGH são mostradas a seguir, com o nome do campo, o tipo de variável e uma descrição. A Tabela 7 mostra o tipo Domínio, utilizado pelas primitivas MInitPrin, ManagerInit, AgentCreate e AgentElimin.

Tabela 7 - Descrição do Tipo Domínio

Campo	Tipo de Variável	Descrição
Manager	integer	número do gerente objeto
Agent	integer	numero do agente objeto
Local	integer	número local do gerente
Authorization	string	senha de autorização
Status	integer	resultado final do comando

Observe-se que a única variável do tipo string é a Authorization, pois seu valor não é exibido, apenas comparado. Como o simulador conseguiu tratar esta estrutura, mantivemos o tipo string. O mesmo se dá nas SDUs descritas abaixo.

A Tabela 8 mostra o tipo Tabela, utilizado pelas primitivas ManagerRead, AgentRead e PolicyMRead. Este tipo é basicamente usado pelas primitivas que não implementam protocolo, restringindo-se apenas à leitura de tabelas locais.

Tabela 8 - Descrição do Tipo Tabela

Campo	Tipo de Variável	Descrição
List	array of integer	lista de parâmetros
Pointer	integer	apontador da lista
Authorization	string	senha de autorização
Status	integer	resultado final do comando

A Tabela 9 mostra o tipo Delegação, utilizado pela primitiva DelegAct.

Tabela 9 - Descrição do Tipo Delegação

Campo	Tipo de Variável	Descrição
Manager	integer	número do gerente objeto
Agent	integer	número do agente objeto
Delegation	string	comando de delegação
Authorization	string	senha de autorização
Status	integer	resultado final do comando

A Tabela 10 mostra o tipo Diretriz, utilizado pelas primitivas PolicyARead, PolicyAChange e PolicyMLoad.

Tabela 10 - Descrição do Tipo Diretriz

Campo	Tipo de Variável	Descrição
Manager	integer	número do gerente objeto
Agent	integer	numero do agente objeto
PolicyList	array of integer	lista de diretrizes
Authorization	string	senha de autorização
Status	integer	resultado final do comando

IV-5.3 Descrição das PDUs CMIS

Mostramos, nesta seção, os formatos das PDUs do serviço CMIS, com o nome do campo, o tipo de variável, uma descrição e seu uso no SGH.

Os parâmetros da primitiva A-Associate, responsável por estabelecer a associação com um elemento, são mostrados na Tabela 11.

Tabela 11 - Descrição da PDU A-Associate

Campo	Tipo Variável	Descrição	Uso no SGH
ACN	string	Aplication Context Name	Nome da Aplicação SGH
APT	string	Aplication Process Title	Nome da Primitiva SGH (Req, Ind)
AEQ	string	Aplication Entity Qualifier	Nome da Primitiva SGH (Req, Ind)
API	integer	Aplication Process Invoc Id	Nome Local
AEI	integer	Aplication Entity Invoc Id	Nome Destino
PA	integer	Presentation Arguments	Não usado
UD	string	User Data	Senha de Autorização
RAPT	string	Resp Aplic Process Title	Nome da Primitiva SGH (Resp, Conf)
RAEQ	string	Resp Aplic Entity Qualifier	Nome da Primitiva SGH (Resp, Conf)
RAPI	integer	Resp Aplic Process Invoc Id	Não usado
RAEI	integer	Resp Aplic Entity Invoc Id	Não usado
R	integer	Result	Código de Status

Os parâmetros da primitiva A-Release, responsável por encerrar a associação com um elemento, são mostrados na Tabela 12.

Tabela 12 - Descrição da PDU A-Release

Campo	Tipo Variável	Descrição	Uso no SGH
ACN	string	Aplication Context Name	Nome da Aplicação SGH
APT	string	Aplication Process Title	Nome da Primitiva SGH
AEQ	string	Aplication Entity Qualifier	Nome da Primitiva SGH
API	integer	Aplication Process Invoc Id	Nome Local
AEI	integer	Aplication Entity Invoc Id	Nome Destino

Os parâmetros da primitiva M-Get, responsável por receber dados de um elemento remoto, são mostrados na Tabela 13.

Tabela 13 - Descrição da PDU M-Get

Campo	TipoVariável	Descrição	Uso no SGH
II	string	Invoke Identifier	Nome da Primitiva SGH
LI	integer	Linked Identifier	Não usado
BOC	string	Base Object Class	Nome tabela solicitada (Req, Ind)
BOI	integer	Base Object Instance	Não usado
SC	integer	Scope	Endereço destino
F	integer	Filter	Não usado
AC	string	Access Control	Senha de Autorização
SY	integer	Synchronization	Endereço origem
AI	integer	Attribute Identifier	Não usado
MOC	string	Managed Object Class	Nome tabela solicitada (Resp, Conf)
MOI	integer	Managed Object Instance	Não usado
CT	integer	Current Time	Não usado
AL	array of integer	Attribute List	Lista de Atributos
E	integer	Error	Código de Status

Os parâmetros da primitiva M-Set, responsável por enviar dados para um elemento remoto, são mostrados na Tabela 14.

Tabela 14 - Descrição da PDU M-Set

Campo	TipoVariável	Descrição	Uso no SGH
II	string	Invoke Identifier	Nome da Primitiva SGH
LI	integer	Linked Identifier	Não usado
M	string	Mode	M-Set Confirmado ou não
BOC	string	Base Object Class	Nome tabela enviada (Req, Ind)
BOI	integer	Base Object Instance	Não usado
SC	integer	Scope	Endereço destino
F	integer	Filter	Não usado
AC	string	Access Control	Senha de Autorização
SY	integer	Synchronization	Endereço origem
MOC	string	Managed Object Class	Nome tabela enviada (Resp, Conf)
MOI	integer	Managed Object Instance	Não usado
MI	integer	Modification List	Não usado
AL	array of integer	Attribute List	Lista de Atributos
CT	integer	Current Time	Não usado
E	integer	Error	Código de Status

Os parâmetros da primitiva M-Action, responsável por enviar comando para um elemento remoto, são mostrados na Tabela 15.

Tabela 15 - Descrição da PDU M-Action

Campo	Tipo Variável	Descrição	Uso no SGH
II	string	Invoke Identifier	Nome da Primitiva SGH
LI	integer	Linked Identifier	Não usado
M	string	Mode	Modo confirmado
BOC	string	Base Object Class	Tabela enviada
BOI	integer	Base Object Instance	Não usado
SC	integer	Scope	Endereço destino
F	integer	Filter	Não usado
MOC	string	Managed Object Class	Tabela recebida
MOI	integer	Managed Object Instance	Não usado
AC	string	Access Control	Senha de Autorização
SY	integer	Synchronization	Endereço origem
AT	string	Action Type	Tipo de ação (ask ou act)
AA	integer	Action Attribute	Argumentos da ação
CT	integer	Current Time	Não usado
AR	integer	Action Result	Não usado
E	integer	Error	Código de Status

IV-5.4 Relação de Resultados de Operação (Lista de erros)

O sistema SGH apresenta dois tipos de erros. O primeiro é o tratamento de erros apresentados pelo CMIS e que são definidos em [9]. O segundo é o tratamento de erros do SGH que se referencia a erros na manipulação de dados nas tabelas. A relação de resultado de operação e de erros, com sua respectiva numeração, é apresentada na Tabela 16.

Tabela 16 - Lista de resultados de operação

Número	Grupo	Descrição
0	CMIS	Execução OK
1	CMIS	Acesso negado
2	CMIS	Conflito de instanciação de classe
3	CMIS	Limite de complexidade
4	CMIS	Chamado duplicado
5	CMIS	Instanciação de objeto duplicado
6	CMIS	Erro na lista GET
7	CMIS	Valor de argumento inválido
8	CMIS	Valor de atributo inválido
9	CMIS	Filtro inválido
10	CMIS	Instanciação de objeto inválida
11	CMIS	Falta de valor de atributo
12	CMIS	Escopo inválido
13	CMIS	Erro de argumento
14	CMIS	Ação inexistente
15	CMIS	Argumento inexistente
16	CMIS	Atributo inexistente
17	CMIS	Tipo de evento inexistente
18	CMIS	Identificador chamado inexistente
19	CMIS	Classe de objeto inexistente
20	CMIS	Instância de objeto inexistente
21	CMIS	Objeto referenciado inexistente
22	CMIS	Falha de processamento
23	CMIS	Limitação de recursos
24	CMIS	Erro na lista SET
25	CMIS	Sincronização não suportada
26	CMIS	Operação não reconhecida
27	SGH	Erro de informações
28	SGH	Endereço errado ou acesso não autorizado
29	SGH	Gerente chamado não aceitou delegação
30	SGH	Falta de espaço na tabela
31	SGH	Elemento não está na tabela

IV-6 Uso do Ambiente de Projeto

O ambiente de desenvolvimento de protocolos utilizado possibilitou ferramentas de compilação e simulação integradas. A primeira etapa, compilação do programa Estelle utilizando um compilador Modula-2, ocorreu sem problemas. Entretanto, o uso do simulador exigiu várias modificações no programa original Estelle, para que este pudesse ser simulado e testado.

A principal razão destas dificuldades está no fato de que o simulador não é adequado para suportar várias conexões entre os módulos instanciados, o que nos obrigou a simplificar os testes. As conexões foram realizadas sempre de dois em dois módulos, focalizando uma primitiva específica de cada vez. Isso, no entanto, não chegou a prejudicar qualitativamente os testes do protocolo.

O primeiro problema da simulação ocorreu na instanciação dos módulos com o comando Modvar. O simulador não aceitou um array de instanciações, forçando que cada instância do módulo SGH recebesse um nome distinto e a conexão dos módulos fosse feita uma a uma. A Figura 21 mostra um exemplo do código inicialmente escrito e o código aceito no simulador.

Código Inicial

```
modvar
   SGH: array[1..quant_MCEP] of SGH_TYPE;
```

Código Usado

```
modvar
   SGH1: SGH_TYPE;
   SGH2: SGH_TYPE;
   SGH3: SGH_TYPE;
    .....
```

Figura 21 - Problema modvar de array

O segundo problema foi a impressão de string variável. Inicialmente, definimos todos os campos da SDU SGH com string, para tornar o sistema o mais genérico possível e permitir que a nomeação de gerentes e agentes pudesse utilizar nomes (é mais confortável referenciar 'UFRJ' do que '1345', por exemplo). No entanto, o comando WriteString somente imprime strings constantes, o que nos obrigou a referenciar as entidades apenas com números. Apesar desta limitação, implementada no sistema por causa do simulador, pode-se facilmente mudar a referência para nomes na implementação definitiva. A Figura 22 mostra um exemplo do código inicialmente escrito e o código aceito no simulador.

Código Inicial

```
var
     var_string : string ;

     var_string :=  'UFRJ' ;
     WriteString  ( var_string ) ;
     WriteLn ;
```

Código Usado

```
var
     manager : integer ;

     manager := 3 ;
     WriteString ( 'Manager' ) ;
     WriteInt ( manager ) ;
     WriteLn ;
```

Figura 22 - Problema impressão de string

Nosso terceiro problema foi fazer o programa ler uma tecla do teclado, identificá-la e executar uma certa operação. Para que isso fosse possível, fomos obrigados a utilizar uma estrutura aparentemente sem sentido. A Figura 23 mostra um exemplo do código inicialmente escrito e o código aceito no simulador.

Código Inicial

```
var
     tecla : char ;

   if KeyPressed then
          tecla := ReadKey ;

   if  tecla = 'a'  then
           .....
```

Código Usado

```
var
     tecla : array [0..0] of char ;

   if KeyPressed then
          tecla[0] := ReadKey ;

   if  tecla[0] = 'a'  then
           ....
```

Figura 23 - Problema leitura de teclado

O quarto problema foi na implementação do módulo CMIS. Como este módulo apresenta vários IPs (um para cada módulo SGH), tentou-se usar o comando ANY de tal forma que qualquer mensagem chegando em qualquer MCEP pudesse ser tratada. O simulador não armazenava o número do MCEP que recebia a mensagem. Com isso, não se podia saber o originador da mensagem e não se conseguia devolver a mensagem de retorno ou uma mensagem de erro. A solução foi fazer o MCEP fixo e repetir várias vezes o trecho de programa, um para cada MCEP. A Figura 24 mostra um exemplo do código inicialmente escrito e o código com o qual o simulador funcionou corretamente.

Código Inicial

```
any NCEP

      when MCEP[NCEP]
            output MCEP[Destino] Msg ;
            output MCEP[NCEP] Erro ;
```

Código Usado

```
      when MCEP[1]
            output MCEP[Destino] Msg ;
            output MCEP[1] Erro ;

      when MCEP[2]
            output MCEP[Destino] Msg ;
            output MCEP[2] Erro ;
```

Figura 24 - Problema NCEP variável

O quinto problema foi a impossibilidade de se estabelecer duas conexões seguidas com um mesmo gerente ou de um mesmo gerente se comunicar com duas entidades distintas (uma seguida da outra). Para esclarecer melhor este problema, exemplificaremos na Figura 25.

A Figura 25(a) mostra uma situação em que o teste não pôde ser realizado: o gerente 1 se conecta com o gerente 2 e, depois de desconectados, não conseguimos que o gerente 3 se comunique com o gerente 2 (dead-lock no simulador). A Figura 25(b) mostra outra situação de impossibilidade do teste: o gerente 1 se conecta com o gerente 2, encerra a conexão e não consegue se comunicar com o gerente 3. A Figura 25(c) mostra uma situação possível de teste: o gerente 1 se conecta com o gerente 2, o gerente

2 se conecta com o gerente 3, o gerente 3 se conecta com o gerente 4, e assim sucessivamente.

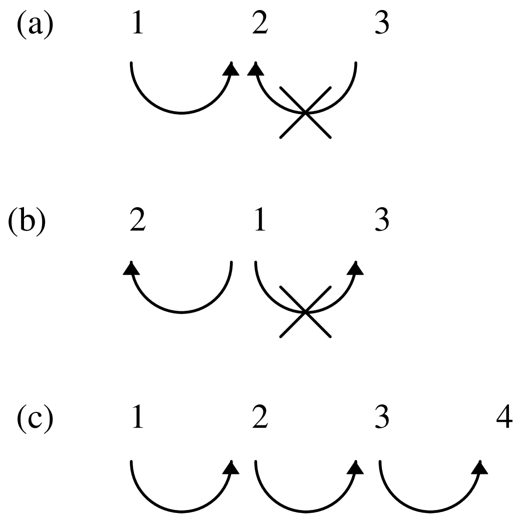

Figura 25 - Problemas de conexão com uma mesma entidade

Para tentar resolver este problema, mudamos a forma de comunicação. A primeira implementação do módulo CMIS tentava simular uma comunicação real, estabelecendo uma conexão com a primitiva A-Associate e encerrando essa conexão com a primitiva A_Release. Esta implementação foi substituída por outra que simulava uma comunicação datagrama, muito mais simples que a solução anterior. Para isso, tivemos de usar um campo nas primitivas M-Get, M-Set e M-Action, indicando o número da entidade originadora da comunicação. Esta alteração, entretanto, não possibilitou os testes.

O problema não pôde ser solucionado e acreditamos isto se deva a uma característica interna do simulador. Notamos que, a partir do momento que se estabelece uma conexão, não se consegue desconectar. Cada entidade somente aceita uma conexão por canal de entrada e uma por canal de saída e não consegue desfazê-la. Para estabelecer outra conexão, é preciso reiniciar o programa simulador.

IV-7 Simulação e Testes

Os testes de simulação realizados tentaram atingir todas as situações previstas dentro das restrições impostas pelo simulador, relativas ao problema de conexão com um mesmo gerente. Criamos então situações, em que as primitivas fossem testadas com apenas uma conexão estabelecida.

A solução para estabelecer a posição do gerente dentro da hierarquia, sem realizar as conexões pelo protocolo, resultou na criação da primitiva INIT, que simplesmente escreve os valores nas tabelas TableM, TableA e TablePolicy e na variável NomeLocal. Essa primitiva, portanto, não implementa protocolo, apenas escreve nas tabelas locais. Ela somente foi usada para realizar os testes, contornando os problemas do simulador.

O primeiro teste foi a iniciação do sistema hierárquico. Iniciamos o gerente 1 com a primitiva MInitPrin, depois iniciamos o gerente 2 em relação ao gerente 1, com a primitiva ManagerInit, e o gerente 3 em relação ao gerente 2, novamente com a primitiva ManagerInit. Após essa iniciação, pudemos testar o funcionamento através das primitivas ManagerRead, no gerente 3, e AgentRead, no gerente 2. Como resultado deste teste, a tabela TableM do gerente 3, lida através da primitiva ManagerRead, apresentou os gerentes 2 e 1, coerente com a seqüência de início do sistema. A leitura da tabela TableA do gerente 2, lida através da primitiva AgentRead, apresentou apenas o gerente 3, coerente com os resultados esperados. O relatório desta simulação, fornecido pelo simulador, encontra-se no item A2-1 do Anexo 2 - Relatórios dos Testes de Simulação.

O segundo teste foi a criação e exclusão de um elemento, utilizando as primitivas AgentCreate e AgentElimin. Com a primitiva MInitPrin, iniciamos o gerente 1 como gerente principal. Após isso, criamos o elemento 2 através da primitiva AgentCreate. Verificamos o funcionamento desta primitiva através das primitivas ManagerRead no elemento 2 e AgentRead no elemento 1. Nesta configuração, executamos a eliminação deste elemento através da primitiva AgentElimin e, novamente, verificamos o funcionamento através das primitivas ManagerRead no elemento 2 e AgentRead no elemento 1. O relatório desta simulação encontra-se no item A2-2 do Anexo 2 - Relatórios dos Testes de Simulação.

O terceiro teste ocupou-se da gerência de diretrizes, testando todas as suas primitivas. Iniciamos SGH1 como gerente principal e valor de diretriz 1, utilizando-se a primitiva MInitPrin. Iniciamos SGH2 como agente de SGH1 e com valor de diretriz 2, utilizando-se a primitiva Init. Realizamos uma leitura de diretrizes locais em SGH1 com a primitiva PolicyMRead. Fizemos a leitura das diretrizes do agente SGH2 através de SGH1 através da primitiva PolicyARead. Alteramos a diretriz do agente SGH2 para valor de diretriz 3 através de SGH1 com a primitiva PolicyAChange. O funcionamento desta pôde ser testado com a primitiva PolicyMRead em SGH2. Finalmente, fizemos uma atualização das diretrizes de SGH2 em relação às diretrizes de SGH1, através da primitiva PolicyMLoad, que tornou as diretrizes de SGH1 e SGH2 iguais ao valor 1. O relatório desta simulação encontra-se no item A2-3 do Anexo 2 - Relatórios dos Testes de Simulação.

O quarto teste simulou uma delegação. Iniciamos o gerente SGH1 com dois agentes SGH2 e SGH3, através da primitiva Init. Iniciamos também o gerente SGH4 com dois agentes SGH5 e SGH6, novamente através da primitiva Init. A intenção era executar a delegação do agente SGH2, do gerente SGH1 para SGH4, utilizando-se a primitiva DelegAct. Como esta primitiva estabelece duas conexões, uma com o gerente que receberá a delegação e outra com o agente que será delegado, aconteceu uma situação semelhante à apresentada na Figura 25(b), em que o simulador não suporta a execução. Somente foi possível testar a primeira etapa da comunicação, entre os gerentes SGH1 e SGH4. Seu funcionamento foi verificado pela leitura da tabela de agentes de SGH4, utilizando-se a primitiva AgentRead, que apresentou o agente SGH2. A segunda conexão, do gerente SGH1 para o agente SGH2, necessária para completar a delegação, não pode se realizada. O relatório desta simulação encontra-se no item A2-4 do Anexo 2 - Relatórios dos Testes de Simulação.

IV-8 Comentários

Neste Capítulo, apresentamos o projeto do Sistema de Gerenciamento Hierárquico, utilizando as técnicas de Rede Predicado Ação e Linguagem de Descrição Formal Estelle. Foram apresentados os modelos em Rede Predicado Ação, a especificação em Estelle com suas variáveis, as SDUs, PDUs e códigos de erro. Mostramos os problemas para simular o programa e finalmente descrevemos os testes realizados.

O Capítulo V apresentará as conclusões deste trabalho e fará algumas indicações para trabalhos futuros.

Capítulo V

Conclusões e Trabalhos Futuros

Este trabalho apresentou uma proposta para um sistema de gerenciamento hierárquico apropriado para grandes redes de telecomunicações. Analisamos várias propostas encontradas na literatura e aproveitamos os conceitos para fazer um sistema hierárquico coerente e flexível. Esta flexibilidade, que advém em grande parte da forma independente com que o SGH opera nos diversos elementos da rede, não provoca, no entanto, perda de controle. O uso criterioso das diretrizes garante coerência e ausência de conflitos ao conjunto. Além disso, a possibilidade de delegação de gerenciamento confere ao sistema grande estabilidade e tolerância a falhas.

O protocolo SGH proposto não apenas pode controlar qualquer quantidade de dispositivos, como também pode adequar-se dinamicamente a qualquer alteração na quantidade e localização de objetos gerenciados. Esta facilidade de modificar-se e a arquitetura adaptável tornam-no apto a gerenciar dispositivos de comunicação móvel. Este trabalho seguiu todas as diretrizes determinadas pelo ITU e produziu uma aplicação consistente com o modelo TMN.

O sistema SGH oferece também recursos de segurança. Como cada gerente enxerga apenas a camada inferior e os critérios para troca de gerente são definidos rigidamente, então a capacidade de acesso de cada gerente pode ser limitada em qualquer nível. Além disso, o uso de técnicas de criptografia, disponíveis no protocolo CMIP, garante segurança de acesso ao gerenciamento da rede.

O uso do modelo em Redes Predicado Ação e da linguagem de especificação formal Estelle facilitou o trabalho de codificação. Mesmo com os problemas encontrados para realizar as simulações, reconhecemos que a codificação utilizada foi mais viável para os nossos fins do que a codificação em linguagem C. A especificação formal foi a maneira adequada para descrever esse sistema em um alto nível de abstração, podendo futuramente ser convertido para código de uma linguagem de programação através de programas apropriados [29].

O protótipo desenvolvido permitiu verificar a funcionalidade do protocolo, mesmo com as limitações impostas pela ferramenta de simulação de especificação em

Estelle. Estas limitações não permitiram um teste complexo, mas todas as partes do sistema foram verificadas isoladamente.

Para a conclusão deste trabalho é necessário realizar duas etapas que estão sendo conduzidas por DELFINO em [30]. A primeira consiste em uma verificação exaustiva do protocolo usando uma ferramenta de geração de seqüências de teste, que possibilitará testar o sistemas utilizando todas as combinações de estados e transições de entrada. A segunda etapa é a implementação do protocolo a partir da especificação em Estelle.

Como trabalho futuro, nesta linha, acreditamos que seria relevante e necessário um aprofundamento na gerência de diretrizes. Esta técnica possibilita a diminuição da quantidade de mensagens trocadas entre gerentes e conseqüentemente uma diminuição de tempo na execução de comandos de configuração. A utilização de Inteligência Artificial e Lógica Fuzzy permitiria um tratamento de diretrizes eficiente e rápido. Assim, seria possível que cada gerente do sistema pudesse tomar grande parte das decisões sozinho, diminuindo inclusive a necessidade de interferência humana.

Paralelo a isso, os testes exaustivos com o simulador Estelle possibilitaram a identificação de falhas no programa de simulação, o que oferece uma indicação de trabalho para a melhoria desta ferramenta.

Capítulo VI

Referências

[1] CASE, J., McCLOGHRIE, K., ROSE, M., et al. *Manager-to-Manager Management Information Base. RFC 1451*. USA, abril 1993.

[2] ITU - INTERNATIONAL TELECOMMUNICATION UNION. *Recomendation M.3010: Principles for a Telecommunications Management Network*. ITU, outubro 1992.

[3] GROUPE BULL. *ISM OpenMaster SupraManager Specification*. Groupe Bull, França, 1996.

[4] DEC - DIGITAL EQUIPAMENT CORPORATION. *TeMIP Overview*. DEC, França, 1997.

[5] FERNANDEZ, M. P., PEDROZA, A. C. "Especificação de um Protocolo para Gerenciamento Hierárquico em Ambiente TMN" In: *15° Simpósio Brasileiro de Redes de Computadores SBRC 97*. São Carlos, pp. 3-18, maio 1997.

[6] SIEGL, M. R., TRAUSMUTH, G. *Hierarchical Network Management - A Concept and its Prototype in SNMPv2*. Technical Report of University of Technology TU-Wien. Áustria, maio 1995.

[7] SCHALLER, N. H. "A concept for hierarchical, decentralized management of the physical configuration in the Internet", In: *Proceedings of KiVS 95*. Chemnitz, 20-24 setembro 1995.

[8] CASE, J. D., FEDOR, M. S., SCHOFFSTALL, M. L., at al. *A simple network management protocol (SNMP). RFC 1157*. USA, maio 1990.

[9] ISO - INTERNATIONAL STANDARDS ORGANIZATION. *9596 Information Technology, Open System Interconnection - Common Management Information Protocol - Part 1: Specification*. ISO/IEC, maio 1990.

[10] CASE, J. D., McCLOGHRIE, K., ROSE, M. T., at al. *Introduction to version 2 of the Internet-standard Network Management Framework. RFC 1441*. USA, abril 1993.

[11] BLACK, U. D. *Network Management Standards - SNMP, CMIP, TMN, MIBs and Object Libraries - 2nd Edition*. 2 ed. McGraw-Hill.USA, 1995.

[12] SLOMAN, M. "Management issues for distributed service", In: *Proceedings of IEEE 2nd International Workshop on Services in Distributed and Networked Environment*. Canada, pp. 52-59, 5-6 junho 1995.

[13] SLOMAN, M., MAGEE, K., TWIDLE, K., at al. "An arquitecture for managing distributed systems", In: *Proceedings of 4th IEEE Workshop on Future Trends of Distributed Computing Systems*. Lisboa, pp. 40-46, 22-24 setembro 1993.

[14] FERNANDEZ, M. P., PEDROZA, A. C. "Sistema para Gerenciamento Hierárquico em Ambiente TMN" In: *15° Simpósio Brasileiro de Telecomunicações SBT 97*. Recife, pp. , setembro 1997.

[15] SPIELMANN, L., KRAISER, T. *Projeto de um Sistema de Gerenciamento Hierárquico de Redes sobre uma Plataforma Distribuída HORB.* Projeto Final de Curso. Departamento de Eletrônica EE/UFRJ. Rio de Janeiro, março 1997.

[16] PUTTER, P., BISHOP, J., ROOS, J. "Towards policy driven systems management", In: *IFIP/IEEE International Symposium on Integrated Network Management*. Santa Barbara, USA, pp. 69-80, 1-5 maio 1995.

[17] MARRIOTT, D.A., SLOMAN, M.S., YIALELIS, N. "Management Policy Service for Distributed Systems". *Imperial College Research Report DoC 95/10.* Inglaterra, outubro 1995.

[18] MOFFETT, J., SLOMAN, M. "Policy hierarchies for distributed systems management", *IEEE Journal on Selected Areas in Communications*. USA, v. 11, n.9, pp. 1404-1414, dezembro 1993.

[19] MARRIOTT, D.A., MANSOURI-SAMANI, M., SLOMAN, M.S. "Specification of Management Policies". In: *Proceedings of the IFIP/IEEE International Workshop on Distributed Systems: Operation and Management*. França, 10-12 outubro 1994.

[20] MOFFETT, J., SLOMAN, M. "Policy Conflict Analysis in Distributed System Management", *Journal o Organizational Computing*. Inglaterra, pp. 115-134, abril 1994.

[21] ALPERS, B., PLANSKY, H. "Concepts and Aplication of Policy-Based Management", In: *IFIP/IEEE International Symposium on Integrated Network Management*. Santa Barbara, USA, pp. 57-68, 1-5 maio 1995.

[22] WIES, R. "Policy Definition and Classification: Aspects, Criteria and Examples", In: *Proceedings of the IFIP/IEEE International Workshop on Distributed Systems: Operation and Management*. França, 10-12 outubro 1994.

[23] WIES, R. "Policies in Network and Systems Management - Formal Definition and Architecture", *Journal of Network and Systems Management*. Plenum Publishing Corp. USA, v. 2, n. 1, pp. 63-83, março 1994.

[24] WIES, R. "Using a Classification of Management Policies for Policy Specification and Policy Transformation", In: *IFIP/IEEE International Symposium on Integrated Network Management*. Santa Barbara, USA, pp. 44-56, 1-5 maio 1995.

[25] YEMINI, Y., GOLDSZMIDT, G.. "Network management by delegation", In: *The Second International Symposium on Integrated Network Management*. Washington, pp. 95-107, abril 1991.

[26] MOFFETT, J. D., SLOMAN, M. S. "Delegation of authority", In: *IFIP/IEEE 2nd International Symposium on Integrated Network Management*. Washington, pp. 595-606, abril 1991.

[27] ITU - INTERNATIONAL TELECOMMUNICATION UNION. *Recomendation M.3100: Generic Network Information Model*. ITU, outubro 1992.

[28] DIAZ, M., at al. *The Formal Description Technique ESTELLE*. 1 ed.North-Holland.USA, 1989.

[29] SANTANA, A. C., PRADO, A. F., SOUZA, W. L. "Utilização do Paradigma Draco para Implementar Especificação Estelle na Linguagem C++" In: *15° Simpósio Brasileiro de Redes de Computadores SBRC 97*. São Carlos, pp. 118-134, maio 1997.

[30] DELFINO, G.M. *Verificação e Implementação do Protocolo de Gerenciamento Hierárquico (SGH).* Projeto Final de Curso. Departamento de Eletrônica EE/UFRJ. A ser defendido em 1998.

Anexo 1

Listagem do Programa em Estelle

Este anexo traz a listagem completa do programa em Estelle utilizada em nosso protótipo, além de todas as rotinas que permitiram a simulação. O programa foi adaptado para teste em nosso sistema de simulação, devidamente comentado. Todo o programa, aliás, encontra-se exaustivamente comentado e, com as explicações apresentadas no Capítulo 4, poderá ser facilmente compreendido.

```
specification SGH systemactivity;
  default individual queue; timescale seconds;

(**********************************************************)
(*                                                        *)
(*                   ESPECIFICACAO SGH                    *)
(*                                                        *)
(**********************************************************)

  {$ external NetUserTMN2}

(**********************************************************)
(*                                                        *)
(*                  DEFINICAO DE TIPOS                    *)
(*                                                        *)
(**********************************************************)


(************* DEFINICAO TIPOS GENERICOS ******************)

type

     string = array [0..30] of char;
     longstring = array [0..50] of char;

(*************** DEFINICAO DE CONSTANTES ******************)

const
     TamTableM = 10 ;        { Tamanho da Tabela de Gerentes }
     TamTableA = 30 ;        { Tamanho da Tabela de Agentes }
     TamTablePolicy = 30 ;   { Tamanho da Tabela de Diretrizes}
     TamMaxTab = 30 ;        { Tamanho maximo das Tabelas }
     quant_MCEP = 7 ;        { quantidade de MCEP }

{  Status 0 : 'Execucao OK',
{  Erro 1 : 'CMIS - Acesso Negado'
{  Erro 2 : 'CMIS - Conflito de Instanciacao de Classe'
{  Erro 3 : 'CMIS - Limite de Complexidade'
{  Erro 4 : 'CMIS - Invocacao Duplicada'
{  Erro 5 : 'CMIS - Instanciacao de Objeto Duplicado'
{  Erro 6 : 'CMIS - Erro na lista GET'
{  Erro 7 : 'CMIS - Valor de Argumento Invalido'
{  Erro 8 : 'CMIS - Valor de Atributo Invalido'
{  Erro 9 : 'CMIS - Filtro Invalido'
{  Erro 10 : 'CMIS - Instanciacao de Objeto Invalida'
{  Erro 11 : 'CMIS - Falta de Valor de Atributo'
{  Erro 12 : 'CMIS - Escopo Invalido'
{  Erro 13 : 'CMIS - Erro de Argumento'
{  Erro 14 : 'CMIS - Acao Inexistente'
{  Erro 15 : 'CMIS - Argumento Inexistente'
{  Erro 16 : 'CMIS - Atributo Inexistente'
{  Erro 17 : 'CMIS - Tipo de Evento Inexistente'
{  Erro 18 : 'CMIS - Identificador Invocado Inexistente'
{  Erro 19 : 'CMIS - Classe de Objeto Inexistente'
{  Erro 20 : 'CMIS - Instacia de Objeto Inexistente'
{  Erro 21 : 'CMIS - Objeto Referenciado Inexistente'
{  Erro 22 : 'CMIS - Falha de Processamento'
{  Erro 23 : 'CMIS - Limitacao de Recursos'
{  Erro 24 : 'CMIS - Erro na Lista SET'
{  Erro 25 : 'CMIS - Sincronizacao nao Suportada'
{  Erro 26 : 'CMIS - Operacao nao reconhecida'
{  Erro 27 : 'SGH - Erro de informacoes'
{  Erro 28 : 'SGH - Endereco Errado ou Acesso nao Autorizado'
```

```
{  Erro 29 : 'SGH - Gerente chamado nao aceitou delegacao'
{  Erro 30 : 'SGH - Estouro na Tabela'
{  Erro 31 : 'SGH - Elemento nao esta na Tabela'          }

type
     MCEP_FAIXA_TYPE = 1..quant_MCEP; {determina faixa de MCEP}

(************** DEFINICAO TIPOS DE SDU SGH ******************)

type
    DOMINIO_TYPE = RECORD
           Manager: integer ;
           Agent: integer ;
           Local: integer ;
           Autorization: string ;
           Status: integer ;
      END ;

    TABELA_TYPE = RECORD
           List: array [1..TamMaxTab] of integer ;
           Autorization: string ;
           Status: integer ;
           Pointer: integer ;
      END ;

    DELEG_TYPE = RECORD
           Manager: integer ;
           Agent: integer ;
           Delegation: string;
           Autorization: string ;
           Status: integer ;
      END ;

    POLICY_TYPE = RECORD
           Manager: integer ;
           Agent: integer ;
           PolicyList:  array [1..TamTablePolicy] of integer ;
           Autorization: string ;
           Status: integer ;
      END ;

(************** DEFINICAO TIPOS DE PDU CMISE ****************)

    A_ASSOCIATE_TYPE = RECORD
           ACN : string ;       { Aplication Context Name }
           APT : string  ;      { Aplication Process Title }
           AEQ : string ;       { Aplication Entity Qualifier }
           API : integer ;      { Aplication Process Invoc Id }
           AEI : integer ;      { Aplication Entity Invoc Id }
           PA : integer ;       { Presentation Arguments }
           UD : string ;        { User Data }
           RAPT : string ;      { Resp Aplic Process Title }
           RAEQ : string ;      { Resp Aplic Entity Qualifier }
           RAPI : integer ;     { Resp Aplic Process Invoc Id }
           RAEI : integer ;     { Resp Aplic Entity Invoc Id }
           R: integer ;         { Result }
     END ;

    A_RELEASE_TYPE = RECORD
           ACN : string ;       { Aplication Context Name }
           APT : string ;       { Aplication Process Title }
           AEQ : string ;       { Aplication Entity Qualifier }
     API : integer ;      { Aplication Process Invoc Id }
```

```
        AEI : integer ;        { Aplication Entity Invoc Id }
     END ;

   M_GET_TYPE = RECORD
        II: string ;           { Invoke Identifier }
        LI: integer ;          { Linked Identifier }
        BOC: string ;          { Base Object Class }
        BOI: integer ;         { Base Object Instance }
        SC: integer ;          { Scope }
        F: integer ;           { Filter }
        AC: string ;           { Access Control }
        SY: integer ;          { Synchronization }
        AI: integer ;          { Attribute Identifier }
        MOC: string ;          { Mananged Object Class }
        MOI: integer ;         { Managed Object Instance }
        CT: integer ;          { Current Time }
        AL: array[1..TamMaxTab] of integer ;
                                { Attribute List }
        E: integer ;           { Error }
     END ;

   M_SET_TYPE = RECORD
        II: string ;           { Invoke Identifier }
        LI: integer ;          { Linked Identifier }
        M: string ;            { Mode }
        BOC: string ;          { Base Object Class }
        BOI: integer ;         { Base Object Instance }
        SC: integer ;          { Scope }
        F: integer ;           { Filter }
        AC: string ;           { Access Control }
        SY: integer ;          { Synchronization }
        MOC: string ;          { Mananged Object Class }
        MOI: integer ;         { Managed Object Instance }
        MI: integer ;          { Modification List }
        AL: array [1..TamMaxTab] of integer;
                                { Attribute List }
        CT: integer ;          { Current Time }
        E: integer ;           { Error }
     END ;

   M_ACTION_TYPE = RECORD
        II: string ;           { Invoke Identifier }
        LI: integer ;          { Linked Identifier }
        M: string ;            { Mode }
        BOC: string;           { Base Object Class }
        BOI: integer ;         { Base Object Instance }
        SC: integer ;          { Scope }
        F: integer ;           { Filter }
        MOC: string ;          { Mananged Object Class }
        MOI: integer ;         { Managed Object Instance }
        AC: string ;           { Access Control }
        SY: integer ;          { Synchronization }
        AT: string ;           { Action Type }
        AA: integer ;          { Action Attribute }
        CT: integer ;          { Current Time }
        AR: integer ;          { Action Result }
        E: integer ;           { Error }
     END ;

(*********** DEFINICAO INTERFACE COM TECLADO E VIDEO *********)

  function KeyPressed : boolean; primitive;
  function ReadKey : char; primitive;
```

```
  procedure WriteLn; primitive;
  procedure WriteInt ( i : integer ); primitive;
  procedure ReadInt ( var i : integer ); primitive;
  procedure ReadString ( var s : string ); primitive;
  procedure WriteString ( s : longstring ); primitive;
  procedure ClrScr; primitive;

(**************************************************************)
(*                                                            *)
(*                   DEFINICAO DE CANAIS                      *)
(*                                                            *)
(**************************************************************)

(***** Canal para entrada de opcoes no sistema de testes  ****)

  channel SYS_interface( user, provider);
    by user :
      INIT_TEST;
      INICMANPRIN;
      INICIAR;
      DELEGAR;
      LE_AGENTE;
      LE_GERENTE;
      POL_MREAD;
      POL_AREAD;
      POL_MLOAD;
      POL_ACHANGE;
      CRIA_AG;
      ELIM_AG;

(***************** Canal entre SGH e usuario **************)

  channel SGH_interface(user,provider);
    by user :
      InitReq(SDU:TABELA_TYPE);
      MInitPrinReq(SDU:DOMINIO_TYPE);
      ManagerInitReq(SDU:DOMINIO_TYPE);
      ManagerReadReq(SDU:TABELA_TYPE);
      AgentReadReq(SDU:TABELA_TYPE);
      AgentEliminReq(SDU:DOMINIO_TYPE);
      AgentCreateReq(SDU:DOMINIO_TYPE);
      DelegActReq(SDU:DELEG_TYPE);
      PolicyMReadReq(SDU:TABELA_TYPE);
      PolicyAReadReq(SDU:POLICY_TYPE);
      PolicyAChangeReq(SDU:POLICY_TYPE);
      PolicyMLoadReq(SDU:POLICY_TYPE);

      ManagerInitResp(SDU:DOMINIO_TYPE);
      AgentEliminResp(SDU:DOMINIO_TYPE);
      AgentCreateResp(SDU:DOMINIO_TYPE);
      DelegActResp(SDU:DELEG_TYPE);

    by provider :
      ManagerInitInd(SDU:DOMINIO_TYPE);
      AgentEliminInd(SDU:DOMINIO_TYPE);
      AgentCreateInd(SDU:DOMINIO_TYPE);
      DelegActInd(SDU:DELEG_TYPE);

      InitConf(SDU:TABELA_TYPE);
      MInitPrinConf(SDU:DOMINIO_TYPE);
      ManagerInitConf(SDU:DOMINIO_TYPE);
      ManagerReadConf(SDU:TABELA_TYPE);
      AgentReadConf(SDU:TABELA_TYPE);
```

```
      AgentEliminConf(SDU:DOMINIO_TYPE);
      AgentCreateConf(SDU:DOMINIO_TYPE);
      DelegActConf(SDU:DELEG_TYPE);
      PolicyMReadConf(SDU:TABELA_TYPE);
      PolicyAReadConf(SDU:POLICY_TYPE);
      PolicyAChangeConf(SDU:POLICY_TYPE);
      PolicyMLoadConf(SDU:POLICY_TYPE);

(***************** Canal entre SGH e CMISE ***************)

  channel CMISE_interface(user,provider);

    by user :
      A_AssociateReq(PDU:A_ASSOCIATE_TYPE);
      M_GetReq(PDU:M_GET_TYPE);
      M_SetReq(PDU:M_SET_TYPE);
      M_ActionReq(PDU:M_ACTION_TYPE);
      A_ReleaseReq(PDU:A_RELEASE_TYPE);
      A_AssociateResp(PDU:A_ASSOCIATE_TYPE);
      M_GetResp(PDU:M_GET_TYPE);
      M_SetResp(PDU:M_SET_TYPE);
      M_ActionResp(PDU:M_ACTION_TYPE);

    by provider :
      A_AssociateInd(PDU:A_ASSOCIATE_TYPE);
      M_GetInd(PDU:M_GET_TYPE);
      M_SetInd(PDU:M_SET_TYPE);
      M_ActionInd(PDU:M_ACTION_TYPE);
      A_ReleaseInd(PDU:A_RELEASE_TYPE);
      A_AssociateConf(PDU:A_ASSOCIATE_TYPE);
      M_GetConf(PDU:M_GET_TYPE);
      M_SetConf(PDU:M_SET_TYPE);
      M_ActionConf(PDU:M_ACTION_TYPE);

(*************************************************************)
(*                                                           *)
(*                ESPECIFICACAO DO MODULO SGH                *)
(*                                                           *)
(*************************************************************)

  module SGH_TYPE activity;

  ip UCEP : SGH_interface(provider);
     MCEP : CMISE_interface(user);
  end;

  body SGH_ENTITY_BODY for SGH_TYPE;

    const
Autorization = 'i';     { Codigo de autorizacao de acesso }

(************** Variaveis Locais Temporarias ***************)
    var
     temp    : integer;      { variavel temporaria }
     status  : integer;
     i,j     : integer;      { variavel de contador }

(**** Variaveis Temporarias do nome do Gerente e Agente ******)
     Manager: integer ;
     Agent: integer ;

(*************** Endereco Origem e Destino ****************)
     LocalName: integer ;    { Nome do gerente local }
```

```
     Destiny: integer ;      { Nome do gerente destino }

(************** Tabela de Gerentes e Apontador ***************)
     TableM: array [1..TamTableM] of integer;
     ManagerPointer: integer ; { posicao do gerente atual }

(************** Tabela de Agentes e Apontador ***************)
     TableA: array [1..TamTableA] of integer ;
     AgentPointer: integer ;   { posicao do ultimo agente }

(************** Tabela de Diretrizes e Apontador *************)
     TablePolicy: array [1..TamTablePolicy] of integer ;
     PolicyPointer: integer ;  { posicao da ultima diretriz }

(******************** Definicao de SDUs  *******************)
     SDU_Init:TABELA_TYPE ;
     SDU_MInitPrin: DOMINIO_TYPE ;
     SDU_ManagerInit: DOMINIO_TYPE ;
     SDU_ManagerRead: TABELA_TYPE ;
     SDU_AgentCreate: DOMINIO_TYPE ;
     SDU_AgentElimin: DOMINIO_TYPE ;
     SDU_AgentRead: TABELA_TYPE ;
     SDU_DelegAct: DELEG_TYPE ;
     SDU_PolicyMRead: TABELA_TYPE ;
     SDU_PolicyARead: POLICY_TYPE ;
     SDU_PolicyAChange: POLICY_TYPE ;
     SDU_PolicyMLoad: POLICY_TYPE ;

(******************** Definicao de PDUs  *******************)
     PDU_A_Associate: A_ASSOCIATE_TYPE ;
     PDU_M_Get: M_GET_TYPE ;
     PDU_M_Set: M_SET_TYPE ;
     PDU_M_Action: M_ACTION_TYPE ;
     PDU_A_Release: A_RELEASE_TYPE ;


(****************** Relacao de Estados ********************)
    state
     REPOUSO,                  { Repouso Geral }

     ESP_CONEC_1,              { ManagerInit Iniciador}
     ESP_TABM1,
     ESP_TABPOL1,
     ENV_TABM1,                { ManagerInit Respondedor }
     ENV_TABPOL1,
     ESP_DESC1,

     ESP_CONEC2,               { AgentCreate Iniciador}
     ENV_TABA2,
     ENV_TABPOL2,
     REC_TABA2,                { AgentCreate Respondedor }
     REC_TABPOL2,
     ESP_DESC2,

     ESP_CONEC3,               { AgentElimin Iniciador }
     ENV_AELIM,
     ESP_AELIM,                { AgentElimin Respondedor }
     ESP_DESC3,

     ESP_CONECM,               { DelegAct Iniciador }
     ESP_ACTIONM,
     MUDA_AGENT,
     ESP_CONECA,
```

```
     ESP_ACTIONA,
     ESP_ACTION,              { DelegAct Respondedor }
     ESP_DESACT,

     ESP_CONEC_10,            { PolicyARead Iniciador}
     ESP_TABPOL10,
     ENV_TABPOL10,            { PolicyARead Respondedor }
     ESP_DESC10,

     ESP_CONEC_11,            { PolicyAChange Iniciador}
     ESP_TABPOL11,
     ENV_TABPOL11,            { PolicyAChange Respondedor }
     ESP_DESC11,

     ESP_CONEC_12,            { PolicyMLoad Iniciador}
     ESP_TABPOL12,
     ENV_TABPOL12,            { PolicyMLoad Respondedor }
     ESP_DESC12;


(************ Inicializacao de SGH_ENTITY_BODY ***************)

    initialize

     to REPOUSO
     name RepousoGeral:

(********** Inicia todas as tabelas e apontadores ************)

     begin
          AgentPointer := 0 ;
          ManagerPointer :=0 ;
          PolicyPointer :=0;
          for i:=1 to TamTableM do
          begin
             TableM[i]:=0;
          end;
          for i:=1 to TamTableA do
          begin
             TableA[i]:=0;
          end;
          for i:=1 to TamTablePolicy do
          begin
             TablePolicy[i]:=0;
          end;
     end;

     trans

(************** Init - Iniciador/Respondedor ****************)

{ Rotina exclusiva para teste }

     from REPOUSO
          when UCEP.InitReq(SDU_Init)

          to REPOUSO
               name  Inicia:
               begin
                    TableM[1] := SDU_Init.List[1];
                    TableM[2] := SDU_Init.List[2];
                    TableM[3] := SDU_Init.List[3];
                    ManagerPointer := SDU_Init.List[4];
```

```
                TableA[1] := SDU_Init.List[5];
                TableA[2] := SDU_Init.List[6];
                TableA[3] := SDU_Init.List[7];
                TableA[4] := SDU_Init.List[8];
                TableA[5] := SDU_Init.List[9];
                AgentPointer:= SDU_Init.List[10];
                SDU_Init.Status := 0 ;
                TablePolicy[1] := SDU_Init.List[11];
                LocalName:=SDU_Init.List[12];
                PolicyPointer:= 1;
                output UCEP.InitConf(SDU_Init);
           end;

(************ MInitPrin - Iniciador/Respondedor ************)

     from REPOUSO
          when UCEP.MInitPrinReq(SDU_MInitPrin)

          to REPOUSO
               name  IniciaGerPrinc:
               begin
                    LocalName := SDU_MInitPrin.Local;
                    TablePolicy[1] := SDU_MInitPrin.Agent;
                    PolicyPointer:= 1;
                    SDU_MInitPrin.Status := 0 ;
                    output UCEP.MInitPrinConf(SDU_MInitPrin);
               end;


(**************** ManagerInit - Iniciador ****************)


     from REPOUSO              {usuario solicita inicio }
          when UCEP.ManagerInitReq(SDU_ManagerInit)

               {******* Parametros de A_Associate :
                         ACN: Nome da Aplicacao SGH
                         APT,AEQ: ManagerInit
                         API: Nome local
                         AEI: Nome gerente chamado
                         UD: Controle de Acesso -autorizacao}

          to ESP_CONEC_1
               name ConexaoManagerInit:
               begin
                    Manager := SDU_ManagerInit.Manager ;
                    Destiny := SDU_ManagerInit.Manager ;
                    LocalName := SDU_ManagerInit.Local ;
                    PDU_A_Associate.ACN := 'SGH' ;
                    PDU_A_Associate.APT := 'ManagerInit' ;
                    PDU_A_Associate.AEQ := 'ManagerInit' ;
                    PDU_A_Associate.API := LocalName ;
                    PDU_A_Associate.AEI := Destiny ;
                    PDU_A_Associate.UD := Autorization ;
                  output MCEP.A_AssociateReq(PDU_A_Associate);
               end;

     from ESP_CONEC_1

          when MCEP.A_AssociateConf(PDU_A_Associate)

          provided    ( PDU_A_Associate.R = 0 ) and
                      ( PDU_A_Associate.RAPT = 'ManagerInit' )
```

```
    to ESP_TABM1
         name  PedeTableMManagerInit:

         {****** Parametros de M-Get
         {       II : Nome da primitiva
         {       BOC: Tabela solicitada
         {       SC: Endereco destino
         {       SY: Endereco origem
         {       AC: Controle de acesso - autorizacao }
         begin
              PDU_M_Get.II := 'ManagerInit' ;
              PDU_M_Get.BOC:= 'TableM' ;
              PDU_M_Get.AC:= Autorization ;
              PDU_M_Get.SC := Destiny ;
              PDU_M_Get.SY := LocalName ;
              output MCEP.M_GetReq(PDU_M_Get);
         end;

    provided     ( PDU_A_Associate.R = 0 ) and
                 ( PDU_A_Associate.APT <> 'ManagerInit' )
    to REPOUSO                { Erros do SGH }
         name ErroSGHManagerInit1:
         begin
             SDU_ManagerInit.Status := 27 ;
            outputUCEP.ManagerInitConf(SDU_ManagerInit);
         end;

    provided        ( PDU_A_Associate.R <> 0 )
    to REPOUSO          { Erros do CMISE }
         name ErroCMISEManagerInit1:
         begin
          SDU_ManagerInit.Status := PDU_A_Associate.R ;
          output UCEP.ManagerInitConf(SDU_ManagerInit) ;
         end;

from ESP_TABM1

    when MCEP.M_GetConf(PDU_M_Get)

         {****** Parametros de M-Get
                   II: Nome da primitiva
                   MOC: Nome da Tabela enviada
                   AL: Tabela enviada
                   E: Erro         }

    provided   ( PDU_M_Get.E = 0 ) and
               ( PDU_M_Get.II= 'ManagerInit' ) and
               ( PDU_M_Get.MOC= 'TableM' )
    to ESP_TABPOL1
         name  PedeTablePolicyManagerInit:
         begin
              for i:=1 to TamTableM do
                   TableM[i] := PDU_M_Get.AL[i];
              ManagerPointer := PDU_M_Get.F ;

    {****** Parametros de M-Get para pedir TablePolicy
              II : Nome da primitiva
              BOC: Tabela solicitada
              SC: Endereco destino
              SY: Endereco origem
              AC: Controle de acesso - autorizacao    }

              PDU_M_Get.II := 'ManagerInit' ;
```

```
                PDU_M_Get.BOC := 'TablePolicy' ;
                PDU_M_Get.AC := Autorization ;
                PDU_M_Get.SC := Destiny ;
                PDU_M_Get.SY := LocalName ;
                output MCEP.M_GetReq(PDU_M_Get);
           end;

      provided   ( PDU_M_Get.E = 0 ) and
                 (( PDU_M_Get.II <> 'ManagerInit' ) or
                 ( PDU_M_Get.MOC <> 'TableM' ))
      to REPOUSO              { Erros do SGH }
           name ErroSGHManagerInit2:
           begin
              SDU_ManagerInit.Status := 27 ;
             output UCEP.ManagerInitConf(SDU_ManagerInit);
           end;

      provided        ( PDU_M_Get.E <> 0 )
      to REPOUSO              { Erros do CMISE }
           name ErroCMISEManagerInit2:
           begin
              SDU_ManagerInit.Status := PDU_M_Get.E ;
            output UCEP.ManagerInitConf(SDU_ManagerInit) ;
           end;

 from ESP_TABPOL1

      when MCEP.M_GetConf(PDU_M_Get)

           {****** Parametros de M-Get
                     II: Nome da primitiva
                     MOC: Nome da Tabela enviada
                     AL: Tabela enviada
                     E: Erro         }

      provided   ( PDU_M_Get.E = 0 ) and
                 ( PDU_M_Get.II= 'ManagerInit' ) and
                 ( PDU_M_Get.MOC= 'TablePolicy' )
      to REPOUSO
           name  DesconectaManagerInit:
           begin
                for i:=1 to TamTablePolicy do
                     TablePolicy[i] := PDU_M_Get.AL[i];
                PolicyPointer := PDU_M_Get.F ;

           {******* Parametros de A_Release :
                     ACN: Nome da Aplicacao SGH
                     APT,AEQ: ManagerInit
                     API: Nome local
                     AEI: Nome gerente chamado }

                PDU_A_Release.ACN := 'SGH' ;
                PDU_A_Release.APT := 'ManagerInit' ;
                PDU_A_Release.AEQ := 'ManagerInit' ;
                PDU_A_Release.API := LocalName ;
                PDU_A_Release.AEI := Destiny ;
                output MCEP.A_ReleaseReq(PDU_A_Release);
                SDU_ManagerInit.Status := 0;
             output UCEP.ManagerInitConf(SDU_ManagerInit);
           end;

      provided   ( PDU_M_Get.E = 0 ) and
                 (( PDU_M_Get.II <> 'ManagerInit' ) or
```

```
            ( PDU_M_Get.MOC <> 'TablePolicy' ))
      to REPOUSO              { Erros do SGH }
           name ErroSGHManagerInit3:
           begin
              SDU_ManagerInit.Status := 27 ;
            output UCEP.ManagerInitConf(SDU_ManagerInit) ;
           end;

      provided        ( PDU_M_Get.E <> 0 )
      to REPOUSO              { Erros do CMISE }
           name ErroCMISEManagerInit3:
           begin
              SDU_ManagerInit.Status := PDU_M_Get.E ;
            output UCEP.ManagerInitConf(SDU_ManagerInit) ;
           end;

(*************** ManagerInit - Respondedor ******************)

   from REPOUSO                  {recebe ManagerInit }
      when MCEP.A_AssociateInd(PDU_A_Associate)

           {******* Parametros de A_Associate :
                      ACN: Nome da Aplicacao SGH
                      APT,AEQ: ManagerInit
                      API: Nome local
                      AEI: Nome gerente chamado
                      UD: Controle de Acesso -autorizacao}

      provided  ( PDU_A_Associate.ACN = 'SGH' ) and
                 ( PDU_A_Associate.APT = 'ManagerInit') and
                 ( PDU_A_Associate.AEQ = 'ManagerInit') and
                 ( PDU_A_Associate.UD = Autorization ) and
                 ( PDU_A_Associate.AEI = LocalName )

      to ENV_TABM1
           name EstabeleceConexaoManagerInit:
           begin
                Agent := PDU_A_Associate.API ;
                PDU_A_Associate.RAPT := 'ManagerInit' ;
                PDU_A_Associate.RAEQ := 'ManagerInit' ;
                PDU_A_Associate.R := 0 ;
             output MCEP.A_AssociateResp(PDU_A_Associate);
           end;

      provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                 ( PDU_A_Associate.APT = 'ManagerInit') and
                 ( PDU_A_Associate.AEQ = 'ManagerInit') and
                 (( PDU_A_Associate.UD <> Autorization ) or
                 ( PDU_A_Associate.AEI <> LocalName ))

      to REPOUSO              { Erros do SGH }
           name ErroSGHManagerInit4:
           begin
              PDU_A_Associate.R := 28 ;
             output MCEP.A_AssociateResp(PDU_A_Associate);
           end;

   from ENV_TABM1

      when MCEP.M_GetInd(PDU_M_Get)

           {****** Parametros de M-Get para pedir TableM
                  II : Nome da primitiva
```

```
                BOC: Tabela solicitada
                AC: Controle de acesso - autorizacao  }

     provided   ( PDU_M_Get.II = 'ManagerInit' ) and
                ( PDU_M_Get.BOC = 'TableM' ) and
                ( PDU_M_Get.AC = Autorization )


     to ENV_TABPOL1
          name  EnviaTableMManagerInit:

          begin

          {****** Parametros de M-Get
                    II: Nome da primitiva
                    MOC: Nome da Tabela enviada
                    AL: Tabela enviada
                    E: Erro         }

               PDU_M_Get.MOC := 'TableM' ;
               for i:=2 to TamTableM do
                    PDU_M_Get.AL[i] := TableM[i-1];
               PDU_M_Get.AL[1] := LocalName ;
               PDU_M_Get.F := ManagerPointer + 1 ;
               PDU_M_Get.E := 0 ;
               output MCEP.M_GetResp(PDU_M_Get);
          end;

     provided   ( PDU_M_Get.II = 'ManagerInit' ) and
                ( PDU_M_Get.AC <> Autorization )
     to REPOUSO            { Erros de seguranca do SGH }
          name ErroSGHManagerInit5:
          begin
               PDU_M_Get.E := 28 ;
               output MCEP.M_GetResp(PDU_M_Get);
          end;

     provided   ( PDU_M_Get.II = 'ManagerInit' ) and
                ( PDU_M_Get.BOC <> 'TableM' )
     to REPOUSO             { Erros de tabelas do SGH }
          name ErroSGHManagerInit6:
          begin
               PDU_M_Get.E := 27 ;
               output MCEP.M_GetResp(PDU_M_Get);
          end;

from ENV_TABPOL1

     when MCEP.M_GetInd(PDU_M_Get)

     {****** Parametros de M-Get para pedir TablePolicy
               II : Nome da primitiva
               BOC: Tabela solicitada
               AC: Controle de acesso - autorizacao    }

     provided   ( PDU_M_Get.II = 'ManagerInit' ) and
                ( PDU_M_Get.BOC = 'TablePolicy' ) and
                ( PDU_M_Get.AC = Autorization )


     to ESP_DESC1
          name  EnviaTablePolicyManagerInit:
```

```
          begin

          {****** Parametros de M-Get
                    II: Nome da primitiva
                    MOC: Nome da Tabela enviada
                    AL: Tabela enviada
                    E: Erro          }

               PDU_M_Get.MOC := 'TablePolicy' ;
               for i:=1 to TamTablePolicy do
                    PDU_M_Get.AL[i] := TablePolicy[i];
               PDU_M_Get.F := PolicyPointer ;
               PDU_M_Get.E := 0 ;
               output MCEP.M_GetResp(PDU_M_Get);
          end;

     provided   ( PDU_M_Get.II = 'ManagerInit' ) and
                ( PDU_M_Get.AC <> Autorization )

     to REPOUSO             { Erros de seguranca do SGH }
          name ErroSGHManagerInit7:
          begin
               PDU_M_Get.E := 28 ;
               output MCEP.M_GetResp(PDU_M_Get);
          end;

     provided   ( PDU_M_Get.II = 'ManagerInit' ) and
                ( PDU_M_Get.BOC <> 'TablePolicy' )

     to REPOUSO              { Erros de tabelas do SGH }
          name ErroSGHManagerInit8:
          begin
               PDU_M_Get.E := 27 ;
               output MCEP.M_GetResp(PDU_M_Get);
          end;

from ESP_DESC1

     when MCEP.A_ReleaseInd(PDU_A_Release)

     to REPOUSO
          name  DesconectaManagerInit:
          begin
             AgentPointer := AgentPointer + 1 ;
             TableA[AgentPointer] := PDU_A_Release.API ;
            SDU_ManagerInit.Agent := PDU_A_Release.API;
             SDU_ManagerInit.Status := 0 ;
            output UCEP.ManagerInitInd(SDU_ManagerInit) ;
          end;

(************ ManagerRead - Iniciador/Respondedor ************)

from REPOUSO
     when UCEP.ManagerReadReq(SDU_ManagerRead)

     to REPOUSO
          name  EnviaTableMManagerRead:
          begin
               for i:=1 to TamTableM do
                 SDU_ManagerRead.List[i] := TableM[i];
               SDU_ManagerRead.Pointer:=ManagerPointer;
               SDU_ManagerRead.Status:=0;
           output UCEP.ManagerReadConf(SDU_ManagerRead);
```

```
          end;

(************* AgentRead - Iniciador/Respondedor *************)

     from REPOUSO
          when UCEP.AgentReadReq(SDU_AgentRead)

          to REPOUSO
               name  EnviaTableAAgentRead:
               begin
                    for i:=1 to TamTableA do
                         SDU_AgentRead.List[i] := TableA[i];
                    SDU_AgentRead.Pointer:=AgentPointer;
                    SDU_AgentRead.Status:=0;
                    output UCEP.AgentReadConf(SDU_AgentRead);
               end;


(****************** AgentCreate-Iniciador ******************)

     from  REPOUSO   {usuario solicita criacao de agente}

     when UCEP.AgentCreateReq(SDU_AgentCreate)

          to ESP_CONEC2
               name PedeAgentCreate:
               begin
                    Agent := SDU_AgentCreate.Agent ;
                    Destiny := SDU_AgentCreate.Agent ;

               {******* Parametros de A_Associate :
                         ACN: Nome da Aplicacao SGH
                         APT,AEQ: AgentCreate
                         API: Nome local
                         AEI: Nome agente chamado
                         U: Controle de Acesso - autorizacao}

                    PDU_A_Associate.ACN := 'SGH' ;
                    PDU_A_Associate.APT := 'AgentCreate' ;
                    PDU_A_Associate.AEQ := 'AgentCreate' ;
                    PDU_A_Associate.API := LocalName ;
                    PDU_A_Associate.AEI := Destiny ;
                    PDU_A_Associate.UD := Autorization ;
                 output MCEP.A_AssociateReq(PDU_A_Associate);
               end;

     from ESP_CONEC2

          when MCEP.A_AssociateConf(PDU_A_Associate)

          provided     ( PDU_A_Associate.R = 0 ) and
                      ( PDU_A_Associate.RAPT = 'AgentCreate' )

          to ENV_TABA2
               name  EnviaAgentCreateTableA:

               {****** Parametros de M-Set
               {       II : Nome da primitiva
               {       BOC: Agent Create
               {      AC: Controle de acesso - autorizacao  }
               begin
                    PDU_M_Set.II := 'AgentCreate' ;
```

```
                PDU_M_Set.BOC:= 'TableM' ;
                PDU_M_Set.M:= 'Confirmed' ;
                PDU_M_Set.MI:=LocalName ;
                PDU_M_Set.SC := Destiny ;
                PDU_M_Set.SY := LocalName ;
                PDU_M_Set.AC:= Autorization ;
                output MCEP.M_SetReq(PDU_M_Set);
           end;

      provided   ( PDU_A_Associate.R = 0 ) and
                 ( PDU_A_Associate.APT <> 'AgentCreate' )

      to REPOUSO              { Erros do SGH }
           name ErroSGHAgentCreate1:
           begin
              SDU_AgentCreate.Status := 27 ;
            output UCEP.AgentCreateConf(SDU_AgentCreate) ;
           end;

      provided    ( PDU_A_Associate.R <> 0 )

      to REPOUSO              { Erros do CMISE }
           name ErroCMISEAgentCreate1:
           begin
             SDU_AgentCreate.Status := PDU_A_Associate.R ;
            output UCEP.AgentCreateConf(SDU_AgentCreate) ;
           end;

 from ENV_TABA2

      when MCEP.M_SetConf(PDU_M_Set)

           {****** Parametros de M-Set
                     II: Nome da primitiva
                     MOC: Nome da Agente criado
                     E: Erro         }

      provided   ( PDU_M_Set.E = 0 ) and
                 ( PDU_M_Set.II= 'AgentCreate' ) and
                 ( PDU_M_Set.MOC= 'TableM' )

      to ENV_TABPOL2
           name  EnviaTablePolicyAgentCreate:

           begin

      {****** Parametros de M-Get para enviar TablePolicy
                II : Nome da primitiva
                BOC: Tabela Enviada
                AC: Controle de acesso - autorizacao    }

                PDU_M_Set.BOC := 'TablePolicy' ;
                PDU_M_Set.MI := TablePolicy[1] ;
                PDU_M_Set.AC := Autorization ;
                PDU_M_Set.SC := Destiny ;
                PDU_M_Set.SY := LocalName ;
                output MCEP.M_SetReq(PDU_M_Set);
           end;

      provided   ( PDU_M_Set.E = 0 ) and
                 (( PDU_M_Set.II <> 'AgentCreate' ) or
                 ( PDU_M_Set.MOC <> 'TableM' ))
```

```
     to REPOUSO              { Erros do SGH }
          name ErroSGHAgentCreate2:
          begin
             SDU_AgentCreate.Status := 27 ;
           output UCEP.AgentCreateConf(SDU_AgentCreate) ;
          end;

     provided        ( PDU_M_Set.E <> 0 )

     to REPOUSO              { Erros do CMISE }
          name ErroCMISEAgentCreate2:
          begin
            SDU_AgentCreate.Status := PDU_M_Set.E ;
           output UCEP.AgentCreateConf(SDU_AgentCreate) ;
          end;

from ENV_TABPOL2

     when MCEP.M_SetConf(PDU_M_Set)

          {****** Parametros de M-Set
                    II: Nome da primitiva
                    MOC: Nome da Tabela recebida
                    E: Erro         }

     provided   ( PDU_M_Set.E = 0 ) and
                ( PDU_M_Set.II= 'AgentCreate' ) and
                ( PDU_M_Set.MOC= 'TablePolicy' )

     to REPOUSO
          name  DesconectaAgentCreate:

          begin

          {******* Parametros de A_Release :
                    ACN: Nome da Aplicacao SGH
                    APT,AEQ: ManagerInit
                    API: Nome local
                    AEI: Nome gerente chamado }

               PDU_A_Release.ACN := 'SGH' ;
               PDU_A_Release.APT := 'AgentCreate' ;
               PDU_A_Release.AEQ := 'AgentCreate' ;
               PDU_A_Release.API := LocalName ;
               PDU_A_Release.AEI := Destiny ;
               output MCEP.A_ReleaseReq(PDU_A_Release);
               AgentPointer := AgentPointer+1 ;
               TableA[AgentPointer] := Destiny ;
               SDU_AgentCreate.Status := 0;
            output UCEP.AgentCreateConf(SDU_AgentCreate);
          end;

     provided   ( PDU_M_Set.E = 0 ) and
                ( PDU_M_Set.II <> 'AgentCreate' )

     to REPOUSO              { Erros do SGH }
          name ErroSGHAgentCreate3:
          begin
             SDU_AgentCreate.Status := 27 ;
            output UCEP.AgentCreateConf(SDU_AgentCreate);
          end;

     provided        ( PDU_M_Set.E <> 0 )
```

```
        to REPOUSO              { Erros do CMISE }
             name ErroCMISEAgentCreate3:
             begin
                SDU_AgentCreate.Status := PDU_M_Set.E ;
               output UCEP.AgentCreateConf(SDU_AgentCreate);
             end;

(**************** AgentCreate-Respondedor *****************)

     from REPOUSO           {recebe AgentCreate }
        when MCEP.A_AssociateInd(PDU_A_Associate)

             {******* Parametros de A_Associate :
                        ACN: Nome da Aplicacao SGH
                        APT,AEQ: AgentCreate
                        API: Nome gerente chamador
                        AEI: Nome local
                        UD:Controle de Acesso - autorizacao}

        provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                   ( PDU_A_Associate.APT = 'AgentCreate') and
                   ( PDU_A_Associate.AEQ = 'AgentCreate') and
                   ( PDU_A_Associate.UD = Autorization )

        to REC_TABA2
             name EstabeleceConexaoAgentCreate:
             begin
                  LocalName := PDU_A_Associate.AEI ;
                  PDU_A_Associate.RAPT := 'AgentCreate' ;
                  PDU_A_Associate.RAEQ := 'AgentCreate' ;
                  PDU_A_Associate.R := 0 ;
               output MCEP.A_AssociateResp(PDU_A_Associate);
             end;

        provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                   ( PDU_A_Associate.APT = 'AgentCreate' )and
                   ( PDU_A_Associate.AEQ = 'AgentCreate') and
                   ( PDU_A_Associate.UD <> Autorization )

        to REPOUSO              { Erros do SGH }
             name ErroSGHAgentCreate4:
             begin
                  PDU_A_Associate.R := 28 ;
               output MCEP.A_AssociateResp(PDU_A_Associate);
             end;

     from REC_TABA2

        when MCEP.M_SetInd(PDU_M_Set)

        {****** Parametros de M-Set para pedir TableM
                  II : Nome da primitiva
                  BOC: Tabela enviada
                  AC: Controle de acesso - autorizacao    }

        provided   ( PDU_M_Set.II = 'AgentCreate' ) and
                   ( PDU_M_Set.BOC = 'TableM' ) and
                   ( PDU_M_Set.AC = Autorization )

        to REC_TABPOL2
             name  RecebeTableMAgentCreate:
```

```
            begin

            {****** Parametros de M-Set
                         II: Nome da primitiva
                         MOC: Nome da Tabela recebida
                         E: Erro          }

                  PDU_M_Set.MOC := 'TableM' ;
                  for i:=0 to (TamTableM-2) do
            TableM[TamTableM-i]:=TableM[TamTableM (i+1)];
                  TableM[1] := PDU_M_Set.MI ;
                  ManagerPointer := ManagerPointer + 1 ;
                  PDU_M_Set.E := 0 ;
                  output MCEP.M_SetResp(PDU_M_Set);
            end;

      provided   ( PDU_M_Set.II <> 'AgentCreate' ) or
                 ( PDU_M_Set.AC <> Autorization )

      to REPOUSO              { Erros de seguranca do SGH}
            name ErroSGHAgentCreate5:
            begin
                  PDU_M_Set.E := 28 ;
                  output MCEP.M_SetResp(PDU_M_Set);
            end;

      provided   ( PDU_M_Set.II = 'AgentCreate' ) and
                 ( PDU_M_Set.BOC <> 'TableM' )

      to REPOUSO              { Erros de tabelas do SGH }
            name ErroSGHManagerInit6:
            begin
                  PDU_M_Set.E := 27 ;
                  output MCEP.M_SetResp(PDU_M_Set);
            end;

from REC_TABPOL2

      when MCEP.M_SetInd(PDU_M_Set)

      {****** Parametros de M-Set para receber TablePolicy
                  II : Nome da primitiva
                  BOC: Tabela enviada
                  AC: Controle de acesso - autorizacao    }

      provided   ( PDU_M_Set.II = 'AgentCreate' ) and
                 ( PDU_M_Set.BOC = 'TablePolicy' ) and
                 ( PDU_M_Set.AC = Autorization )

      to ESP_DESC2
            name  RecebeTablePolicyAgentCreate:

            begin

            {****** Parametros de M-Set
                         II: Nome da primitiva
                         MOC: Nome da Tabela enviada
                         E: Erro          }

                  PDU_M_Set.MOC := 'TablePolicy' ;
                  TablePolicy[1] := PDU_M_Set.MI;
                  PolicyPointer := 1 ;
                  PDU_M_Set.E := 0 ;
```

```
                output MCEP.M_SetResp(PDU_M_Set);
            end;

        provided   ( PDU_M_Set.II <> 'AgentCreate' ) or
                   ( PDU_M_Set.AC <> Autorization )

        to REPOUSO              { Erros de seguranca do SGH}
            name ErroSGHAgentCreate7:
            begin
                PDU_M_Set.E := 28 ;
                output MCEP.M_SetResp(PDU_M_Set);
            end;

        provided   ( PDU_M_Set.II = 'AgentCreate' ) and
                   ( PDU_M_Set.BOC <> 'TablePolicy' )

        to REPOUSO           { Erros de tabelas do SGH }
            name ErroSGHAgentCreate8:
            begin
                PDU_M_Set.E := 27 ;
                output MCEP.M_SetResp(PDU_M_Set);
            end;

    from ESP_DESC2

        when MCEP.A_ReleaseInd(PDU_A_Release)

            {******* Parametros de A_Release :
                    ACN: Nome da Aplicacao SGH
                    APT,AEQ: ManagerInit
                    API: Nome gerente chamador
                    AEI: Nome gerente chamado }

        to REPOUSO
            name  DesconectaAgentCreate:
            begin
              SDU_AgentCreate.Manager:=
    PDU_A_Release.API;
              SDU_AgentCreate.Status := 0 ;
              output UCEP.AgentCreateInd(SDU_AgentCreate) ;
            end;

(***************** AgentElimin Iniciador *******************)

    from REPOUSO          {usuario solicita eliminar agente}
        when UCEP.AgentEliminReq(SDU_AgentElimin)

            {******* Parametros de A_Associate :
                    ACN: Nome da Aplicacao SGH
                    APT,AEQ: AgentElimin
                    API: Nome local
                    AEI: Nome agente chamado
                    UD: Controle de Acesso- autorizacao}

        to ESP_CONEC3
            name ConexaoAgentElimin:
            begin
                Agent := SDU_AgentElimin.Agent ;
                Destiny := Agent ;
                PDU_A_Associate.ACN := 'SGH' ;
                PDU_A_Associate.APT := 'AgentElimin' ;
                PDU_A_Associate.AEQ := 'AgentElimin' ;
                PDU_A_Associate.API := LocalName ;
```

```
                PDU_A_Associate.AEI := Destiny ;
                PDU_A_Associate.UD := Autorization ;
           output MCEP.A_AssociateReq(PDU_A_Associate);
           end;


from ESP_CONEC3

      when MCEP.A_AssociateConf(PDU_A_Associate)

      provided    ( PDU_A_Associate.R = 0 ) and
                  ( PDU_A_Associate.RAPT = 'AgentElimin' )
      to ENV_AELIM
           name  EnviaAElimin:

           {****** Parametros de M-Set
           {       II : Nome da primitiva
           {       M : Modo Confirmado
           {       BOC: Tabela solicitada
           {       AC: Controle de acesso - autorizacao  }
           begin
                PDU_M_Set.II := 'AgentElimin' ;
                PDU_M_Set.M := 'Confirmed' ;
                PDU_M_Set.BOC:= 'TableM' ;
                PDU_M_Set.SC := Destiny ;
                PDU_M_Set.SY := LocalName ;
                PDU_M_Set.AC:= Autorization ;
                output MCEP.M_SetReq(PDU_M_Set);
           end;

      provided   ( PDU_A_Associate.R = 0 ) and
                 ( PDU_A_Associate.APT <> 'AgentElimin' )
      to REPOUSO              { Erros do SGH }
           name ErroSGHAgentElimin1:
           begin
              SDU_AgentElimin.Status := 27 ;
             output UCEP.AgentEliminConf(SDU_AgentElimin);
           end;

      provided        ( PDU_A_Associate.R <> 0 )
      to REPOUSO              { Erros do CMISE }
           name ErroCMISEAgentElimin1:
           begin
             SDU_AgentElimin.Status := PDU_A_Associate.R ;
            output UCEP.AgentEliminConf(SDU_AgentElimin) ;
           end;

from ENV_AELIM

      when MCEP.M_SetConf(PDU_M_Set)

           {****** Parametros de M-Set
                     II: Nome da primitiva
                     MOC: Nome da Tabela enviada
                     AL: Tabela enviada
                     E: Erro         }

      provided   ( PDU_M_Set.E = 0 ) and
                 ( PDU_M_Set.II= 'AgentElimin' ) and
                 ( PDU_M_Set.MOC= 'TableM' )
      to REPOUSO
           name  DesconectaAgentElimin:
```

```
          begin
               SDU_AgentElimin.Status:= 31;
               for i := 1 to TamTableA do
               begin
                    if (TableA[i] = Agent) then
                    begin
                         SDU_AgentElimin.Status:=0;
                         AgentPointer :=AgentPointer-1;
                         for j := i to TamTableA-1 do
                         begin
                             TableA[j] := TableA[j+1];
                         end;
                    end;
               end;

          {******* Parametros de A_Release :
                    ACN: Nome da Aplicacao SGH
                    APT,AEQ: AgentElimin
                    API: Nome local
                    AEI: Nome agente chamado }

               PDU_A_Release.ACN := 'SGH' ;
               PDU_A_Release.APT := 'AgentElimin' ;
               PDU_A_Release.AEQ := 'AgentElimin' ;
               PDU_A_Release.API := LocalName ;
               PDU_A_Release.AEI := Destiny ;
               output MCEP.A_ReleaseReq(PDU_A_Release);
            output UCEP.AgentEliminConf(SDU_AgentElimin);
          end;

     provi      ( PDU_M_Set.E = 0 ) and
                (( PDU_M_Set.II <> 'AgentElimin' ) or
                ( PDU_M_Set.MOC <> 'TableM' ))
     to REPOUSO              { Erros do SGH }
          name ErroSGHAgentElimin3:
          begin
             SDU_AgentElimin.Status := 27 ;
            output UCEP.AgentEliminConf(SDU_AgentElimin);
          end;

     provided        ( PDU_M_Set.E <> 0 )
     to REPOUSO              { Erros do CMISE }
          name ErroCMISEAgentElimin3:
          begin
             SDU_AgentElimin.Status := PDU_M_Set.E ;
            output UCEP.AgentEliminConf(SDU_AgentElimin);
          end;

(*************** AgentELimin Respondedor ******************)

     from REPOUSO
          when MCEP.A_AssociateInd(PDU_A_Associate)

               {******* Parametros de A_Associate :
                         ACN: Nome da Aplicacao SGH
                         APT,AEQ: AgentElimin
                         API: Nome local
                         AEI: Nome agente chamado
                         UD: Controle de Acesso -autorizacao}

          provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                     ( PDU_A_Associate.APT = 'AgentElimin' )and
                     ( PDU_A_Associate.AEQ = 'AgentElimin' )and
```

```
            ( PDU_A_Associate.UD = Autorization ) and
            ( PDU_A_Associate.AEI = LocalName )

to ESP_AELIM
     name EstabeleceConexaoAgentElimin:
     begin
          Manager := PDU_A_Associate.API ;
          PDU_A_Associate.RAPT := 'AgentElimin' ;
          PDU_A_Associate.RAEQ := 'AgentElimin' ;
          PDU_A_Associate.R := 0 ;
       output MCEP.A_AssociateResp(PDU_A_Associate);
     end;

provided    ( PDU_A_Associate.ACN = 'SGH' ) and
            ( PDU_A_Associate.APT = 'AgentElimin' )and
            ( PDU_A_Associate.AEQ = 'AgentElimin' )and
            (( PDU_A_Associate.UD <> Autorization ) or
            ( PDU_A_Associate.AEI <> LocalName ))

to REPOUSO              { Erros do SGH }
     name ErroSGHAgentElimin4:
     begin
          PDU_A_Associate.R := 28 ;
       output MCEP.A_AssociateResp(PDU_A_Associate);
     end;

from ESP_AELIM

     when MCEP.M_SetInd(PDU_M_Set)

     {****** Parametros de M-Set para eliminar Agente
               II : Nome da primitiva
               BOC: Tabela modificada
               AC: Controle de acesso - autorizacao    }

provided    ( PDU_M_Set.II = 'AgentElimin' ) and
            ( PDU_M_Set.BOC = 'TableM' ) and
            ( PDU_M_Set.AC = Autorization )
to ESP_DESC3
     name  ApagaTableMAgentElimin:
     begin
     {****** Parametros de M-Set
                    II: Nome da primitiva
                    MOC: Nome da Tabela enviada
                    AL: Tabela alterada
                    E: Erro         }
               PDU_M_Set.MOC := 'TableM' ;
               PDU_M_Set.E := 31 ;
               if TableM[1] = PDU_M_Set.SY then
               begin
                    for i:=1 to (TamTableM - 1) do
                         TableM[i] := TableM[i+1];
                    TableM[TamTableM] := 0 ;
                    ManagerPointer := ManagerPointer-1;
                    PDU_M_Set.E := 0 ;
               end ;
               output MCEP.M_SetResp(PDU_M_Set);
     end;

provided    ( PDU_M_Set.II = 'AgentElimin' ) and
            ( PDU_M_Set.AC <> Autorization )
to REPOUSO              { Erros de seguranca do SGH}
     name ErroSGHAgentElimin7:
```

```
            begin
                 PDU_M_Set.E := 28 ;
                 output MCEP.M_SetResp(PDU_M_Set);
            end;

       provided   ( PDU_M_Set.II <> 'AgentElimin' ) or
                  ( PDU_M_Set.BOC <> 'TablePolicy' )
       to REPOUSO            { Erros de tabelas do SGH }
            name ErroSGHAgentElimin8:
            begin
                 PDU_M_Set.E := 27 ;
                 output MCEP.M_SetResp(PDU_M_Set);
            end;

  from ESP_DESC3

       when MCEP.A_ReleaseInd(PDU_A_Release)

       to REPOUSO
            name  DesconectaAgentElimin:
            begin
              SDU_AgentElimin.Manager:=
  PDU_A_Release.API;
              SDU_AgentElimin.Status := 0 ;
              output UCEP.AgentEliminInd(SDU_AgentElimin) ;
            end;


(******************* DelegAct Iniciador ********************)

  from REPOUSO               {usuario solicita delegacao }
       when UCEP.DelegActReq(SDU_DelegAct)

            {******* Parametros de A_Associate :
                      ACN: Nome da Aplicacao SGH
                      APT,AEQ: DelegAct
                      API: Nome local
                      AEI: Nome gerente chamado
                      U: Controle de Acesso - autorizacao}

       to ESP_CONECM
            name ConexaoDelegAct:
            begin
                 Manager := SDU_DelegAct.Manager ;
                 Agent := SDU_DelegAct.Agent ;
                 Destiny := SDU_DelegAct.Manager ;
                 PDU_A_Associate.ACN := 'SGH' ;
                 PDU_A_Associate.APT := 'DelegAct' ;
                 PDU_A_Associate.AEQ := 'DelegAct' ;
                 PDU_A_Associate.API := LocalName ;
                 PDU_A_Associate.AEI := Destiny ;
                 PDU_A_Associate.UD := Autorization ;
            output MCEP.A_AssociateReq(PDU_A_Associate);
            end;

  from ESP_CONECM

       when MCEP.A_AssociateConf(PDU_A_Associate)

       provided    ( PDU_A_Associate.R = 0 ) and
                   ( PDU_A_Associate.RAPT = 'DelegAct' )
       to ESP_ACTIONM
            name  SolicitaDelegacao:
```

```
        {****** Parametros de M-Action
        {       II : Nome da primitiva
        {       M : Modo Confirmado
        {       BOC: Tabela objeto
        {       AC: Controle de acesso - autorizacao
        {       AT : Tipo de acao ( ask ou execute )
        {      A : Argumentos de acao ( nome do agente}
        {       E: Erro          }
        begin
             PDU_M_Action.II := 'DelegAct' ;
             PDU_M_Action.M := 'Confirmed' ;
             PDU_M_Action.BOC:= 'TableA' ;
             PDU_M_Action.AT:= 'Ask' ;
             PDU_M_Action.AA:= Agent ;
             PDU_M_Action.SC := Destiny ;
             PDU_M_Action.SY := LocalName ;
             PDU_M_Action.AC:= Autorization ;
             output MCEP.M_ActionReq(PDU_M_Action);
        end;

   provided   ( PDU_A_Associate.R = 0 ) and
              ( PDU_A_Associate.APT <> 'DelegAct' )
   to REPOUSO              { Erros do SGH }
        name ErroSGHDelegAct1:
        begin
             SDU_DelegAct.Status := 27 ;
             output UCEP.DelegActConf(SDU_DelegAct) ;
        end;

   provided        ( PDU_A_Associate.R <> 0 )
   to REPOUSO              { Erros do CMISE }
        name ErroCMISEDelegAct1:
        begin
             SDU_DelegAct.Status := PDU_A_Associate.R ;
             output UCEP.DelegActConf(SDU_DelegAct) ;
        end;

from ESP_ACTIONM

   when MCEP.M_ActionConf(PDU_M_Action)

        {****** Parametros de M-Action
        {       II : Nome da primitiva
        {       M : Modo Confirmado
        {       BOC: Tabela objeto
        {       AC: Controle de acesso - autorizacao
        {       AT : Tipo de acao ( ask ou execute )
        {     AA : Argumentos de acao ( nome do agente}
        {       E: Erro          }

   provided   ( PDU_M_Action.E = 0 ) and
              ( PDU_M_Action.II = 'DelegAct' ) and
              ( PDU_M_Action.AT = 'Ask' ) and
              ( PDU_M_Action.MOC = 'TableA' )


   to MUDA_AGENT
        name  DesconectaDelegActM:

        begin
        {******* Parametros de A_Release :
                  ACN: Nome da Aplicacao SGH
                  APT,AEQ: DelegAct
```

```
                  API: Nome local
                  AEI: Nome gerente chamado }

             PDU_A_Release.ACN := 'SGH' ;
             PDU_A_Release.APT := 'DelegAct' ;
             PDU_A_Release.AEQ := 'DelegAct' ;
             PDU_A_Release.API := LocalName ;
             PDU_A_Release.AEI := Destiny ;
             output MCEP.A_ReleaseReq(PDU_A_Release);
        end;


   provided      ( PDU_M_Action.E = 0 ) and
                 ( PDU_M_Action.II <> 'DelegAct' )

   to REPOUSO             { Erros do SGH }
        name ErroSGHDelegAct4:
        begin
             SDU_DelegAct.Status := 27 ;
             output UCEP.DelegActConf(SDU_DelegAct) ;
        end;

   provided        ( PDU_M_Action.E <> 0 )

   to REPOUSO   { Erros do CMISE ou nao aceitou deleg }
        name ErroCMISEDelegAct3:
        begin
             SDU_DelegAct.Status := PDU_M_Action.E ;
             output UCEP.DelegActConf(SDU_DelegAct) ;
        end;

from MUDA_AGENT

        {******* Parametros de A_Associate :
                  ACN: Nome da Aplicacao SGH
                  APT,AEQ: DelegAct
                  API: Nome local
                  AEI: Nome gerente chamado
                  U: Controle de Acesso - autorizacao}
   to ESP_CONECA
        name ConexaoDelegActA:
        begin
             Manager := SDU_DelegAct.Manager ;
             Agent := SDU_DelegAct.Agent ;
             Destiny := Agent ;
             PDU_A_Associate.ACN := 'SGH' ;
             PDU_A_Associate.APT := 'DelegAct' ;
             PDU_A_Associate.AEQ := 'DelegAct' ;
             PDU_A_Associate.API := LocalName ;
             PDU_A_Associate.AEI := Destiny ;
             PDU_A_Associate.UD := Autorization ;
        output MCEP.A_AssociateReq(PDU_A_Associate);
        end;

from ESP_CONECA

   when MCEP.A_AssociateConf(PDU_A_Associate)

   provided    ( PDU_A_Associate.R = 0 ) and
               ( PDU_A_Associate.RAPT = 'DelegAct' )

   to ESP_ACTIONA
        name  SolicitaDelegacao:
```

```
        {****** Parametros de M-Action
        {       II : Nome da primitiva
        {       M : Modo Confirmado
        {       BOC: Tabela objeto
        {       AC: Controle de acesso - autorizacao
        {      AT : Tipo de acao ( ask ou execute )
        {      AA :Argumentos de acao ( nome do agente}
        {      E: Erro         }
        begin
             Manager := SDU_DelegAct.Manager ;
             PDU_M_Action.II := 'DelegAct' ;
             PDU_M_Action.M := 'Confirmed' ;
             PDU_M_Action.BOC:= 'TableM' ;
             PDU_M_Action.AT:= 'Act' ;
             PDU_M_Action.AA:= Manager ;
             PDU_M_Action.SC := Destiny ;
             PDU_M_Action.SY := LocalName ;
             PDU_M_Action.AC:= Autorization ;
             output MCEP.M_ActionReq(PDU_M_Action);
        end;

   provided   ( PDU_A_Associate.R = 0 ) and
              ( PDU_A_Associate.APT <> 'DelegAct' )

   to REPOUSO              { Erros do SGH }
        name ErroSGHDelegAct1:
        begin
             SDU_DelegAct.Status := 27 ;
             output UCEP.DelegActConf(SDU_DelegAct) ;
        end;

   provided        ( PDU_A_Associate.R <> 0 )

   to REPOUSO              { Erros do CMISE }
        name ErroCMISEDelegAct1:
        begin
             SDU_DelegAct.Status := PDU_A_Associate.R ;
             output UCEP.DelegActConf(SDU_DelegAct) ;
        end;

from ESP_ACTIONA

   when MCEP.M_ActionConf(PDU_M_Action)

        {****** Parametros de M-Action
        {       II : Nome da primitiva
        {       M : Modo Confirmado
        {       BOC: Tabela objeto
        {       AC: Controle de acesso - autorizacao
        {       AT : Tipo de acao ( ask ou execute )
        {      AA :Argumentos de acao ( nome do agente}
        {       E: Erro         }

   provided   ( PDU_M_Action.E = 0 ) and
              ( PDU_M_Action.II = 'DelegAct' ) and
              ( PDU_M_Action.AT = 'Act' ) and
              ( PDU_M_Action.MOC = 'TableM' )
   to REPOUSO
        name  DesconectaDelegActA:

        begin
        {******* Parametros de A_Release :
```

```
                    ACN: Nome da Aplicacao SGH
                    APT,AEQ: DelegAct
                    API: Nome local
                    AEI: Nome gerente chamado }

               Agent := SDU_DelegAct.Agent ;
               Destiny := Agent ;
               PDU_A_Release.ACN := 'SGH' ;
               PDU_A_Release.APT := 'DelegAct' ;
               PDU_A_Release.AEQ := 'DelegAct' ;
               PDU_A_Release.API := LocalName ;
               PDU_A_Release.AEI := Destiny ;
               output MCEP.A_ReleaseReq(PDU_A_Release);
               SDU_DelegAct.Status := 0 ;
               output UCEP.DelegActConf(SDU_DelegAct) ;
          end;

     provided      ( PDU_M_Action.E = 0 ) and
                   ( PDU_M_Action.II <> 'DelegAct' )
     to REPOUSO              { Erros do SGH }
          name ErroSGHDelegAct4:
          begin
               SDU_DelegAct.Status := 27 ;
               output UCEP.DelegActConf(SDU_DelegAct) ;
          end;

     provided        ( PDU_M_Action.E <> 0 )
     to REPOUSO    { Erros do CMISE ou nao aceitou deleg}
          name ErroCMISEDelegAct3:
          begin
               SDU_DelegAct.Status := PDU_M_Action.E ;
               output UCEP.DelegActConf(SDU_DelegAct) ;
          end;

     provided  otherwise
     to REPOUSO              { Erros nao definido }
          name ErroND4DelegAct:
          begin
               SDU_DelegAct.Status := 99 ;
               output UCEP.DelegActConf(SDU_DelegAct) ;
          end;

(********************* DelegAct Respondedor *****************)

     from REPOUSO
          when MCEP.A_AssociateInd(PDU_A_Associate)

               {******* Parametros de A_Associate :
                    ACN: Nome da Aplicacao SGH
                    APT,AEQ: DelegAct
                    API: Nome local
                    AEI: Nome gerente chamado
                    UD:Controle de Acesso - autorizacao}

     provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                ( PDU_A_Associate.APT = 'DelegAct' ) and
                ( PDU_A_Associate.AEQ = 'DelegAct' ) and
                ( PDU_A_Associate.UD = Autorization ) and
                ( PDU_A_Associate.AEI = LocalName )

     to ESP_ACTION
          name EstabeleceConexaoDelegAct:
          begin
```

```
                Agent := PDU_A_Associate.API ;
                PDU_A_Associate.RAPT := 'DelegAct' ;
                PDU_A_Associate.RAEQ := 'DelegAct' ;
                PDU_A_Associate.R := 0 ;
              output MCEP.A_AssociateResp(PDU_A_Associate);
            end;

       provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                  ( PDU_A_Associate.APT = 'DelegAct' ) and
                  ( PDU_A_Associate.AEQ = 'DelegAct' ) and
                  (( PDU_A_Associate.UD <> Autorization ) or
                  ( PDU_A_Associate.AEI <> LocalName ))

       to REPOUSO             { Erros do SGH }
            name ErroSGHDelegAct4:
            begin
               PDU_A_Associate.R := 28 ;
               output MCEP.A_AssociateResp(PDU_A_Associate);
            end;

   from ESP_ACTION

       when MCEP.M_ActionInd(PDU_M_Action)

{******* Situacao Gerente - avalia se aceita delegacao ****** }

            {****** Parametros de M-Action
            {       II : Nome da primitiva
            {       M : Modo Confirmado
            {       BOC: Tabela objeto
            {       AC: Controle de acesso - autorizacao
            {       AT : Tipo de acao ( ask ou act )
            {      AA :Argumentos de acao ( nome do agente}

       provided   ( PDU_M_Action.II = 'DelegAct' ) and
                  ( PDU_M_Action.BOC = 'TableA' ) and
                  ( PDU_M_Action.AT = 'Ask' ) and
                  ( PDU_M_Action.AC = Autorization )
       to ESP_DESACT
            name  EsperaDesconexaoDelegActM:

            begin
            {****** Parametros de M-Action
            {         II : Nome da primitiva
            {         M : Modo Confirmado
            {         BOC: Tabela objeto
            {         AC: Controle de acesso - autorizacao
            {         AT : Tipo de acao ( ask ou act )
            {      AA :Argumentos de acao ( nome do agente}
            {         E: Erro         }
                 PDU_M_Action.MOC:= 'TableA' ;
                 if ( AgentPointer < TamTableA ) then
                 begin
                      AgentPointer := AgentPointer + 1 ;
                    TableA[AgentPointer] := PDU_M_Action.AA;
                      SDU_DelegAct.Delegation := 'Agent' ;
                 SDU_DelegAct.Agent := PDU_M_Action.AA;
                      SDU_DelegAct.Status := 0 ;
                      PDU_M_Action.E := 0 ;
                 output UCEP.DelegActInd(SDU_DelegAct);
                 end
                 else
                      PDU_M_Action.E := 29 ;
```

```
          output MCEP.M_ActionResp(PDU_M_Action);
     end;


     { Situacao Agente - altera tabela de gerente }
provided   ( PDU_M_Action.II = 'DelegAct' ) and
           ( PDU_M_Action.BOC = 'TableM' ) and
           ( PDU_M_Action.AT = 'Act' ) and
           ( PDU_M_Action.AC = Autorization )

to ESP_DESACT
     name  EsperaDesconexaoDelegActA:

     begin
     {****** Parametros de M-Action
     {       II : Nome da primitiva
     {       M : Modo Confirmado
     {       BOC: Tabela objeto
     {       AC: Controle de acesso - autorizacao
     {       AT : Tipo de acao ( ask ou act )
     {      AA :Argumentos de acao (nome do gerente}
     {       E: Erro         }

          PDU_M_Action.MOC:= 'TableM' ;
          if ( ManagerPointer < TamTableM ) then
          begin
               ManagerPointer := ManagerPointer+1;
               for i:=ManagerPointer downto 2 do
                    TableM[i] := TableM[i-1] ;
               TableM[1] := PDU_M_Action.AA ;
               PDU_M_Action.E := 0 ;
          SDU_DelegAct.Delegation := 'Manager';
            SDU_DelegAct.Manager := PDU_M_Action.AA;
               SDU_DelegAct.Status := 0 ;
          output UCEP.DelegActInd(SDU_DelegAct);
          end
          else
               PDU_M_Action.E := 29 ;
          output MCEP.M_ActionResp(PDU_M_Action);
     end;


provided   ( PDU_M_Action.II = 'DelegAct' ) and
           ( PDU_M_Action.AC <> Autorization )

to REPOUSO             { Erros de seguranca do SGH }
     name ErroSGHDelegAct7:
     begin
          PDU_M_Action.E := 28 ;
          output MCEP.M_ActionResp(PDU_M_Action);
     end;

provided   ( PDU_M_Action.II = 'DelegAct' ) and
           (( PDU_M_Action.BOC <> 'TableA' ) or
           ( PDU_M_Action.BOC <> 'TableM' ))

to REPOUSO              { Erros de tabelas do SGH }
     name ErroSGHPolicyARead8:
     begin
          PDU_M_Action.E := 27 ;
          output MCEP.M_ActionResp(PDU_M_Action);
     end;
```

```
    from ESP_DESACT

         when MCEP.A_ReleaseInd(PDU_A_Release)

         to REPOUSO
              name  DesconectaDelegAct:
              begin
              end;

(************* PolicyMRead Iniciador/Respondedor ************)

    from REPOUSO
         when UCEP.PolicyMReadReq(SDU_PolicyMRead)

         to REPOUSO
              name  LeTablePolicy:
              begin
                  for i:=1 to TamTablePolicy do
                    SDU_PolicyMRead.List[i]:= TablePolicy[i];
                output UCEP.PolicyMReadConf(SDU_PolicyMRead);
              end;

(**************** PolicyARead Iniciador ********************)

    from REPOUSO              {usuario solicita diretriz }
         when UCEP.PolicyAReadReq(SDU_PolicyARead)

              {******* Parametros de A_Associate :
                        ACN: Nome da Aplicacao SGH
                        APT,AEQ: PolicyARead
                        API: Nome local
                        AEI: Nome agente chamado
                        U: Controle de Acesso - autorizacao}

         to ESP_CONEC_10
              name ConexaoPolicyARead:
              begin
                   Agent := SDU_PolicyARead.Agent ;
                   Destiny := SDU_PolicyARead.Agent ;
                   PDU_A_Associate.ACN := 'SGH' ;
                   PDU_A_Associate.APT := 'PolicyARead' ;
                   PDU_A_Associate.AEQ := 'PolicyARead' ;
                   PDU_A_Associate.API := LocalName ;
                   PDU_A_Associate.AEI := Destiny ;
                   PDU_A_Associate.UD := Autorization ;
              output MCEP.A_AssociateReq(PDU_A_Associate);
              end;

    from ESP_CONEC_10

         when MCEP.A_AssociateConf(PDU_A_Associate)

         provided    ( PDU_A_Associate.R = 0 ) and
                     ( PDU_A_Associate.RAPT = 'PolicyARead' )

         to ESP_TABPOL10
              name  PedeTablePolicy:

              {****** Parametros de M-Get
              {       II : Nome da primitiva
              {       BOC: Tabela solicitada
              {       AC: Controle de acesso - autorizacao}
              begin
```

```
            PDU_M_Get.II := 'PolicyARead' ;
            PDU_M_Get.BOC:= 'TablePolicy' ;
            PDU_M_Get.SC := Destiny ;
            PDU_M_Get.SY := LocalName ;
            PDU_M_Get.AC:= Autorization ;
            output MCEP.M_GetReq(PDU_M_Get);
        end;

    provided   ( PDU_A_Associate.R = 0 ) and
               ( PDU_A_Associate.RAPT <> 'PolicyARead' )

    to REPOUSO              { Erros do SGH }
        name ErroSGHPolicyARead1:
        begin
           SDU_PolicyARead.Status := 27 ;
          output UCEP.PolicyAReadConf(SDU_PolicyARead);
        end;

    provided        ( PDU_A_Associate.R <> 0 )
    to REPOUSO              { Erros do CMISE }
        name ErroCMISEPolicyARead1:
        begin
           SDU_PolicyARead.Status := PDU_A_Associate.R;
          output UCEP.PolicyAReadConf(SDU_PolicyARead);
        end;

from ESP_TABPOL10

    when MCEP.M_GetConf(PDU_M_Get)

        {****** Parametros de M-Get
                  II: Nome da primitiva
                  MOC: Nome da Tabela enviada
                  AL: Tabela enviada
                  E: Erro         }

    provided   ( PDU_M_Get.E = 0 ) and
               ( PDU_M_Get.II= 'PolicyARead' ) and
               ( PDU_M_Get.MOC= 'TablePolicy' )
    to REPOUSO
        name  DesconectaPolicyARead:

    begin  { escreve tabela do agente em PolicyList }
       for i:=1 to TamTablePolicy do
    SDU_PolicyARead.PolicyList[i]:= PDU_M_Get.AL[i];
       SDU_PolicyARead.Status := 0 ;
       output UCEP.PolicyAReadConf(SDU_PolicyARead);

        {******* Parametros de A_Release :
                  ACN: Nome da Aplicacao SGH
                  APT,AEQ: PolicyARead
                  API: Nome local
                  AEI: Nome agente chamado }

        PDU_A_Release.ACN := 'SGH' ;
        PDU_A_Release.APT := 'PolicyARead' ;
        PDU_A_Release.AEQ := 'PolicyARead' ;
        PDU_A_Release.API := LocalName ;
        PDU_A_Release.AEI := Destiny ;
        output MCEP.A_ReleaseReq(PDU_A_Release);
    end;

    provided   ( PDU_M_Get.E = 0 ) and
```

```
            ( PDU_M_Get.II <> 'PolicyARead' )
      to REPOUSO              { Erros do SGH }
           name ErroSGHPolicyARead3:
           begin
              SDU_PolicyARead.Status := 27 ;
             output UCEP.PolicyAReadConf(SDU_PolicyARead);
           end;

      provided        ( PDU_M_Get.E <> 0 )
      to REPOUSO              { Erros do CMISE }
           name ErroCMISEPolicyARead3:
           begin
              SDU_PolicyARead.Status := PDU_M_Get.E ;
             output UCEP.PolicyAReadConf(SDU_PolicyARead);
           end;

(*************** PolicyARead Respondedor ******************)

   from REPOUSO
      when MCEP.A_AssociateInd(PDU_A_Associate)
           {******* Parametros de A_Associate :
                      ACN: Nome da Aplicacao SGH
                      APT,AEQ: PolicyARead
                      API: Nome local
                      AEI: Nome agente chamado
                      UD:Controle de Acesso - autorizacao}

      provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                 ( PDU_A_Associate.APT = 'PolicyARead' )and
                 ( PDU_A_Associate.AEQ = 'PolicyARead' )and
                 ( PDU_A_Associate.UD = Autorization ) and
                 ( PDU_A_Associate.AEI = LocalName )

      to ENV_TABPOL10
           name EstabeleceConexaoPolicyARead:
           begin
                Agent := PDU_A_Associate.API ;
                PDU_A_Associate.RAPT := 'PolicyARead' ;
                PDU_A_Associate.RAEQ := 'PolicyARead' ;
                PDU_A_Associate.R := 0 ;
             output MCEP.A_AssociateResp(PDU_A_Associate);
           end;

      provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                 ( PDU_A_Associate.APT = 'PolicyARead' )and
                 ( PDU_A_Associate.AEQ = 'PolicyARead' )and
                 (( PDU_A_Associate.UD <> Autorization ) or
                 ( PDU_A_Associate.AEI <> LocalName ))

      to REPOUSO              { Erros do SGH }
           name ErroSGHPolicyARead4:
           begin
                PDU_A_Associate.R := 28 ;
             output MCEP.A_AssociateResp(PDU_A_Associate);
           end;

   from ENV_TABPOL10

      when MCEP.M_GetInd(PDU_M_Get)

      {****** Parametros de M-Get para pedir TablePolicy
                II : Nome da primitiva
                BOC: Tabela solicitada
```

```
                AC: Controle de acesso - autorizacao    }

        provided   ( PDU_M_Get.II = 'PolicyARead' ) and
                   ( PDU_M_Get.BOC = 'TablePolicy' ) and
                   ( PDU_M_Get.AC = Autorization )

        to ESP_DESC10
             name  EnviaTablePPolicyARead:

             begin

             {****** Parametros de M-Get
                      II: Nome da primitiva
                      MOC: Nome da Tabela enviada
                      AL: Tabela enviada
                      E: Erro         }

                  PDU_M_Get.MOC := 'TablePolicy' ;
                  for i:=1 to TamTablePolicy do
                       PDU_M_Get.AL[i] := TablePolicy[i];
                  PDU_M_Get.E := 0 ;
                  output MCEP.M_GetResp(PDU_M_Get);
             end;

        provided   ( PDU_M_Get.II = 'PolicyARead' ) and
                   ( PDU_M_Get.AC <> Autorization )

        to REPOUSO             { Erros de seguranca do SGH }
             name ErroSGHPolicyARead7:
             begin
                  PDU_M_Get.E := 28 ;
                  output MCEP.M_GetResp(PDU_M_Get);
             end;

        provided   ( PDU_M_Get.II = 'PolicyARead' ) and
                   ( PDU_M_Get.BOC <> 'TablePolicy' )
        to REPOUSO              { Erros de tabelas do SGH }
             name ErroSGHPolicyARead8:
             begin
                  PDU_M_Get.E := 27 ;
                  output MCEP.M_GetResp(PDU_M_Get);
             end;

   from ESP_DESC10
        when MCEP.A_ReleaseInd(PDU_A_Release)

        to REPOUSO
             name  DesconectaPolicyARead:
             begin
             end;

(**************** PolicyAChange Iniciador *****************)

   from REPOUSO          { altera diretriz de agente }
        when UCEP.PolicyAChangeReq(SDU_PolicyAChange)
             {******* Parametros de A_Associate :
                      ACN: Nome da Aplicacao SGH
                      APT,AEQ: PolicyAChange
                      API: Nome local
                      AEI: Nome agente chamado
                      U: Controle de Acesso - autorizacao}
        to ESP_CONEC_11
             name ConexaoPolicyAChange:
```

```
        begin
             Agent := SDU_PolicyAChange.Agent ;
             Destiny := SDU_PolicyAChange.Agent ;
             PDU_A_Associate.ACN := 'SGH' ;
             PDU_A_Associate.APT := 'PolicyAChange' ;
             PDU_A_Associate.AEQ := 'PolicyAChange' ;
             PDU_A_Associate.API := LocalName ;
             PDU_A_Associate.AEI := Destiny ;
             PDU_A_Associate.UD := Autorization ;
        output MCEP.A_AssociateReq(PDU_A_Associate);
        end;

from ESP_CONEC_11

    when MCEP.A_AssociateConf(PDU_A_Associate)

    provided    ( PDU_A_Associate.R = 0 ) and
                ( PDU_A_Associate.RAPT='PolicyAChange')
    to ESP_TABPOL11
        name  EscreveTablePolicy:

        {****** Parametros de M-Set
        {       II : Nome da primitiva
        {       M : Modo Confirmado
        {       BOC: Tabela solicitada
        {       AC: Controle de acesso - autorizacao  }
        begin
             PDU_M_Set.II := 'PolicyAChange' ;
             PDU_M_Set.M := 'Confirmed' ;
             PDU_M_Set.BOC:= 'TablePolicy' ;
             for i := 1 to TamTablePolicy do
    PDU_M_Set.AL[i]:=SDU_PolicyAChange.PolicyList[i] ;
             PDU_M_Set.SC := Destiny ;
             PDU_M_Set.SY := LocalName ;
             PDU_M_Set.AC:= Autorization ;
             output MCEP.M_SetReq(PDU_M_Set);
        end;

    provided   ( PDU_A_Associate.R = 0 ) and
               ( PDU_A_Associate.RAPT <> 'PolicyAChange')
    to REPOUSO              { Erros do SGH }
        name ErroSGHPolicyAChange1:
        begin
             SDU_PolicyAChange.Status := 27 ;
    output UCEP.PolicyAChangeConf(SDU_PolicyAChange);
        end;

    provided        ( PDU_A_Associate.R <> 0 )
    to REPOUSO              { Erros do CMISE }
        name ErroCMISEPolicyAChange1:
        begin
        SDU_PolicyAChange.Status := PDU_A_Associate.R;
    output UCEP.PolicyAChangeConf(SDU_PolicyAChange);
        end;

from ESP_TABPOL11

    when MCEP.M_SetConf(PDU_M_Set)

        {****** Parametros de M-Set
                II: Nome da primitiva
                MOC: Nome da Tabela enviada
                AL: Tabela enviada
```

```
                    E: Erro          }

          provided   ( PDU_M_Set.E = 0 ) and
                     ( PDU_M_Set.II= 'PolicyAChange' ) and
                     ( PDU_M_Set.MOC= 'TablePolicy' )
          to REPOUSO
               name  DesconectaPolicyAChange:
               begin

               {******* Parametros de A_Release :
                          ACN: Nome da Aplicacao SGH
                          APT,AEQ: PolicyAChange
                          API: Nome local
                          AEI: Nome agente chamado }

                     PDU_A_Release.ACN := 'SGH' ;
                     PDU_A_Release.APT := 'PolicyAChange' ;
                     PDU_A_Release.AEQ := 'PolicyAChange' ;
                     PDU_A_Release.API := LocalName ;
                     PDU_A_Release.AEI := Destiny ;
                     output MCEP.A_ReleaseReq(PDU_A_Release);
                     SDU_PolicyAChange.Status := 0 ;
          output UCEP.PolicyAChangeConf(SDU_PolicyAChange);
               end;

          provided   ( PDU_M_Set.E = 0 ) and
                     ( PDU_M_Set.II <> 'PolicyAChange' )
          to REPOUSO             { Erros do SGH }
               name ErroSGHPolicyAChange3:
               begin
               SDU_PolicyAChange.Status := 27 ;
          output UCEP.PolicyAChangeConf(SDU_PolicyAChange);
               end;

          provided        ( PDU_M_Set.E <> 0 )
          to REPOUSO             { Erros do CMISE }
               name ErroCMISEPolicyAChange3:
               begin
               SDU_PolicyAChange.Status := PDU_M_Set.E ;
          output UCEP.PolicyAChangeConf(SDU_PolicyAChange);
               end;

(**************** PolicyAChange Respondedor ****************)

     from REPOUSO
          when MCEP.A_AssociateInd(PDU_A_Associate)

               {******* Parametros de A_Associate :
                          ACN: Nome da Aplicacao SGH
                          APT,AEQ: PolicyAChange
                          API: Nome local
                          AEI: Nome agente chamado
                          UD: Controle de Acesso -autorizacao}

          provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                     ( PDU_A_Associate.APT= 'PolicyAChange')and
                     ( PDU_A_Associate.AEQ='PolicyAChange') and
                     ( PDU_A_Associate.UD = Autorization ) and
                     ( PDU_A_Associate.AEI = LocalName )

          to ENV_TABPOL11
               name EstabeleceConexaoPolicyAChange:
               begin
```

```
                Agent := PDU_A_Associate.API ;
                PDU_A_Associate.RAPT := 'PolicyAChange' ;
                PDU_A_Associate.RAEQ := 'PolicyAChange' ;
                PDU_A_Associate.R := 0 ;
              output MCEP.A_AssociateResp(PDU_A_Associate);
            end;

      provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                 ( PDU_A_Associate.APT ='PolicyAChange')and
                 ( PDU_A_Associate.AEQ='PolicyAChange') and
                 (( PDU_A_Associate.UD <> Autorization ) or
                 ( PDU_A_Associate.AEI <> LocalName ))
      to REPOUSO              { Erros do SGH }
            name ErroSGHPolicyAChange4:
            begin
                PDU_A_Associate.R := 28 ;
              output MCEP.A_AssociateResp(PDU_A_Associate);
            end;

  from ENV_TABPOL11

      when MCEP.M_SetInd(PDU_M_Set)

      {****** Parametros de M-Set com TablePolicy
                II : Nome da primitiva
                BOC: Tabela solicitada
                AC: Controle de acesso - autorizacao    }

      provided   ( PDU_M_Set.II = 'PolicyAChange' ) and
                 ( PDU_M_Set.BOC = 'TablePolicy' ) and
                 ( PDU_M_Set.AC = Autorization )
      to ESP_DESC11
            name  EnviaTablePPolicyAChange:

            begin
            {****** Parametros de M-Set
                      II: Nome da primitiva
                      MOC: Nome da Tabela enviada
                      AL: Tabela enviada
                      E: Erro         }

                PDU_M_Set.MOC := 'TablePolicy' ;
                for i:=1 to TamTablePolicy do
                      TablePolicy[i] := PDU_M_Set.AL[i] ;
                PDU_M_Set.E := 0 ;
                output MCEP.M_SetResp(PDU_M_Set);
            end;

      provided   ( PDU_M_Set.II = 'PolicyAChange' ) and
                 ( PDU_M_Set.AC <> Autorization )
      to REPOUSO             { Erros de seguranca do SGH }
            name ErroSGHPolicyAChange7:
            begin
                PDU_M_Set.E := 28 ;
                output MCEP.M_SetResp(PDU_M_Set);
            end;

      provided   ( PDU_M_Set.II = 'PolicyAChange' ) and
                 ( PDU_M_Set.BOC <> 'TablePolicy' )
      to REPOUSO              { Erros de tabelas do SGH }
            name ErroSGHPolicyAChange8:
            begin
                PDU_M_Set.E := 27 ;
```

```
                output MCEP.M_SetResp(PDU_M_Set);
           end;

     from ESP_DESC11
          when MCEP.A_ReleaseInd(PDU_A_Release)

          to REPOUSO
               name  DesconectaPolicyAChange:
               begin
               end;


(***************** PolicyMLoad Iniciador *******************)

     from REPOUSO               {usuario solicita diretriz }
          when UCEP.PolicyMLoadReq(SDU_PolicyMLoad)

               {******* Parametros de A_Associate :
                         ACN: Nome da Aplicacao SGH
                         APT,AEQ: PolicyMLoad
                         API: Nome local
                         AEI: Nome gerente chamado
                         U: Controle de Acesso - autorizacao}

          to ESP_CONEC_12
               name ConexaoPolicyMLoad:
               begin
                    Manager := SDU_PolicyMLoad.Manager ;
                    Destiny := SDU_PolicyMLoad.Manager ;
                    PDU_A_Associate.ACN := 'SGH' ;
                    PDU_A_Associate.APT := 'PolicyMLoad' ;
                    PDU_A_Associate.AEQ := 'PolicyMLoad' ;
                    PDU_A_Associate.API := LocalName ;
                    PDU_A_Associate.AEI := Destiny ;
                    PDU_A_Associate.UD := Autorization ;
               output MCEP.A_AssociateReq(PDU_A_Associate);
               end;

     from ESP_CONEC_12

          when MCEP.A_AssociateConf(PDU_A_Associate)

          provided    ( PDU_A_Associate.R = 0 ) and
                       ( PDU_A_Associate.RAPT = 'PolicyMLoad' )

          to ESP_TABPOL12
               name  PedeTablePolicy:

               {****** Parametros de M-Get
               {       II : Nome da primitiva
               {       BOC: Tabela solicitada
               {       AC: Controle de acesso - autorizacao }
               begin
                    PDU_M_Get.II := 'PolicyMLoad' ;
                    PDU_M_Get.BOC:= 'TablePolicy' ;
                    PDU_M_Get.SC := Destiny ;
                    PDU_M_Get.SY := LocalName ;
                    PDU_M_Get.AC:= Autorization ;
                    output MCEP.M_GetReq(PDU_M_Get);
               end;

          provided   ( PDU_A_Associate.R = 0 ) and
                     ( PDU_A_Associate.RAPT <> 'PolicyMLoad' )
```

```
      to REPOUSO              { Erros do SGH }
           name ErroSGHPolicyMLoad1:
           begin
              SDU_PolicyMLoad.Status := 27 ;
             output UCEP.PolicyMLoadConf(SDU_PolicyMLoad);
           end;

      provided        ( PDU_A_Associate.R <> 0 )

      to REPOUSO              { Erros do CMISE }
           name ErroCMISEPolicyMLoad1:
           begin
              SDU_PolicyMLoad.Status := PDU_A_Associate.R;
             output UCEP.PolicyMLoadConf(SDU_PolicyMLoad);
           end;

 from ESP_TABPOL12

      when MCEP.M_GetConf(PDU_M_Get)

           {****** Parametros de M-Get
                     II: Nome da primitiva
                     MOC: Nome da Tabela enviada
                     AL: Tabela enviada
                     E: Erro         }

      provided   ( PDU_M_Get.E = 0 ) and
                 ( PDU_M_Get.II= 'PolicyMLoad' ) and
                 ( PDU_M_Get.MOC= 'TablePolicy' )

      to REPOUSO
           name  DesconectaPolicyMLoad:

           begin      { atualiza tabela de diretrizes }
                for i:=1 to TamTablePolicy do
                     TablePolicy[i] := PDU_M_Get.AL[i];

           {******* Parametros de A_Release :
                     ACN: Nome da Aplicacao SGH
                     APT,AEQ: PolicyMLoad
                     API: Nome local
                     AEI: Nome gerente chamado }

                PDU_A_Release.ACN := 'SGH' ;
                PDU_A_Release.APT := 'PolicyMLoad' ;
                PDU_A_Release.AEQ := 'PolicyMLoad' ;
                PDU_A_Release.API := LocalName ;
                PDU_A_Release.AEI := Destiny ;
                output MCEP.A_ReleaseReq(PDU_A_Release);
                SDU_PolicyMLoad.Status := 0 ;
             output UCEP.PolicyMLoadConf(SDU_PolicyMLoad);
           end;

      provided   ( PDU_M_Get.E = 0 ) and
                 ( PDU_M_Get.II <> 'PolicyMLoad' )

      to REPOUSO              { Erros do SGH }
           name ErroSGHPolicyMLoad3:
           begin
              SDU_PolicyMLoad.Status := 27 ;
             output UCEP.PolicyMLoadConf(SDU_PolicyMLoad);
           end;
```

```
        provided        ( PDU_M_Get.E <> 0 )

        to REPOUSO              { Erros do CMISE }
             name ErroCMISEPolicyMLoad3:
             begin
                SDU_PolicyMLoad.Status := PDU_M_Get.E ;
               output UCEP.PolicyMLoadConf(SDU_PolicyMLoad);
             end;

(**************** PolicyMLoad Respondedor *****************)

     from REPOUSO
        when MCEP.A_AssociateInd(PDU_A_Associate)

             {******* Parametros de A_Associate :
                        ACN: Nome da Aplicacao SGH
                        APT,AEQ: PolicyMLoad
                        API: Nome local
                        AEI: Nome gerente chamado
                        UD: Controle de Acesso -autorizacao}

        provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                   ( PDU_A_Associate.APT = 'PolicyMLoad')and
                   ( PDU_A_Associate.AEQ = 'PolicyMLoad' )and
                   ( PDU_A_Associate.UD = Autorization ) and
                   ( PDU_A_Associate.AEI = LocalName )

        to ENV_TABPOL12
             name EstabeleceConexaoPolicyMLoad:
             begin
                  Agent := PDU_A_Associate.API ;
                  PDU_A_Associate.RAPT := 'PolicyMLoad' ;
                  PDU_A_Associate.RAEQ := 'PolicyMLoad' ;
                  PDU_A_Associate.R := 0 ;
               output MCEP.A_AssociateResp(PDU_A_Associate);
             end;

        provided   ( PDU_A_Associate.ACN = 'SGH' ) and
                   ( PDU_A_Associate.APT = 'PolicyMLoad' )and
                   ( PDU_A_Associate.AEQ = 'PolicyMLoad' )and
                   (( PDU_A_Associate.UD <> Autorization ) or
                   ( PDU_A_Associate.AEI <> LocalName ))

        to REPOUSO              { Erros do SGH }
             name ErroSGHPolicyMLoad4:
             begin
                 PDU_A_Associate.R := 28 ;
               output MCEP.A_AssociateResp(PDU_A_Associate);
             end;

     from ENV_TABPOL12

        when MCEP.M_GetInd(PDU_M_Get)

     {****** Parametros de M-Get para pedir TablePolicy
                  II : Nome da primitiva
                  BOC: Tabela solicitada
                  AC: Controle de acesso - autorizacao    }

        provided   ( PDU_M_Get.II = 'PolicyMLoad' ) and
                   ( PDU_M_Get.BOC = 'TablePolicy' ) and
                   ( PDU_M_Get.AC = Autorization )
```

```
        to ESP_DESC12
             name  EnviaTablePPolicyMLoad:

             begin

             {****** Parametros de M-Get
                       II: Nome da primitiva
                       MOC: Nome da Tabela enviada
                       AL: Tabela enviada
                       E: Erro          }

                  PDU_M_Get.II := 'PolicyMLoad' ;
                  PDU_M_Get.MOC := 'TablePolicy' ;
                  for i:=1 to TamTablePolicy do
                       PDU_M_Get.AL[i] := TablePolicy[i];
                  PDU_M_Get.E := 0 ;
                  output MCEP.M_GetResp(PDU_M_Get);
             end;

        provided   ( PDU_M_Get.II = 'PolicyMLoad' ) and
                   ( PDU_M_Get.AC <> Autorization )

        to REPOUSO              { Erros de seguranca do SGH}
             name ErroSGHPolicyMLoad7:
             begin
                  PDU_M_Get.E := 28 ;
                  output MCEP.M_GetResp(PDU_M_Get);
             end;

        provided   ( PDU_M_Get.II = 'PolicyMLoad' ) and
                   ( PDU_M_Get.BOC <> 'TablePolicy' )

        to REPOUSO              { Erros de tabelas do SGH }
             name ErroSGHPolicyMLoad8:
             begin
                  PDU_M_Get.E := 27 ;
                  output MCEP.M_GetResp(PDU_M_Get);
             end;

   from ESP_DESC12
        when MCEP.A_ReleaseInd(PDU_A_Release)

        to REPOUSO
             name  DesconectaPolicyMLoad:
             begin
             end;

  end; (* de SGH_ENTITY_BODY *)


(**************************************************************)
(****                                                      ****)
(****  ESPECIFICACAO DO MODULO SISTEMA DE GERENCIAMENTO    ****)
(****                                                      ****)
(**************************************************************)

module USER_TYPE activity;

  ip
     UCEP : array [0..quant_MCEP] of SGH_interface(user);
     USYS: SYS_interface( provider);
  end;
```

```
  body USER_BODY for USER_TYPE;
    ip SYS: SYS_interface(user);
       USR: SYS_interface(provider);

(******************** Variaveis Locais *********************)

    var
     temp    : integer;  { variavel temporaria }
     status  : integer;
     i       : integer;  { variavel contador }
     n       : integer;  { numero do UCEP}
     teste   : boolean ;
     tecla   : array[0..0] of char;

     Autorization: string;
     Manager: integer ;
     Agent: integer ;
     LocalName: integer ;      { Nome do gerente local }
     Policy: integer;
     TableM: array [1..TamTableM] of integer;
     TableA: array [1..TamTableA] of integer;
     SDU_Init:TABELA_TYPE;
     SDU_MInitPrin: DOMINIO_TYPE ;
     SDU_ManagerInit: DOMINIO_TYPE ;
     SDU_AgentCreate: DOMINIO_TYPE ;
     SDU_AgentElimin: DOMINIO_TYPE ;
     SDU_ManagerRead: TABELA_TYPE ;
     SDU_AgentRead: TABELA_TYPE ;
     SDU_DelegAct: DELEG_TYPE ;
     SDU_PolicyMRead: TABELA_TYPE ;
     SDU_PolicyARead: POLICY_TYPE ;
     SDU_PolicyAChange: POLICY_TYPE ;
     SDU_PolicyMLoad: POLICY_TYPE ;


(***************** Relacao de Estados *********************)

    state
      REPOUSO_USER,
      INICMAN,
      INITALL,
      PEDE_INIC,
      LE_GER,
      LE_AG,
      DELEGA,
      POLMR,
      POLAR,
      POLML,
      POLAC,
      ESP_AGCR,
      ESP_AGELIM ;

(****************** Programa de Usuario ******************)

    initialize
      to REPOUSO_USER
      name RepousoUsr:
      begin
     connect SYS to USR;
     WriteLn;
     WriteString ('Lista de Alternativas:');
     WriteLn;
     WriteString ('<P> Inicia Gerente Principal');
```

```
    WriteLn;
    WriteString ('<I> Iniciar');
    WriteLn;
    WriteString ('<K> Criar Agente');
    WriteLn;
    WriteString ('<E> Eliminar Agente');
    WriteLn;
    WriteString ('<D> Delegar');
    WriteLn;
    WriteString ('<G> Ler Gerente');
    WriteLn;
    WriteString ('<A> Ler Agente');
    WriteLn;
    WriteString ('<M> Ler Diretriz do Gerente');
    WriteLn;
    WriteString ('<R> Ler Diretriz do Agente ');
    WriteLn;
    WriteString ('<L> Carregar Diretriz');
    WriteLn;
    WriteString ('<C> Mudar Diretriz');
    WriteLn;
    WriteString ('Escolha :');
    WriteLn;
(* ESTA PARTE FOI RETIRADA SOMENTE PARA SIMULACAO *)
     (* teste:=false;
    repeat
      if KeyPressed then
      begin
        tecla[0]:=ReadKey;
        teste:=true;
      end;
    until teste;
    if (tecla='p') or (tecla='P')
    then output SYS.INICMANPRIN;
    if (tecla='i') or (tecla='I')
    then output SYS.INICIAR;
    if (tecla ='k') or (tecla='K')
    then output SYS.CRIA_AG;
    if (tecla ='E') or (tecla='e')
    then output SYS.ELIM_AG;
    if (tecla='d') or (tecla='D')
    then output SYS.DELEGAR;
    if (tecla='g') or (tecla='G')
    then output SYS.LE_GERENTE;
    if (tecla='a') or (tecla='A')
    then output SYS.LE_AGENTE;
    if (tecla='m') or (tecla='M')
    then output SYS.POL_MREAD;
    if (tecla='r') or (tecla='R')
    then output SYS.POL_AREAD;
    if (tecla='l') or (tecla='L')
    then output SYS.POL_MLOAD;
    if (tecla='c') or (tecla='C')
    then output SYS.POL_ACHANGE;  *)
     end;

(* ESTA PARTE FOI ACRESCENTADA SOMENTE PARA SIMULACAO *)
    trans
    when USYS.INICMANPRIN
    name D0:
    begin
       output SYS.INICMANPRIN;
    end;
```

```
 trans
when USYS.INIT_TEST
name D11:
begin
   output SYS.INIT_TEST;
end;

 trans
when USYS.INICIAR
name D1:
begin
   output SYS.INICIAR;
end;

trans
when USYS.DELEGAR
name D2:
begin
   output SYS.DELEGAR;
end;
trans
when USYS.LE_GERENTE
name D3:
begin
   output SYS.LE_GERENTE;
end;

trans
when USYS.LE_AGENTE
name D4:
begin
   output SYS.LE_AGENTE;
end;
trans
when USYS.POL_MREAD
name D5:
begin
   output SYS.POL_MREAD;
end;
trans
when USYS.POL_AREAD
name D6:
begin
   output SYS.POL_AREAD;
end;
trans
when USYS.POL_MLOAD
name D7:
begin
   output SYS.POL_MLOAD;
end;

trans
when USYS.POL_ACHANGE
name D8:
begin
   output SYS.POL_ACHANGE;
end;

trans
when USYS.CRIA_AG
name D9:
begin
```

```
      output SYS.CRIA_AG;
    end;
   trans
    when USYS.ELIM_AG
    name D10:
    begin
      output SYS.ELIM_AG;
    end;

trans
     any NCEP: MCEP_FAIXA_TYPE do

     (******Init*****)
   from REPOUSO_USER
     when USR.INIT_TEST
     to INITALL
    name InitTudo:
    begin
      WriteLn;
      WriteString ('Numero do UCEP:');
      ReadInt(n);
      WriteString ('LocalName=');
      ReadInt ( SDU_Init.List[12]);
      WriteLn;
      WriteString ('TableM[1]=');
      ReadInt ( SDU_Init.List[1]);
      WriteLn;
      WriteString ('TableM[2]=');
      ReadInt ( SDU_Init.List[2]);
      WriteLn;
      WriteString ('TableM[3]=');
      ReadInt ( SDU_Init.List[3]);
      WriteLn;
      WriteString ('ManagerPointer=');
      ReadInt ( SDU_Init.List[4]);
      WriteLn;
      WriteString ('TableA[1]=');
      ReadInt ( SDU_Init.List[5]);
      WriteLn;
      WriteString ('TableA[2]=');
      ReadInt ( SDU_Init.List[6]);
      WriteLn;
      WriteString ('TableA[3]=');
      ReadInt ( SDU_Init.List[7]);
      WriteLn;
      WriteString ('TableA[4]=');
      ReadInt ( SDU_Init.List[8]);
      WriteLn;
      WriteString ('TableA[5]=');
      ReadInt ( SDU_Init.List[9]);
      WriteLn;
      WriteString ('AgentPointer=');
      ReadInt ( SDU_Init.List[10]);
      WriteLn;
      WriteString ('Diretriz=');
      ReadInt ( SDU_Init.List[11]);
      WriteLn;
      output UCEP[n].InitReq(SDU_Init);
    end ;

     from INITALL

    when  UCEP[NCEP].InitConf(SDU_Init)
```

```
  to REPOUSO_USER
  name ConfInit:
  begin
    WriteLn;
    WriteString('Status do Gerente: ');
    WriteInt(SDU_Init.Status);
    WriteLn;
   end;

(******Iniciar Gerente Principal*****)
from REPOUSO_USER
   when USR.INICMANPRIN
   to INICMAN
  name InicGerPrin:
  begin
    WriteLn;
    WriteString ('Numero do UCEP:');
    ReadInt(n);
    WriteString ('Nome local:');
    ReadInt ( LocalName);
    WriteLn;
    SDU_MInitPrin.Local:= LocalName;
    WriteString ( 'Senha de autorizacao :');
    ReadString ( Autorization);
    WriteLn;
    SDU_MInitPrin.Autorization := Autorization;
    WriteString('Diretriz Primeira:');
    ReadInt( Policy);
    WriteLn;
    SDU_MInitPrin.Agent :=Policy;
    output UCEP[n].MInitPrinReq(SDU_MInitPrin);
  end ;

   from INICMAN

  when  UCEP[NCEP].MInitPrinConf(SDU_MInitPrin)
  to REPOUSO_USER
  name ConfInicGerPrin:
  begin
    WriteLn;
    WriteString('Status do Gerente: ');
    WriteInt(SDU_MInitPrin.Status);
    WriteLn;
  end;

(******Iniciar Gerente *****)
from REPOUSO_USER
   when USR.INICIAR
   to PEDE_INIC
  name InicGerente:
  begin
    WriteLn;
    WriteString ('Numero do UCEP:');
    ReadInt(n);
    WriteString ('Gerente a ser chamado:');
    ReadInt ( Manager);
    WriteLn;
    SDU_ManagerInit.Manager:= Manager;
    WriteString ('Nome local:');
    ReadInt ( LocalName);
    WriteLn;
    SDU_ManagerInit.Local:= LocalName;
    WriteString ( 'Senha de autorizacao :');
```

```
    ReadString ( Autorization);
    WriteLn;
    SDU_ManagerInit.Autorization := Autorization;
    output UCEP[n].ManagerInitReq(SDU_ManagerInit);
  end ;

   from PEDE_INIC

  when  UCEP[NCEP].ManagerInitConf(SDU_ManagerInit)

  to REPOUSO_USER
  name ConfInicGer:
  begin
    WriteLn;
    WriteString('Status do Gerente: ');
    WriteInt(SDU_ManagerInit.Status);
    WriteLn;
   end;

   (******Criar Agente*****)
from REPOUSO_USER
   when USR.CRIA_AG
   to ESP_AGCR
  name CriaAgente:
  begin
    WriteLn;
    WriteString ('Numero do UCEP:');
    ReadInt(n);
    WriteString ('Nome local:');
    ReadInt ( LocalName);
    WriteLn;
    WriteString ('Nome do Agente:');
    ReadInt  ( Agent);
    SDU_AgentCreate.Agent:=Agent;
    SDU_AgentCreate.Local:= LocalName;
    WriteString ( 'Senha de autorizacao :');
    ReadString ( Autorization);
    WriteLn;
    SDU_AgentCreate.Autorization := Autorization;
    output UCEP[n].AgentCreateReq(SDU_AgentCreate);
  end ;

   from ESP_AGCR

  when  UCEP[NCEP].AgentCreateConf(SDU_AgentCreate)
  to REPOUSO_USER
  name ConfCriaAg:
  begin
     WriteLn;
    WriteString('Status do Agente: ');
    WriteInt(SDU_AgentCreate.Status);
    WriteLn;
    WriteLn;
   end;

  (******Eliminar Agente*****)
from REPOUSO_USER
   when USR.ELIM_AG
   to ESP_AGELIM
  name ElimAgente:
  begin
    WriteLn;
    WriteString ('Numero do UCEP:');
```

```
    ReadInt(n);
    WriteString ('Nome local:');
    ReadInt ( LocalName);
    WriteLn;
    WriteString ('Nome do Agente:');
    ReadInt ( Agent);
    SDU_AgentElimin.Agent:=Agent;
    SDU_AgentElimin.Local:= LocalName;
    WriteString ( 'Senha de autorizacao :');
    ReadString( Autorization);
    WriteLn;
    SDU_AgentElimin.Autorization := Autorization;
    output UCEP[n].AgentEliminReq(SDU_AgentElimin);
  end ;

   from ESP_AGELIM

  when  UCEP[NCEP].AgentEliminConf(SDU_AgentElimin)
  to REPOUSO_USER
  name ConfElimAg:
  begin
    WriteLn;
    WriteString('Status do Agente: ');
    WriteInt(SDU_AgentElimin.Status);
    WriteLn;
    WriteLn;
   end;

(***********Ler Gerente*********)
  from REPOUSO_USER
   when USR.LE_GERENTE
   to LE_GER
  name LeGerente:
  begin
    WriteLn;
    WriteString ('Numero do UCEP:');
    ReadInt(n);
    WriteString ('Senha de autorizacao:');
    ReadString( Autorization);
    SDU_ManagerRead.Autorization:=Autorization;
    output UCEP[n].ManagerReadReq(SDU_ManagerRead);
  end ;

  from LE_GER
  when  UCEP[NCEP].ManagerReadConf(SDU_ManagerRead)

  to REPOUSO_USER
  name ConfLeGer:
  begin
    WriteLn;
    WriteString('Status do Gerente:');
    WriteInt(SDU_ManagerRead.Status);
    WriteLn;
    WriteString('Tabela de Gerentes:');
    WriteLn;
    temp:=SDU_ManagerRead.Pointer;
    i:=1;
    while (SDU_ManagerRead.List[i]<>0)
    do begin
       TableM[i]:=SDU_ManagerRead.List[i];
       WriteString('MANAGER');
       WriteInt(TableM[i]);
       WriteLn;
```

```
        i:=i+1;
        end;
   end;

  (*****Ler Agente*****)

  from REPOUSO_USER
    when USR.LE_AGENTE
    to LE_AG
   name ReadAg:
   begin
     WriteLn;
     WriteString ('Numero do UCEP:');
     ReadInt(n);
     WriteLn;
     WriteString ('Senha de autorizacao:');
     ReadString( Autorization);
     WriteLn;
     SDU_AgentRead.Autorization:=Autorization;
     output UCEP[n].AgentReadReq(SDU_AgentRead);
   end ;

 from LE_AG
   when  UCEP[NCEP].AgentReadConf(SDU_AgentRead)

   to REPOUSO_USER
   name ConfLeAg:
   begin
     WriteLn;
     WriteString('Status do Agente:');
     WriteInt(SDU_AgentRead.Status);
     WriteLn;
     WriteString('Tabela de Agentes:');
     WriteLn;
     i:=1;
     while (SDU_AgentRead.List[i]<>0)
     do begin
         TableA[i]:=SDU_AgentRead.List[i];
         WriteString('MANAGER');
         WriteInt(TableA[i]);
         WriteLn;
         i:=i+1;
        end;
   end;

(****Delegar*****)

  from REPOUSO_USER
     when USR.DELEGAR
     to DELEGA
   name MandaDeleg:
   begin
     WriteLn;
     WriteString('Numero do UCEP:');
     ReadInt(n);
     WriteLn;
     WriteString('Gerente que aceitara deleg:');
     ReadInt(Manager);
     SDU_DelegAct.Manager:=Manager;
     WriteLn;
     WriteString('Agente que sera delegado:');
     ReadInt(Agent);
     SDU_DelegAct.Agent:=Agent;
```

```
      WriteLn;
      WriteString('Senha de Autorizacao:');
      ReadString(Autorization);
      SDU_DelegAct.Autorization:=Autorization;
      WriteLn;
      output UCEP[n].DelegActReq(SDU_DelegAct);
    end;

    from DELEGA

        when  UCEP[NCEP].DelegActConf(SDU_DelegAct)

        to REPOUSO_USER
        name ConfDeleg:
        begin
          WriteLn;
          WriteString('Status da Delegacao=');
          WriteInt(SDU_DelegAct.Status);
          WriteLn;
        end;

        (****Diretriz MRead****)

  from REPOUSO_USER
    when USR.POL_MREAD
     to POLMR
    name PedePolMR:
    begin
      WriteLn;
      WriteString('Numero do UCEP:');
      ReadInt(n);
      WriteLn;
      WriteString('Senha de Autorizacao:');
      ReadString(Autorization);
      SDU_PolicyMRead.Autorization :=Autorization;
      output UCEP[n].PolicyMReadReq(SDU_PolicyMRead);
    end;

   from POLMR

        when  UCEP[NCEP].PolicyMReadConf(SDU_PolicyMRead)

        to REPOUSO_USER
        name ConfPolMR:
        begin
          WriteLn;
          WriteString('Status do Gerente =');
          WriteInt(SDU_PolicyMRead.Status);
          WriteLn;
          WriteString('Tabela de Diretrizes:');
          WriteLn;
          i:=1;
          while (SDU_PolicyMRead.List[i]<>0)
          do begin
            Policy:=SDU_PolicyMRead.List[i];
            WriteString('DIRETRIZ');
            WriteInt(Policy);
            WriteLn;
            i:=i+1;
           end;
        end;

(***Diretriz ARead****)
```

```
from REPOUSO_USER
  when USR.POL_AREAD
   to POLAR
  name PedePolAR:
  begin
    WriteLn;
    WriteString('Numero do UCEP:');
    ReadInt(n);
    WriteLn;
    WriteString('Agente:');
    ReadInt(Agent);
    SDU_PolicyARead.Agent :=Agent ;
    WriteLn;
    WriteString('Senha de Autorizacao:');
    ReadString(Autorization);
    SDU_PolicyARead.Autorization :=Autorization;
    output UCEP[n].PolicyAReadReq(SDU_PolicyARead);
  end;

   from POLAR

      when  UCEP[NCEP].PolicyAReadConf(SDU_PolicyARead)

      to REPOUSO_USER
      name ConfPolAR:
      begin
        WriteLn;
        WriteString('Status do Gerente =');
        WriteInt(SDU_PolicyARead.Status);
        WriteLn;
        WriteString('Tabela de Diretrizes:');
        WriteLn;
        i:=1;
        while (SDU_PolicyARead.PolicyList[i]<>0)
        do begin
         Policy:=SDU_PolicyARead.PolicyList[i];
         WriteString('DIRETRIZ');
         WriteInt(Policy);
         WriteLn;
         i:=i+1;
        end;
      end;

(******Diretriz MLoad******)


 from REPOUSO_USER
   when USR.POL_MLOAD
   to POLML
  name PedePolML:
  begin
    WriteLn;
    WriteString('Numero do UCEP:');
    ReadInt(n);
    WriteLn;
    WriteString('Gerente:');
    ReadInt(Manager);
    SDU_PolicyMLoad.Manager :=Manager ;
    WriteLn;
    WriteString('Senha de Autorizacao:');
    ReadString(Autorization);
    SDU_PolicyMLoad.Autorization :=Autorization;
```

```
        output UCEP[n].PolicyMLoadReq(SDU_PolicyMLoad);
      end;

      from POLML

          when  UCEP[NCEP].PolicyMLoadConf(SDU_PolicyMLoad)

          to REPOUSO_USER
          name ConfPolML:
          begin
            WriteLn;
            WriteString('Status do Gerente =');
            WriteInt(SDU_PolicyMLoad.Status);
            WriteLn;
            WriteString('Tabela de Diretrizes:');
            WriteLn;
             Policy:=SDU_PolicyMLoad.PolicyList[1];
             WriteString('DIRETRIZ');
             WriteInt(Policy);
             WriteLn;
          end;

(*****Diretriz AChange******)

   from REPOUSO_USER
     when USR.POL_ACHANGE
      to POLAC
     name PedePolAC:
     begin
       WriteLn;
       WriteString('Numero do UCEP:');
       ReadInt(n);
       WriteLn;
       WriteString('Agente:');
       ReadInt(Agent);
       SDU_PolicyAChange.Agent :=Agent ;
       WriteLn;
       WriteString('Senha de Autorizacao:');
       ReadString(Autorization);
       SDU_PolicyAChange.Autorization :=Autorization;
       WriteString('Diretriz:');
       ReadInt(Policy);
       WriteLn;
       SDU_PolicyAChange.PolicyList[1]:= Policy;
       WriteLn;
       output UCEP[n].PolicyAChangeReq(SDU_PolicyAChange);

     end;

     from POLAC

          when  UCEP[NCEP].PolicyAChangeConf(SDU_PolicyAChange)

          to REPOUSO_USER
          name ConfPolAC:
          begin
            WriteLn;
            WriteString('Status do Gerente =');
            WriteInt(SDU_PolicyAChange.Status);
            WriteLn;
          end;

  end; (* de USER_BODY *)
```

```
(**********************************************************)
(*                                                        *)
(*                MODULO DE SIMULACAO CMISE               *)
(*                                                        *)
(**********************************************************)

module CMISE_TYPE activity;
  ip MCEP : array [1..quant_MCEP] of CMISE_interface(provider);
end;

body CMISE_BODY for CMISE_TYPE;

{ Convencoes A_Associate e A_Release:
                       AEI => Nome do destino chamado
                       API => Nome do destino origem

M_Get M_Set e M_Actio:  SC => Nome do destino chamado
                       SY => Nome do destino origem  }

    var
     PDU_A_Associate: A_ASSOCIATE_TYPE ;
     PDU_M_Get: M_GET_TYPE ;
     PDU_A_Release: A_RELEASE_TYPE ;
     PDU_M_Set: M_SET_TYPE ;
     PDU_M_Action: M_ACTION_TYPE ;

     Destiny: integer ;

     initialize  {Inicia com nome dos gerentes ligados ao MCEP}
     begin
     end ;

    trans
    (* comportamento do modulo CMISE  *)

(******************** CMISE Iniciador *********************)

{********************** A-Associate ***********************}

     when MCEP[1].A_AssociateReq(PDU_A_Associate)
     name CMISEA_Associate1Req:
     begin
       Destiny := PDU_A_Associate.AEI ;
       output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
     end;

     when MCEP[2].A_AssociateReq(PDU_A_Associate)
     name CMISEA_Associate2Req:
     begin
       Destiny := PDU_A_Associate.AEI ;
       output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
     end;

     when MCEP[3].A_AssociateReq(PDU_A_Associate)
     name CMISEA_Associate3Req:
     begin
       Destiny := PDU_A_Associate.AEI ;
       output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
     end;

     when MCEP[4].A_AssociateReq(PDU_A_Associate)
```

```
      name CMISEA_Associate4Req:
      begin
        Destiny := PDU_A_Associate.AEI ;
        output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
      end;

      when MCEP[5].A_AssociateReq(PDU_A_Associate)
      name CMISEA_Associate5Req:
      begin
        Destiny := PDU_A_Associate.AEI ;
        output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
      end;

      when MCEP[6].A_AssociateReq(PDU_A_Associate)
      name CMISEA_Associate6Req:
      begin
        Destiny := PDU_A_Associate.AEI ;
        output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
      end;

      when MCEP[7].A_AssociateReq(PDU_A_Associate)
      name CMISEA_Associate7Req:
      begin
        Destiny := PDU_A_Associate.AEI ;
        output MCEP[Destiny].A_AssociateInd(PDU_A_Associate) ;
      end;

{********************* A-Release *****************************}

      when MCEP[1].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release1Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;

      when MCEP[2].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release2Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;

      when MCEP[3].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release3Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;

      when MCEP[4].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release4Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;

      when MCEP[5].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release5Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;
```

```
      when MCEP[6].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release6Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;

      when MCEP[7].A_ReleaseReq(PDU_A_Release)
      name CMISEA_Release7Req:
      begin
        Destiny := PDU_A_Release.AEI ;
        output MCEP[Destiny].A_ReleaseInd(PDU_A_Release);
      end;

{************************** M-Get **************************}

      when MCEP[1].M_GetReq(PDU_M_Get)
      name CMISEM_Get1Req:
      begin
           Destiny := PDU_M_Get.SC ;
           output MCEP[Destiny].M_GetInd(PDU_M_Get);
      end;

      when MCEP[2].M_GetReq(PDU_M_Get)
      name CMISEM_Get2Req:
      begin
           Destiny := PDU_M_Get.SC ;
           output MCEP[Destiny].M_GetInd(PDU_M_Get);
      end;

      when MCEP[3].M_GetReq(PDU_M_Get)
      name CMISEM_Get3Req:
      begin
           Destiny := PDU_M_Get.SC ;
           output MCEP[Destiny].M_GetInd(PDU_M_Get);
      end;

      when MCEP[4].M_GetReq(PDU_M_Get)
      name CMISEM_Get4Req:
      begin
           Destiny := PDU_M_Get.SC ;
           output MCEP[Destiny].M_GetInd(PDU_M_Get);
      end;

      when MCEP[5].M_GetReq(PDU_M_Get)
      name CMISEM_Get5Req:
      begin
           Destiny := PDU_M_Get.SC ;
           output MCEP[Destiny].M_GetInd(PDU_M_Get);
      end;

      when MCEP[6].M_GetReq(PDU_M_Get)
      name CMISEM_Get6Req:
      begin
           Destiny := PDU_M_Get.SC ;
           output MCEP[Destiny].M_GetInd(PDU_M_Get);
      end;

      when MCEP[7].M_GetReq(PDU_M_Get)
      name CMISEM_Get7Req:
      begin
           Destiny := PDU_M_Get.SC ;
```

```
        output MCEP[Destiny].M_GetInd(PDU_M_Get);
    end;

{************************** M-Set **************************}

    when MCEP[1].M_SetReq(PDU_M_Set)
    name CMISEM_Set1Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

    when MCEP[2].M_SetReq(PDU_M_Set)
    name CMISEM_Set2Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

    when MCEP[3].M_SetReq(PDU_M_Set)
    name CMISEM_Set3Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

    when MCEP[4].M_SetReq(PDU_M_Set)
    name CMISEM_Set4Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

    when MCEP[5].M_SetReq(PDU_M_Set)
    name CMISEM_Set5Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

    when MCEP[6].M_SetReq(PDU_M_Set)
    name CMISEM_Set6Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

    when MCEP[7].M_SetReq(PDU_M_Set)
    name CMISEM_Set7Req:
    begin
        Destiny := PDU_M_Set.SC ;
        output MCEP[Destiny].M_SetInd(PDU_M_Set);
    end;

{************************ M-Action ************************}

    when MCEP[1].M_ActionReq(PDU_M_Action)
    name CMISEM_Action1Req:
    begin
        Destiny := PDU_M_Action.SC ;
        output MCEP[Destiny].M_ActionInd(PDU_M_Action);
    end;
```

```
      when MCEP[2].M_ActionReq(PDU_M_Action)
      name CMISEM_Action2Req:
      begin
           Destiny := PDU_M_Action.SC ;
           output MCEP[Destiny].M_ActionInd(PDU_M_Action);
      end;

      when MCEP[3].M_ActionReq(PDU_M_Action)
      name CMISEM_Action3Req:
      begin
           Destiny := PDU_M_Action.SC ;
           output MCEP[Destiny].M_ActionInd(PDU_M_Action);
      end;

      when MCEP[4].M_ActionReq(PDU_M_Action)
      name CMISEM_Action4Req:
      begin
           Destiny := PDU_M_Action.SC ;
           output MCEP[Destiny].M_ActionInd(PDU_M_Action);
      end;

      when MCEP[5].M_ActionReq(PDU_M_Action)
      name CMISEM_Action5Req:
      begin
           Destiny := PDU_M_Action.SC ;
           output MCEP[Destiny].M_ActionInd(PDU_M_Action);
      end;

      when MCEP[6].M_ActionReq(PDU_M_Action)
      name CMISEM_Action6Req:
      begin
           Destiny := PDU_M_Action.SC ;
           output MCEP[Destiny].M_ActionInd(PDU_M_Action);
      end;

      when MCEP[7].M_ActionReq(PDU_M_Action)
      name CMISEM_Action7Req:
      begin
           Destiny := PDU_M_Action.SC ;
           output MCEP[Destiny].M_ActionInd(PDU_M_Action);
      end;


(******************** CMISE Respondedor ********************)

{********************** A-Associate ************************}

      when MCEP[1].A_AssociateResp(PDU_A_Associate)
      name CMISEA_Associate1Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;

      when MCEP[2].A_AssociateResp(PDU_A_Associate)
      name CMISEA_Associate2Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;

      when MCEP[3].A_AssociateResp(PDU_A_Associate)
```

```
      name CMISEA_Associate3Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;

      when MCEP[4].A_AssociateResp(PDU_A_Associate)
      name CMISEA_Associate4Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;

      when MCEP[5].A_AssociateResp(PDU_A_Associate)
      name CMISEA_Associate5Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;

      when MCEP[6].A_AssociateResp(PDU_A_Associate)
      name CMISEA_Associate6Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;

      when MCEP[7].A_AssociateResp(PDU_A_Associate)
      name CMISEA_Associate7Resp:
      begin
        Destiny := PDU_A_Associate.API ;
        output MCEP[Destiny].A_AssociateConf(PDU_A_Associate) ;
      end;


{************************ M-Get ****************************}

      when MCEP[1].M_GetResp(PDU_M_Get)
      name CMISEM_Get1Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
      end;

      when MCEP[2].M_GetResp(PDU_M_Get)
      name CMISEM_Get2Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
      end;

      when MCEP[3].M_GetResp(PDU_M_Get)
      name CMISEM_Get3Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
      end;

      when MCEP[4].M_GetResp(PDU_M_Get)
      name CMISEM_Get4Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
```

```
      end;

      when MCEP[5].M_GetResp(PDU_M_Get)
      name CMISEM_Get5Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
      end;

      when MCEP[6].M_GetResp(PDU_M_Get)
      name CMISEM_Get6Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
      end;

      when MCEP[7].M_GetResp(PDU_M_Get)
      name CMISEM_Get7Resp:
      begin
        Destiny := PDU_M_Get.SY ;
        output MCEP[Destiny].M_GetConf(PDU_M_Get) ;
      end;


{************************ M-Set ***************************}

      when MCEP[1].M_SetResp(PDU_M_Set)
      name CMISEM_Set1Resp:
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;

      when MCEP[2].M_SetResp(PDU_M_Set)
      name CMISEM_Set2Resp:
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;

      when MCEP[3].M_SetResp(PDU_M_Set)
      name CMISEM_Set3Resp:
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;

      when MCEP[4].M_SetResp(PDU_M_Set)
      name CMISEM_Set4Resp:
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;

      when MCEP[5].M_SetResp(PDU_M_Set)
      name CMISEM_Set5Resp:
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;

      when MCEP[6].M_SetResp(PDU_M_Set)
      name CMISEM_Set6Resp:
```

```
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;

      when MCEP[7].M_SetResp(PDU_M_Set)
      name CMISEM_Set7Resp:
      begin
        Destiny := PDU_M_Set.SY ;
        output MCEP[Destiny].M_SetConf(PDU_M_Set) ;
      end;


{************************ M-Action *************************}

      when MCEP[1].M_ActionResp(PDU_M_Action)
      name CMISEM_Action1Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;

      when MCEP[2].M_ActionResp(PDU_M_Action)
      name CMISEM_Action2Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;

      when MCEP[3].M_ActionResp(PDU_M_Action)
      name CMISEM_Action3Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;

      when MCEP[4].M_ActionResp(PDU_M_Action)
      name CMISEM_Action4Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;

      when MCEP[5].M_ActionResp(PDU_M_Action)
      name CMISEM_Action5Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;

      when MCEP[6].M_ActionResp(PDU_M_Action)
      name CMISEM_Action6Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;

      when MCEP[7].M_ActionResp(PDU_M_Action)
      name CMISEM_Action7Resp:
      begin
        Destiny := PDU_M_Action.SY ;
        output MCEP[Destiny].M_ActionConf(PDU_M_Action) ;
      end;
```

```
   end; (* de CMISE_BODY *)

(**********************************************************)
(*                                                        *)
(*                    MODULO PRINCIPAL                    *)
(*                                                        *)
(**********************************************************)

  modvar

    SGH1: SGH_TYPE;
    SGH2: SGH_TYPE;
    SGH3: SGH_TYPE;
    SGH4: SGH_TYPE;
    SGH5: SGH_TYPE;
    SGH6: SGH_TYPE;
    SGH7: SGH_TYPE;
    SIST_GEREN : USER_TYPE;
    CMISE : CMISE_TYPE;

   initialize (* do sistema de teste SGH *)

    name initsist:
    begin
       init CMISE with CMISE_BODY;
       init SIST_GEREN with USER_BODY;
         init SGH1 with SGH_ENTITY_BODY;
         connect SGH1.MCEP to CMISE.MCEP[1];
         connect SGH1.UCEP to SIST_GEREN.UCEP[1] ;
         init SGH2 with SGH_ENTITY_BODY;
         connect SGH2.MCEP to CMISE.MCEP[2];
         connect SGH2.UCEP to SIST_GEREN.UCEP[2] ;
         init SGH3 with SGH_ENTITY_BODY;
         connect SGH3.MCEP to CMISE.MCEP[3];
         connect SGH3.UCEP to SIST_GEREN.UCEP[3] ;
         init SGH4 with SGH_ENTITY_BODY;
         connect SGH4.MCEP to CMISE.MCEP[4];
         connect SGH4.UCEP to SIST_GEREN.UCEP[4] ;
         init SGH5 with SGH_ENTITY_BODY;
         connect SGH5.MCEP to CMISE.MCEP[5];
         connect SGH5.UCEP to SIST_GEREN.UCEP[5] ;
         init SGH6 with SGH_ENTITY_BODY;
         connect SGH6.MCEP to CMISE.MCEP[6];
         connect SGH6.UCEP to SIST_GEREN.UCEP[6] ;
         init SGH7 with SGH_ENTITY_BODY;
         connect SGH7.MCEP to CMISE.MCEP[7];
         connect SGH7.UCEP to SIST_GEREN.UCEP[7] ;
      end ;

end. (* do programa SGH *)
```

Anexo 2

Relatórios dos Testes de Simulação

A2-1 Teste de Iniciação do Sistema

- Inicia SGH1 como gerente principal (MInitPrin)
- Inicia SGH2 como agente de SGH1 (ManagerInit)
- Inicia SGH3 como agente de SGH2 (ManagerInit)
- Lê Tabela de Agentes de SGH1 (AgentRead)
- Lê Tabela de Gerentes de SGH3 (ManagerRead)

```
===============================================================
     | SISTGEREN[ 0]               | D0axu                  |     0
     |                             |-----------------------|
     |  --> SYS[ 0]                | INICMANPRIN            |
===============================================================
     | SISTGEREN[ 0]               | INICGERPRINbkx         |     0
     |                             |-----------------------|
     |  --> UCEP[ 1]               | MINITPRINREQ           |
===============================================================
     | SGH1[ 0]                    | INICIAGERPRINCane      |     0
     |                             |-----------------------|
     |  --> UCEP[ 0]               | MINITPRINCONF          |
===============================================================
     | SISTGEREN[ 0]               | CONFINICGERPRINbll     |     0
===============================================================
     | SISTGEREN[ 0]               | D1azu                  |     0
     |                             |-----------------------|
     |  --> SYS[ 0]                | INICIAR                |
===============================================================
     | SISTGEREN[ 0]               | INICGERENTEbmb         |     0
     |                             |-----------------------|
     |  --> UCEP[ 2]               | MANAGERINITREQ         |
===============================================================
     | SGH2[ 0]                    | CONEXAOMANAGERINang    |     0
     |                             |-----------------------|
     |  --> MCEP[ 0]               | AASSOCIATEREQ          |
===============================================================
     | CMISE[ 0]                   | CMISEAASSOCIATE2bxi    |     0
     |                             |-----------------------|
     |  --> MCEP[ 0]               | AASSOCIATEIND          |
===============================================================
     | SGH1[ 0]                    | ESTABELECECONEXAaoa    |     0
     |                             |-----------------------|
     |  --> MCEP[ 0]               | AASSOCIATERESP         |
===============================================================
     | CMISE[ 0]                   | CMISEAASSOCIATE1bzy    |     0
     |                             |-----------------------|
     |  --> MCEP[ 1]               | AASSOCIATECONF         |
===============================================================
     | SGH2[ 0]                    | PEDETABLEMMANAGEani    |     0
     |                             |-----------------------|
     |  --> MCEP[ 0]               | MGETREQ                |
```

```
==============================================================
     | CMISE[ 0]                  | CMISEMGET2REQbyk      |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | MGETIND               |
==============================================================
     | SGH1[ 0]                   | ENVIATABLEMMANAGaoe   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | MGETRESP              |
==============================================================
     | CMISE[ 0]                  | CMISEMGET1RESPcam     |    0
     |                            |-----------------------|
     |  --> MCEP[ 1]              | MGETCONF              |
==============================================================
     | SGH2[ 0]                   | PEDETABLEPOLICYMano   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | MGETREQ               |
==============================================================
     | CMISE[ 0]                  | CMISEMGET2REQbyk      |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | MGETIND               |
==============================================================
     | SGH1[ 0]                   | ENVIATABLEPOLICYaok   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | MGETRESP              |
==============================================================
     | CMISE[ 0]                  | CMISEMGET1RESPcam     |    0
     |                            |-----------------------|
     |  --> MCEP[ 1]              | MGETCONF              |
==============================================================
     | SGH2[ 0]                   | DESCONECTAMANAGEanu   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | ARELEASEREQ           |
     |                            |-----------------------|
     |  --> UCEP[ 0]              | MANAGERINITCONF       |
==============================================================
     | SISTGEREN[ 0]              | CONFINICGERbmp        |    0
==============================================================
     | CMISE[ 0]                  | CMISEARELEASE2REbxw   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | ARELEASEIND           |
==============================================================
     | SGH1[ 0]                   | DESCONECTAMANAGEaoq   |    0
==============================================================
     | SISTGEREN[ 0]              | D1azu                 |    0
     |                            |-----------------------|
     |  --> SYS[ 0]               | INICIAR               |
==============================================================
     | SISTGEREN[ 0]              | INICGERENTEbmd        |    0
     |                            |-----------------------|
     |  --> UCEP[ 3]              | MANAGERINITREQ        |
==============================================================
     | SGH3[ 0]                   | CONEXAOMANAGERINang   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | AASSOCIATEREQ         |
==============================================================
     | CMISE[ 0]                  | CMISEAASSOCIATE3bxk   |    0
     |                            |-----------------------|
     |  --> MCEP[ 1]              | AASSOCIATEIND         |
==============================================================
     | SGH2[ 0]                   | ESTABELECECONEXAaoa   |    0
     |                            |-----------------------|
     |  --> MCEP[ 0]              | AASSOCIATERESP        |
==============================================================
```

```
      | CMISE[ 0]                    | CMISEAASSOCIATE2caa    |      0
      |                              |------------------------|
      |  --> MCEP[ 2]                | AASSOCIATECONF         |
=================================================================
      | SGH3[ 0]                     | PEDETABLEMMANAGEani    |      0
      |                              |------------------------|
      |  --> MCEP[ 0]                | MGETREQ                |
=================================================================
      | CMISE[ 0]                    | CMISEMGET3REQbym       |      0
      |                              |------------------------|
      |  --> MCEP[ 1]                | MGETIND                |
=================================================================
      | SGH2[ 0]                     | ENVIATABLEMMANAGaoe    |      0
      |                              |------------------------|
      |  --> MCEP[ 0]                | MGETRESP               |
=================================================================
      | CMISE[ 0]                    | CMISEMGET2RESPcao      |      0
      |                              |------------------------|
      |  --> MCEP[ 2]                | MGETCONF               |
=================================================================
      | SGH3[ 0]                     | PEDETABLEPOLICYMano    |      0
      |                              |------------------------|
      |  --> MCEP[ 0]                | MGETREQ                |
=================================================================
      | CMISE[ 0]                    | CMISEMGET3REQbym       |      0
      |                              |------------------------|
      |  --> MCEP[ 1]                | MGETIND                |
=================================================================
      | SGH2[ 0]                     | ENVIATABLEPOLICYaok    |      0
      |                              |------------------------|
      |  --> MCEP[ 0]                | MGETRESP               |
=================================================================
      | CMISE[ 0]                    | CMISEMGET2RESPcao      |      0
      |                              |------------------------|
      |  --> MCEP[ 2]                | MGETCONF               |
=================================================================
      | SGH3[ 0]                     | DESCONECTAMANAGEanu    |      0
      |                              |------------------------|
      |  --> MCEP[ 0]                | ARELEASEREQ            |
      |                              |------------------------|
      |  --> UCEP[ 0]                | MANAGERINITCONF        |
=================================================================
      | SISTGEREN[ 0]                | CONFINICGERbmr         |      0
=================================================================
      | CMISE[ 0]                    | CMISEARELEASE3REbxy    |      0
      |                              |------------------------|
      |  --> MCEP[ 1]                | ARELEASEIND            |
=================================================================
      | SGH2[ 0]                     | DESCONECTAMANAGEaoq    |      0
=================================================================
      | SISTGEREN[ 0]                | D4bcu                  |      0
      |                              |------------------------|
      |  --> SYS[ 0]                 | LEAGENTE               |
=================================================================
      | SISTGEREN[ 0]                | READAGbqh              |      0
      |                              |------------------------|
      |  --> UCEP[ 1]                | AGENTREADREQ           |
=================================================================
      | SGH1[ 0]                     | ENVIATABLEAAGENTaou    |      0
      |                              |------------------------|
      |  --> UCEP[ 0]                | AGENTREADCONF          |
=================================================================
      | SISTGEREN[ 0]                | CONFLEAGbqv            |      0
```

```
===============================================================
     | SISTGEREN[ 0]               | D3bbu                 |     0
     |                             |-----------------------|
     |  --> SYS[ 0]                | LEGERENTE             |
===============================================================
     | SISTGEREN[ 0]               | LEGERENTEbpj          |     0
     |                             |-----------------------|
     |  --> UCEP[ 3]               | MANAGERREADREQ        |
===============================================================
     | SGH3[ 0]                    | ENVIATABLEMMANAGaos   |     0
     |                             |-----------------------|
     |  --> UCEP[ 0]               | MANAGERREADCONF       |
===============================================================
     | SISTGEREN[ 0]               | CONFLEGERbpx          |     0
```

A2-2 Teste de Criação e Eliminação de um Elemento

- Inicia SGH1 como gerente principal (MInitPrin)
- Cria o elemento SGH2 como agente de SGH1 (AgentCreate)
- Lê Tabela de Agentes de SGH1 (AgentRead)
- Lê Tabela de Gerentes de SGH3 (ManagerRead)
- Elimina o elemento SGH2, agente de SGH1 (AgentElimin)
- Lê Tabela de Agentes de SGH1 (AgentRead)
- Lê Tabela de Gerentes de SGH3 (ManagerRead)

```
===============================================================
     | SISTGEREN[ 0]                | D0baj                 |      0
     |                              |-----------------------|
     |  --> SYS[ 0]                 | INICMANPRIN           |
===============================================================
     | SISTGEREN[ 0]                | INICGERPRINbnm        |      0
     |                              |-----------------------|
     |  --> UCEP[ 1]                | MINITPRINREQ          |
===============================================================
     | SGH1[ 0]                     | INICIAGERPRINCand     |      0
     |                              |-----------------------|
     |  --> UCEP[ 0]                | MINITPRINCONF         |
===============================================================
     | SISTGEREN[ 0]                | CONFINICGERPRINboa    |      0
===============================================================
     | SISTGEREN[ 0]                | D9bkj                 |      0
     |                              |-----------------------|
     |  --> SYS[ 0]                 | CRIAAG                |
===============================================================
     | SISTGEREN[ 0]                | CRIAAGENTEbpq         |      0
     |                              |-----------------------|
     |  --> UCEP[ 1]                | AGENTCREATEREQ        |
===============================================================
     | SGH1[ 0]                     | PEDEAGENTCREATEaph    |      0
     |                              |-----------------------|
     |  --> MCEP[ 0]                | AASSOCIATEREQ         |
===============================================================
     | CMISE[ 0]                    | CMISEAASSOCIATE1bzv   |      0
     |                              |-----------------------|
     |  --> MCEP[ 1]                | AASSOCIATEIND         |
===============================================================
     | SGH2[ 0]                     | ESTABELECECONEXAaqh   |      0
     |                              |-----------------------|
     |  --> MCEP[ 0]                | AASSOCIATERESP        |
===============================================================
     | CMISE[ 0]                    | CMISEAASSOCIATE2ccp   |      0
     |                              |-----------------------|
     |  --> MCEP[ 0]                | AASSOCIATECONF        |
===============================================================
     | SGH1[ 0]                     | ENVIAAGENTCREATEapj   |      0
     |                              |-----------------------|
     |  --> MCEP[ 0]                | MSETREQ               |
===============================================================
     | CMISE[ 0]                    | CMISEMSET1REQcbl      |      0
     |                              |-----------------------|
```

--> MCEP[1]	MSETIND	
SGH2[0] --> MCEP[0]	RECEBETABLEMAGENaqn MSETRESP	0
CMISE[0] --> MCEP[0]	CMISEMSET2RESPcdr MSETCONF	0
SGH1[0] --> MCEP[0]	ENVIATABLEPOLICYapr MSETREQ	0
CMISE[0] --> MCEP[1]	CMISEMSET1REQcbl MSETIND	0
SGH2[0] --> MCEP[0]	RECEBETABLEPOLICaqv MSETRESP	0
CMISE[0] --> MCEP[0]	CMISEMSET2RESPcdr MSETCONF	0
SGH1[0] --> MCEP[0] --> UCEP[0]	DESCONECTAAGENTCapz ARELEASEREQ AGENTCREATECONF	0
SISTGEREN[0]	CONFCRIAAGbqe	0
CMISE[0] --> MCEP[1]	CMISEARELEASE1REcaj ARELEASEIND	0
SGH2[0]	DESCONECTAAGENTCard	0
SISTGEREN[0] --> SYS[0]	D4bfj LEAGENTE	0
SISTGEREN[0] --> UCEP[1]	READAGbsw AGENTREADREQ	0
SGH1[0] --> UCEP[0]	ENVIATABLEAAGENTapf AGENTREADCONF	0
SISTGEREN[0]	CONFLEAGbtk	0
SISTGEREN[0] --> SYS[0]	D3bej LEGERENTE	0
SISTGEREN[0] --> UCEP[2]	LEGERENTEbrw MANAGERREADREQ	0
SGH2[0]	ENVIATABLEMMANAGapd	0

```
      |                            |----------------------|
      |  --> UCEP[ 0]              | MANAGERREADCONF      |
===============================================================
      | SISTGEREN[ 0]              | CONFLEGERbsk         |    0
===============================================================
      | SISTGEREN[ 0]              | D10blj               |    0
      |                            |----------------------|
      |  --> SYS[ 0]               | ELIMAG               |
===============================================================
      | SISTGEREN[ 0]              | ELIMAGENTEbqs        |    0
      |                            |----------------------|
      |  --> UCEP[ 1]              | AGENTELIMINREQ       |
===============================================================
      | SGH1[ 0]                   | CONEXAOAGENTELIMarf  |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | AASSOCIATEREQ        |
===============================================================
      | CMISE[ 0]                  | CMISEAASSOCIATE1bzv  |    0
      |                            |----------------------|
      |  --> MCEP[ 1]              | AASSOCIATEIND        |
===============================================================
      | SGH2[ 0]                   | ESTABELECECONEXAarx  |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | AASSOCIATERESP       |
===============================================================
      | CMISE[ 0]                  | CMISEAASSOCIATE2ccp  |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | AASSOCIATECONF       |
===============================================================
      | SGH1[ 0]                   | ENVIAAELIMINarh      |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | MSETREQ              |
===============================================================
      | CMISE[ 0]                  | CMISEMSET1REQcbl     |    0
      |                            |----------------------|
      |  --> MCEP[ 1]              | MSETIND              |
===============================================================
      | SGH2[ 0]                   | APAGATABLEMAGENTasd  |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | MSETRESP             |
===============================================================
      | CMISE[ 0]                  | CMISEMSET2RESPcdr    |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | MSETCONF             |
===============================================================
      | SGH1[ 0]                   | DESCONECTAAGENTEarp  |    0
      |                            |----------------------|
      |  --> MCEP[ 0]              | ARELEASEREQ          |
      |                            |----------------------|
      |  --> UCEP[ 0]              | AGENTELIMINCONF      |
===============================================================
      | SISTGEREN[ 0]              | CONFELIMAGbrg        |    0
===============================================================
      | CMISE[ 0]                  | CMISEARELEASE1REcaj  |    0
      |                            |----------------------|
      |  --> MCEP[ 1]              | ARELEASEIND          |
===============================================================
      | SGH2[ 0]                   | DESCONECTAAGENTEasl  |    0
===============================================================
      | SISTGEREN[ 0]              | D4bfj                |    0
      |                            |----------------------|
      |  --> SYS[ 0]               | LEAGENTE             |
===============================================================
```

```
     | SISTGEREN[ 0]               | READAGbsw             |      0
     |                             |-----------------------|
     |  --> UCEP[ 1]               | AGENTREADREQ          |
================================================================
     | SGH1[ 0]                    | ENVIATABLEAAGENTapf   |      0
     |                             |-----------------------|
     |  --> UCEP[ 0]               | AGENTREADCONF         |
================================================================
     | SISTGEREN[ 0]               | CONFLEAGbtk           |      0
================================================================
     | SISTGEREN[ 0]               | D3bej                 |      0
     |                             |-----------------------|
     |  --> SYS[ 0]                | LEGERENTE             |
================================================================
     | SISTGEREN[ 0]               | LEGERENTEbrw          |      0
     |                             |-----------------------|
     |  --> UCEP[ 2]               | MANAGERREADREQ        |
================================================================
     | SGH2[ 0]                    | ENVIATABLEMMANAGapd   |      0
     |                             |-----------------------|
     |  --> UCEP[ 0]               | MANAGERREADCONF       |
================================================================
     | SISTGEREN[ 0]               | CONFLEGERbsk          |      0
```

A2-3 Teste de Diretrizes

- Inicia SGH1 como gerente principal (MInitPrin), com valor de diretriz 1
- Inicia SGH2 como agente de SGH1 (Init), com valor de diretriz 2
- Lê diretriz de SGH1 (PolicyMRead)
- Lê diretriz de SGH2 através de SGH1 (PolicyARead)
- Muda diretriz de SGH2 através de SGH1 (PolicyAChange) para valor de diretriz 3
- Lê diretriz de SGH2 (PolicyMRead)
- Atualiza diretriz de SGH2 em relação a SGH1 (PolicyMLoad) para valor de diretriz 1

```
=================================================================
     | SISTGEREN[ 0]              | D0baj                 |      0
     |                            |-----------------------|
     |  --> SYS[ 0]               | INICMANPRIN           |
=================================================================
     | SISTGEREN[ 0]              | INICGERPRINbnm        |      0
     |                            |-----------------------|
     |  --> UCEP[ 1]              | MINITPRINREQ          |
=================================================================
     | SGH1[ 0]                   | INICIAGERPRINCand     |      0
     |                            |-----------------------|
     |  --> UCEP[ 0]              | MINITPRINCONF         |
=================================================================
     | SISTGEREN[ 0]              | CONFINICGERPRINboa    |      0
=================================================================
     | SISTGEREN[ 0]              | D11bbj                |      0
     |                            |-----------------------|
     |  --> SYS[ 0]               | INITTEST              |
=================================================================
     | SISTGEREN[ 0]              | INITTUDObmm           |      0
     |                            |-----------------------|
     |  --> UCEP[ 2]              | INITREQ               |
=================================================================
     | SGH2[ 0]                   | INICIAanb             |      0
     |                            |-----------------------|
     |  --> UCEP[ 0]              | INITCONF              |
=================================================================
     | SISTGEREN[ 0]              | CONFINITbna           |      0
=================================================================
     | SISTGEREN[ 0]              | D5bgj                 |      0
     |                            |-----------------------|
     |  --> SYS[ 0]               | POLMREAD              |
=================================================================
     | SISTGEREN[ 0]              | PEDEPOLMRbva          |      0
     |                            |-----------------------|
     |  --> UCEP[ 1]              | POLICYMREADREQ        |
=================================================================
     | SGH1[ 0]                   | LETABLEPOLICYaup      |      0
     |                            |-----------------------|
     |  --> UCEP[ 0]              | POLICYMREADCONF       |
=================================================================
     | SISTGEREN[ 0]              | CONFPOLMRbvo          |      0
```

SISTGEREN[0] --> SYS[0]	D6bhj POLAREAD	0
SISTGEREN[0] --> UCEP[1]	PEDEPOLARbwc POLICYAREADREQ	0
SGH1[0] --> MCEP[0]	CONEXAOPOLICYAREaur AASSOCIATEREQ	0
CMISE[0] --> MCEP[1]	CMISEAASSOCIATE1bzv AASSOCIATEIND	0
SGH2[0] --> MCEP[0]	ESTABELECECONEXAavj AASSOCIATERESP	0
CMISE[0] --> MCEP[0]	CMISEAASSOCIATE2ccp AASSOCIATECONF	0
SGH1[0] --> MCEP[0]	PEDETABLEPOLICYaut MGETREQ	0
CMISE[0] --> MCEP[1]	CMISEMGET1REQcax MGETIND	0
SGH2[0] --> MCEP[0]	ENVIATABLEPPOLICavp MGETRESP	0
CMISE[0] --> MCEP[0]	CMISEMGET2RESPcdd MGETCONF	0
SGH1[0] --> UCEP[0] --> MCEP[0]	DESCONECTAPOLICYavb POLICYAREADCONF ARELEASEREQ	0
SISTGEREN[0]	CONFPOLARbwq	0
CMISE[0] --> MCEP[1]	CMISEARELEASE1REcaj ARELEASEIND	0
SGH2[0]	DESCONECTAPOLICYavx	0
SISTGEREN[0] --> SYS[0]	D8bjj POLACHANGE	0
SISTGEREN[0] --> UCEP[1]	PEDEPOLACbyg POLICYACHANGEREQ	0

SGH1[0] --> MCEP[0]	CONEXAOPOLICYACHavz AASSOCIATEREQ	0
CMISE[0] --> MCEP[1]	CMISEAASSOCIATE1bzv AASSOCIATEIND	0
SGH2[0] --> MCEP[0]	ESTABELECECONEXAawr AASSOCIATERESP	0
CMISE[0] --> MCEP[0]	CMISEAASSOCIATE2ccp AASSOCIATECONF	0
SGH1[0] --> MCEP[0]	ESCREVETABLEPOLIawb MSETREQ	0
CMISE[0] --> MCEP[1]	CMISEMSET1REQcbl MSETIND	0
SGH2[0] --> MCEP[0]	ENVIATABLEPPOLICawx MSETRESP	0
CMISE[0] --> MCEP[0]	CMISEMSET2RESPcdr MSETCONF	0
SGH1[0] --> UCEP[0] --> MCEP[0]	DESCONECTAPOLICYawj POLICYACHANGECON ARELEASEREQ	0
SISTGEREN[0]	CONFPOLACbyu	0
CMISE[0] --> MCEP[1]	CMISEARELEASE1REcaj ARELEASEIND	0
SGH2[0]	DESCONECTAPOLICYaxf	0
SISTGEREN[0] --> SYS[0]	D7bij POLMLOAD	0
SISTGEREN[0] --> UCEP[2]	PEDEPOLMLbxg POLICYMLOADREQ	0
SGH2[0] --> MCEP[0]	CONEXAOPOLICYMLOaxh AASSOCIATEREQ	0
CMISE[0] --> MCEP[0]	CMISEAASSOCIATE2bzx AASSOCIATEIND	0
SGH1[0]	ESTABELECECONEXAaxz	0

```
      |                            |----------------------|
      |  --> MCEP[ 0]              | AASSOCIATERESP       |
===============================================================
      | CMISE[ 0]                  | CMISEAASSOCIATE1ccn  |     0
      |                            |----------------------|
      |  --> MCEP[ 1]              | AASSOCIATECONF       |
===============================================================
      | SGH2[ 0]                   | PEDETABLEPOLICYaxj   |     0
      |                            |----------------------|
      |  --> MCEP[ 0]              | MGETREQ              |
===============================================================
      | CMISE[ 0]                  | CMISEMGET2REQcaz     |     0
      |                            |----------------------|
      |  --> MCEP[ 0]              | MGETIND              |
===============================================================
      | SGH1[ 0]                   | ENVIATABLEPPOLICayf  |     0
      |                            |----------------------|
      |  --> MCEP[ 0]              | MGETRESP             |
===============================================================
      | CMISE[ 0]                  | CMISEMGET1RESPcdb    |     0
      |                            |----------------------|
      |  --> MCEP[ 1]              | MGETCONF             |
===============================================================
      | SGH2[ 0]                   | DESCONECTAPOLICYaxr  |     0
      |                            |----------------------|
      |  --> UCEP[ 0]              | POLICYMLOADCONF      |
      |                            |----------------------|
      |  --> MCEP[ 0]              | ARELEASEREQ          |
===============================================================
      | SISTGEREN[ 0]              | CONFPOLMLbxu         |     0
```

A2-4 Teste de Delegação de um Elemento

- Inicia SGH1 como gerente, com os agentes SGH2 e SGH3 (Init)
- Inicia SGH4 como gerente, com os agentes SGH5 e SGH6 (Init)
- Delega o agente SGH2 de SGH1 para SGH4 (DelegAct)
- Lê Tabela de Agentes de SGH4 (AgentRead)

SISTGEREN[0] --> SYS[0]	D11bbj INITTEST	0
SISTGEREN[0] --> UCEP[1]	INITTUDObmk INITREQ	0
SGH1[0] --> UCEP[0]	INICIAanb INITCONF	0
SISTGEREN[0]	CONFINITbmy	0
SISTGEREN[0] --> SYS[0]	D11bbj INITTEST	0
SISTGEREN[0] --> UCEP[4]	INITTUDObmq INITREQ	0
SGH4[0] --> UCEP[0]	INICIAanb INITCONF	0
SISTGEREN[0]	CONFINITbne	0
SISTGEREN[0] --> SYS[0]	D2bdj DELEGAR	0
SISTGEREN[0] --> UCEP[1]	MANDADELEGbty DELEGACTREQ	0
SGH1[0] --> MCEP[0]	CONEXAODELEGACTasn AASSOCIATEREQ	0
CMISE[0] --> MCEP[3]	CMISEAASSOCIATE1bzv AASSOCIATEIND	0
SGH4[0] --> MCEP[0]	ESTABELECECONEXAatx AASSOCIATERESP	0
CMISE[0] --> MCEP[0]	CMISEAASSOCIATE4cct AASSOCIATECONF	0

SGH1[0] --> MCEP[0]	SOLICITADELEGACAasp MACTIONREQ	0
CMISE[0] --> MCEP[3]	CMISEMACTION1REQcbz MACTIONIND	0
SGH4[0] --> MCEP[0]	ESPERADESCONEXAOaud MACTIONRESP	0
CMISE[0] --> MCEP[0]	CMISEMACTION4RESce j MACTIONCONF	0
SGH1[0] --> MCEP[0] --> UCEP[0]	DESCONECTADELEGAasx ARELEASEREQ DELEGACTCONF	0
SISTGEREN[0]	CONFDELEGbum	0
CMISE[0] --> MCEP[3]	CMISEARELEASE1REcaj ARELEASEIND	0
SGH4[0]	DESCONECTADELEGAaun	0
SISTGEREN[0] --> SYS[0]	D4bfj LEAGENTE	0
SISTGEREN[0] --> UCEP[4]	READAGbtc AGENTREADREQ	0
SGH4[0] --> UCEP[0]	ENVIATABLEAAGENTapf AGENTREADCONF	0
SISTGEREN[0]	CONFLEAGbtq	0