Gestão de Redes
e
Sistemas Distribuídos

Folhas de apoio

Conceitos fundamentais
Arquitecturas de Gestão
Ferramentas de Gestão
Gestão de Serviço

Teresa Maria Vazão

IST/2001

2ª versão

Introdução

Durante o primeiro ano de funcionamento da cadeira de Gestão de Rede e Sistemas Distribuídos fui-me deparando com diversos tipos de dificuldades nos alunos. Entre elas realço a dificuldade que os alunos sentiam em transportar os conhecimentos que lhes estavam a ser leccionados para a realidade. Com o objectivo de clarificar os conceitos de Gestão de Redes e de Sistemas Distribuídos foi produzida a primeira versão deste texto. A sua aceitação junto dos alunos, conduziu a que fossem realizadas algumas alterações no texto original e inseridos novos temas de bastante actualidade.

Manteve-se no entanto a mesma estrutura da edição anterior. A escrita é bastante informal, para tornar a leitura mais fácil. Os vários assuntos que são abordados ao longo do semestre apresentam-se organizados por aulas. No início de cada aula, a seguir ao título, encontra-se uma lista de palavras chave que representam os principais conceitos que irão ser descritos. No final de cada capítulo encontra-se uma lista de sugestões de leitura, onde os assuntos abordados poderão ser estudados em mais detalhe, ou com mais formalismo.

Ao longo de cada capítulo, são-vos sugeridos alguns assuntos para reflexão. Embora as respostas se possam encontrar, directamente, no decorrer do texto, sugiro que pensem neles por vocês e que tentem responder às questões que vos são levantadas, sem recorrer á tentação de espreitar o que vem a seguir !! Seguindo esta abordagem ao problema, por certo demoram mais tempo, aprendem mais e estimulam o vosso sentido crítico e criatividade.

Agradeço todos os comentários que possam fazer para a melhoria da qualidade deste texto.

Lisboa, Julho 2001

Teresa Maria Vazão

Módulo I

Conceitos fundamentais

Capítulo I - Enquadramento e estruturação da gestão

Palavras chave: Processamento Cooperativo, Gestão Integrada. Áreas Funcionais, Desempenho, Contabilidade, Faltas, Configuração, Contabilidade, Segurança, Pirâmide de Gestão, Nível de Elemento de Rede, Nível de Rede e Sistema, Nível de Serviço e Nível de Negócio, Qualidade de Serviço, *Service Level Agreements.*

1 Introdução

Durante este curso irão ser abordados diversos aspectos da Gestão de Redes e dos Sistemas Distribuídos, tendo em consideração a evolução que os conceitos associados a esta temática tem sofrido.

É importante começar por identificar como surgiu a necessidade de gestão e delinear como é quer esta necessidade evoluiu no sentido da Gestão Integrada das Redes, Serviços e Aplicações.

Posteriormente, vão ser analisadas três situações distintas, de complexidade crescente, com o objectivo de identificar tarefas de gestão associadas a cada uma delas. A identificação de tarefas comuns conduzirá à estruturação da gestão em Áreas Funcionais. Como esta estruturação não é suficiente para caracterizar a gestão será também introduzido o Modelo de Responsabilidade ou Pirâmide de Gestão.

2 Processamento Cooperativo e Gestão Integrada

Nesta aula, antes de se iniciar o estudo sistemático de gestão, vale a pena utilizar algum tempo para responder a algumas questões simples que permitem definir, de uma forma mais clara, qual a importância que este assunto tem na tão divulgada sociedade de informação. Eis então as questões que vos proponho:

- O que é o processamento cooperativo e porque razão tem uma importância tão grande nos dias de hoje ?
- Em que medida este tipo de processamento contribuiu para a evolução dos sistemas em rede ?
- Como se caracterizam os sistemas em rede dos dias de hoje ?
- Em que medida esta caracterização condiciona a evolução do próprio conceito de Gestão ?
- Qual o grau de complexidade que a gestão pode assumir neste cenário ?
- O que fazer então para gerir um sistema em rede ?

Interrompam um pouco a leitura deste texto, para reflectirem sobre estas questões. Não se preocupem, nesta fase, em perceber qual a relação de algumas destas questões com o tema central desta disciplina. Vão ver que no final, tudo se clarificou. Aproveitem então para tomar um café, tirar umas notas e depois continuem esta leitura.

Bom, vamos agora sintetizar as respostas às questões propostas. Espero que tenham acertado !!!!

QUESTÃO 1

Comecemos então por definir o que é o processamento cooperativo. De acordo com [1], o **processamento cooperativo** é uma tarefa executada de forma distribuída, de tal modo que o conjunto das sub-tarefas que a constituem são realizadas em sistemas diferentes, em coordenação, para que determinados objectivos possam ser cumpridos.

É fácil de perceber que este tipo de processamento traz flexibilidade e rentabilização de custos. Num cenário de competição global estes são factores críticos para qualquer empresa.

No entanto, só é possível usufruir destas vantagens porque: (i) as redes tiverem um grande desenvolvimento; (ii) existem novos serviços de comunicações; (iii) houve enormes avanços nas tecnologias de SW e, (iv) a própria visão da organização empresarial se alterou.

QUESTÃO 2

Nos dias de hoje, o processamento cooperativo é a base de funcionamento de numerosas organizações que utilizam as Tecnologias de Informação, sendo a infra-estrutura de comunicação fornecida pelas redes corporativas. Para servir cada vez melhor o processamento cooperativo foi necessário que as redes satisfizessem novos requisitos. Os que a seguir se descrevem são aqueles que são mais relevantes e que condicionaram mais fortemente a evolução das redes.

1. Largura de banda : cada vez mais se torna necessário dispor de mais largura de banda, para suportar tráfego e número de utilizadores crescente.
2. Capacidade de Interligação: é necessário garantir a existência de protocolos de transporte uniformes, que possibilitem a interligação das redes.
3. Integração de voz e dados: também se torna necessário proceder rapidamente à integração das redes de voz e de dados e, consequentemente, dos serviços associados.
4. Serviços acessíveis globalmente: a existência de serviços acessíveis globalmente, através de interfaces, espaço de nomes e de directórios uniformes é fundamental.
5. Segurança: deve-se assegurar a implementação de mecanismos de segurança que protejam o acesso à Internet, redes públicas ou outras redes corporativas.

QUESTÃO 3

Resumindo, os sistemas em rede dos dias de hoje caracterizam-se então pela diversidade, complexidade e possibilidade de interligação, independente da tecnologia. Os serviços são mais diversificados, os meios de transmissão e os equipamentos mais rápidos e flexíveis, o tráfego é mais diversificado e os utilizadores mais numerosos e mais exigentes em termos de qualidade de serviço.

QUESTÃO 4

Num cenário de rede heterogénea, com uma grande diversidade de serviços e aplicações as tarefas de gestão assumem uma dificuldade acrescida,. Por outro lado, também assumem uma importância crescente no contexto do aumento da competitividade. Não é mais é possível efectuar uma gestão individualizada e dedicada a cada tipo de equipamento ou serviço. Quer porque os custos de operação e manutenção se tornam muito elevados, quer ainda porque se perde a visão global da rede, não se podendo assim oferecer um serviço de qualidade ao cliente.

Neste novo contexto, a evolução da gestão aponta no sentido de uma **Gestão Integrada**, que englobe os vários tipos de equipamentos e serviços e as diversas vertentes da gestão.

QUESTÃO 5

Numa primeira análise, pode-se concluir que a complexidade da Gestão Integrada está directamente relacionada com a diversidade de tecnologias e serviços existente nos sistemas em rede e com a dimensão do sistema em rede.

Menos evidentes, mas igualmente importantes são todos os aspectos que estão directamente relacionados com as próprias tarefas de gestão. Nesta fase, pouco mais se pode dizer sobre o assunto, porque os vossos conhecimentos ainda agora começaram a ser adquiridos. Vamos esperar pelas próximas aulas e utilizar os exemplos que irão ser dados para aprofundar um pouco mais este tema.

QUESTÃO 6

Claro que nesta fase poucos de vocês saberão responder a esta pergunta. Aliás é esse mesmo o desafio desta disciplina: aprender o que significa gerir um sistema em rede, tanto dum ponto de vista prático, como numa perspectiva mais teórica.

Apesar de tudo isto, uma ideia já vos deve ter ficado na mente: a Gestão serve para garantir o bom funcionamento dos sistemas em rede, deve ser realizada de uma forma Integrada e é uma tarefa complexa. Como tarefa complexa que é deve ser dividida em tarefas mais simples.

Esperemos então pelo desenvolver do curso para saber como se pode dividir a gestão, que tarefas se realizam, quais as ferramentas que temos disponíveis para gerir e quais os princípios teóricos de base.

3 Identificação de tarefas de gestão

De uma forma muito simplista, gerir uma rede ou um sistema distribuído significa garantir que eles funcionem bem. Obviamente, garantir que eles funcionem tem significados diferentes, consoante aquilo que se pretende deles. Por exemplo, para gerir uma impressora ligada a um PC é necessário lidar com muito menos problemas do que para gerir a RNL. A situação complica-se ainda mais se pensarmos numa rede de um fornecedor de serviços Internet. Analisemos então cada uma destas situações isoladamente:

3.1 Case Study I: Impressora ligada a um PC

Possivelmente todos vocês já ligaram uma impressora a um PC. Quais as tarefas de Gestão que estão por trás desta acção ? Configurar a comunicação entre o PC e a impressora; eventualmente, instalar o driver da impressora, após a configuração estar concluída, testar se a comunicação se está a realizar correctamente e por fim, garantir que a comunicação está sempre operacional.

3.2 Case Study II: RNL

Para gerir a RNL a situação complica-se um pouco mais. Comecemos por analisar, as características da RNL. A RNL é composta por um conjunto de laboratórios, equipados, essencialmente, com máquinas LINUX ou WINDOWS, e um conjunto de serviços centrais, tais como servidores WEB, de email ou de ftp, ou ainda computadores onde cada aluno possui a sua própria área de trabalho. O SW em utilização nos equipamentos varia ao longo do ano, de acordo com as disciplinas que o partilham. A população estudantil que tem acesso aos laboratórios e que tem contas abertas também varia de ano para ano, quer em número quer em constituição. Pelo facto dos laboratórios da RNL serem a base de trabalho de alunos de diversas licenciaturas, é necessário garantir o seu correcto funcionamento, se possível 24 horas por dia. Neste contexto, falhas de segurança são particularmente graves, ainda que bastante prováveis dada a frequente utilização da Internet e a heterogeneidade da população estudantil.

Por esta breve descrição é possível antever a variedade de tarefas de gestão que existem por detrás desta estrutura. É preciso configurar equipamentos, configurar a rede, monitorizar e corrigir falhas de forma a garantir a acessibilidade das diversas máquinas, instalar e remover SW, gerir contas de utilizador, implementar protecções contra ataques de segurança e detectar estes ataques.

3.3 Case Study III: REDE de Fornecedor de Serviços INTERNET

No caso desta rede a situação ainda é mais complexa. Primeiro, porque o negócio de um fornecedor de serviços de Internet é exactamente vender serviços de comunicações. Segundo, porque pode existir, por detrás deste negócio, uma hierarquia complexa cliente-fornecedor. Vamos então abordar cada um destes assuntos individualmente.

O facto de se vender um serviço de comunicações introduz dois novos requisitos em termos de gestão: avaliação do serviço e facturação. Na perspectiva da avaliação do serviço é necessário monitorizar o seu desempenho, garantir a existência de uma configuração adequada, detectar e corrigir as falhas atempadamente e avaliar a qualidade de serviço. Por outro lado, a facturação implica todas as tarefas associadas ao processo, tais como, tarifação, taxação, facturação e cobrança.

Também é importante equacionar que, hoje em dia, a venda de qualquer serviço tem sempre por alvo a satisfação do cliente. O mercado das comunicações não é excepção. Neste contexto, os sistemas de gestão orientados para o cliente assumem uma importância particular. Nestes se enquadram os sistemas de gestão de problemas, bem

como a disponibilização de informação de gestão ao cliente. Adicionalmente, existe informação de gestão que pertence exclusivamente ao fornecedor de serviços .

Quando existe uma hierarquia complexa de clientes-fornecedores, algumas tarefas podem ser repartidas com outras empresas. Por exemplo, um fornecedor de serviços Internet pode operar sobre uma infra-estrutura de rede de uma operadora de telecomunicações. Quais serão as tarefas de gestão que, nesta circunstância, não lhe competem ?

4 Estruturação da Gestão em Áreas Funcionais

Vamos tratar de sistematizar o resultado da análise destes exemplos numa tabela. O objectivo é identificar tarefas de gestão que possam ser agrupadas. Se outros casos fossem analisados, era possível que a lista viesse aumentada. No entanto, esta já é suficiente para o que se pretende estudar em seguida.

Antes de continuarem, sugiro uma pausa. Construam a vossa própria lista e depois comparem com a que se segue.

Descrição	I	II	III
Configurar equipamento	✔	✔	✔
Efectuar testes de configuração	✔	✔	✔
Garantir a existência de configuração correcta	✔	✔	✔
Remover e instalar SW	✔	✔	✔
Configurar rede		✔	✔
Detectar e resolver anomalias		✔	✔
Proteger contra ataques de segurança		✔	✔
Detectar ataques de segurança		✔	✔
Gerir contas de utilizador		✔	✔
Taxar e facturar os utilizadores			✔
Monitorizar o desempenho			✔
Avaliar a Qualidade de Serviço			✔

Tabela 1 : Funções de Gestão associadas aos case-studies

Nesta lista, as tarefas encontram-se agrupadas funcionalmente em cinco grupos distintos, que em termos de gestão se chamam de **Áreas Funcionais** (AF). O primeiro grupo corresponde à AF de **Gestão de Configuração**; o segundo à AF de **Gestão de Faltas**; o terceiro à AF de **Gestão de Segurança**; o quarto à AF **de Gestão de Contabilidade e Administração de Utilizador** e, por último, o quinto à AF de **Gestão de Desempenho**. A decomposição em Afs tem como objectivo simplificar a tarefa de gestão, decompondo-a em problema de âmbito mais limitado.

Esta classificação tem origem na Arquitectura de Gestão OSI, que será estudada posteriormente neste curso. Apesar da classificação proposta pelo OSI para a estruturação das Afs não ser aceite de uma forma inquestionável, não existe outra que esteja tão divulgada e que, apesar de tudo, seja tão reconhecida.

Vamos então sistematizar, as funcionalidades atribuídas a cada AF.

4.1 Área Funcional de Gestão de Configuração

A Gestão de Configuração tem três objectivos essenciais: (i) descrever um Sistema Distribuído de acordo com a localização dos seus recursos, (ii) configurá-lo e (iii) e identificar o conjunto de parâmetros que resultam da sua configuração.

Mas afinal de contas, o que é se pode considerar como sendo um recurso de um Sistema Distribuído ? Tudo ! Isto é, os recursos físicos, tais como o meio de transmissão, os componentes de rede, os equipamentos terminais de utilizador, ou recursos lógicos, nos quais se englobam módulos de SW, bases de dados etc…

Agora que a noção de recurso está mais clara, analisemos como é que se podem atingir os três objectivos de gestão, identificados anteriormente. Assim, para descrever um Sistema Distribuído de acordo com a localização dos seus recursos é necessário identificar as relações físicas e lógicas que existem entre estes. Por exemplo, identificar a topologia de uma rede, identificar as relações entre as tabelas que compõem uma base de dados, ou ainda definir acessos possíveis aos vários campos dessa base de dados. Significa então que, estas relações podem estar associadas a aspectos geográficos, organizacionais, de segurança etc..

Em relação ao segundo objectivo, pode-se considerar que configurar um Sistema Distribuído implica activar ou alterar os parâmetros que controlam a operação normal dos seus recursos. Por exemplo, configurar as características da comunicação através de uma porta série de um PC implica definir o ritmo de transmissão, o número de bits de caracteres, a paridade e o número de stop bits. O resultado que se obtém deste processo é exactamente o conjunto de parâmetros referidos no terceiro objectivo.

Dada a diversidade de recursos, o processo de configuração resulta na execução de um conjunto de funções muito variado. Nele se incluem, funções que permitem instalar ou modificar SW, alterar a topologia da rede, alterar a distribuição de tráfego, programar as entradas nas tabelas de encaminhamento e nos servidores de nomes, programar filtros nas pontes, definir a dimensão máxima dos ficheiros e os serviços permitidos a cada utilizador.

Afinal, estas tarefas não são novidade para vocês. Pelo menos algumas das que aqui se indicaram já fazem parte do vosso dia-a-dia. O mesmo se passa em relação à AF de Gestão de Faltas, que vamos analisar em seguida.

4.2 Área Funcional de Gestão de Faltas

A AF de Gestão de Faltas tem como objectivo detectar e resolver as situações de mau-funcionamento que ocorrem nos Sistemas Distribuídos.

A estas situações de mau-funcionamento é usual chamar de faltas. Em termos mais rigorosos, uma falta pode ser entendida como um desvio que as operações dos Sistemas Distribuídos apresentam, relativamente aos objectivos operacionais, funcionais ou de serviço.

Mas afinal, o que pode falhar num Sistema Distribuído ? Mais uma vez, tudo !! Citemos apenas alguns casos, para a lista não ser muito longa: os sistemas de transmissão, os componentes da rede, os sistemas terminais, os módulos de SW, um serviço de segurança ou de transmissão de dados etc…

Conclusão, esta é decididamente uma tarefa difícil ! Vejamos porquê: num Sistema Distribuído, existe uma grande diversidade e quantidade de recursos a gerir, com elevado número de componentes e HW e SW heterogéneos. Estes recursos estão distribuídos fisicamente, podendo ser partilhados por diferente organizações. Se analisarem um pouco sobre este assunto é possível que identifiquem outras causas para tornar esta tarefa complexa. Pensem por exemplo, na diversidade de causas que originam estas faltas e na dificuldade inerente de diagnóstico de erro.

Tal como anteriormente, o processo de Gestão de Faltas é realizado através de um conjunto de funções de âmbito mais limitado. As mais importante são: monitorização de estado, vigilância de alarmes, diagnóstico de faltas, resolução de faltas, realização de testes de diagnóstico, operação de sistemas de Gestão de Problemas (*Trouble Ticket)* e assistência ao cliente (*User Help Desk*).

Nesta altura, algumas destas funções talvez não sejam muito claras para vocês. Brevemente iremos abordar alguma delas e vocês mesmos terão oportunidade de as experimentar no laboratório. Passemos então á próxima AF !

4.3 Área Funcional de Gestão de Desempenho

Fazer com um sistema funcione sem faltas não quer dizer que se garante que o sistema funciona bem. Esta situação é bem ilustrada através da nossa experiência diária com a Internet. Normalmente, consegue-se ir buscar a informação que se pretende sem erros, mas quanto tempo de demora ?

Este tipo de problemas são abrangidos exactamente pela AF de Gestão de Desempenho, cujo objectivo é avaliar a **Qualidade de Serviço** (QoS - Quality of Service).

Esta avaliação efectua-se na interface entre o cliente e o fornecedor do serviço e envolve várias acções, agrupadas por diferentes funções. Assim, na AF de Gestão de Desempenho incluem-se funções que permitem: estabelecer métricas de QoS, monitorizar recursos, realizar medidas e avaliar tendências para antever falhas, manter e analisar ficheiros de *logs* com o histórico do estado do sistema, processar dados e compilar relatórios de desempenho, por último, planear o desempenho e a capacidade do sistema.

Vários problemas existem na Gestão de Desempenho. Iremos falar deles mais detalhadamente na próxima aula. Entretanto, preparem-se para dar a vossa própria opinião…

4.4 Área Funcional de Gestão de Segurança

A AF de Gestão da segurança tem por objectivo detectar, prevenir e resolver todos os problemas associados a violações de segurança dos Sistemas Distribuídos.

Mas afinal, o que é isso de uma violação de segurança ? Não entrando em definições formais, pode-se dizer que uma violação de segurança é uma acção que prejudica, intencionalmente, o funcionamento do sistema sobre o qual foi realizada. Por exemplo, um roubo de *password*, alteração de prioridades, manipulação de recursos, introdução de sobrecarga numa rede etc…

Conforme podem constatar, é uma AF em continua expansão. Quer porque, todos os aspectos dos Sistemas Distribuídos estão sujeitos a violações de segurança: a informação, os serviços e a produção. Quer ainda, porque estas violações afectam fortemente o desempenho das organizações que baseiam os seus resultados no processamento cooperativo.

Neste contexto, no âmbito da AF de Gestão de Segurança é necessário realizar funções que permitam: analisar e localizar tentativas de violação e definir políticas de segurança. Exemplo de procedimentos associados a estas funções são a alteração periódica de *passwords* e a existência de ficheiro de *Logs* com registo das violações de segurança.

Passemos agora à última AF: a AF de Gestão de Contabilidade e Administração de Utilizador. Conforme vão puder constatar, poucos de vós têm experiência prática desta AF, mas muita como clientes ;((!

4.5 Área Funcional de Gestão de Contabilidade e Administração de Utilizador

A Gestão de Contabilidade e Administração de Utilizador tem dois objectivos distintos: (i) administrar nomes e endereços e (ii) gerir contas de utilizador.

A administração de nomes e endereços está associada a um serviço de directórios. Podem surgir algumas dificuldades nesta tarefas, associadas a situações de duplicação, a limitações do próprio espaço de endereçamento ou à existência de sistemas móveis, que necessitem de endereços temporários.

Relativamente à gestão de contas de utilizador, as questões que se colocam são essencialmente de natureza organizacional. Todo o sistema de contabilização tem de ser realizado de acordo com a estrutura e os objectivos da organização em que se insere. Por exemplo, uma empresa com várias dependências, geograficamente distribuídas, pode necessitar de um sistema de contabilização de custos igualmente distribuído, em que cada centro é responsável pelos clientes da sua região geográfica.

A própria estrutura de custos é função da política da empresa. Normalmente engloba dois tipos de custos:

- fixos, que englobam os encargos com o espaço, manutenção e desvalorização de equipamento;
- variáveis, que estão relacionados com a quantidade de tráfego, duração e data (hora e dia) da ligação, largura de banda e QoS, localização dos restantes

intervenientes na comunicação, utilização de serviços e equipamentos, e utilização de produtos de SW (controlo de licenças).

As funções executadas no âmbito da AF de Gestão de Contabilidade e Administração de Utilizador estão associadas ao serviço de directório, à autorização de utilização de recursos, à identificação de custos de utilizador, à tarifação e, por último, à facturação das contas de utilizador.

5 Estruturação da Informação de Gestão em Níveis

5.1 Conceito de Nível de Gestão

As AFs que foram apresentadas serviram para agrupar as tarefas de gestão, consoante as suas características, de forma a realizar o objectivo que foi apresentado no final na aula passada: decompor a gestão em tarefas mais simples. Mesmo assim, a quantidade de funções realizadas em cada AF é muito grande e diversificada. Será que não se pode arranjar outras formas de decomposição de informação, que ajudem a simplificar, ainda mais, a abordagem da gestão ? Antes de terminar esta aula, vamos reflectir um pouco sobre este assunto. Para o fazer, escolhamos uma das Afs e um dos *case-studies* que foram estudados. Aproveitando o aparecimento de novos operadores de telecomunicações em Portugal, analisemos então, à luz desta nova perspectiva, a AF de gestão de desempenho e o *case-study*, referente ao fornecedor de serviços Internet.

Tal como já foi dito anteriormente, o negócio de um fornecedor de serviços Internet é vender serviços de telecomunicações, tendo por base a rede Internet. A oferta incide essencialmente em serviços de transferência de dados, correio electrónico, antevendo-se que, num futuro próximo, venha a oferecer ainda serviços de voz e aplicações multimédia, do tipo video-conferência. Como se trata de um negócio, existem contratos (SLA – **Service Level Agreements**) estabelecidos com os utilizadores, nos quais é possível definir Níveis de Serviço, que representam compromissos de parte-a-parte. Para garantir que os Níveis de Serviço estão a ser cumpridos, é necessário monitorizar o desempenho e detectar situações em que haja violação dos SLAs. É neste contexto que a AF de Gestão de Desempenho se insere.

Consideremos então dois dos parâmetros que usualmente são utilizados para definir os níveis de serviços, nas aplicações de transferência de dados: percentagem de perdas e acessibilidade. A contabilização das perdas efectua-se nos vários elementos que compõem a rede (routers, bridges etc…), por acesso à informação existente nos repositórios de informação de gestão, vulgarmente conhecidos por MIBs (Management Information Base). A contabilização da acessibilidade na Internet, efectua-se entre dois pontos da rede, tipicamente através de pings. Para além de contabilizar e recolher periodicamente a informação a armazenar, do ponto de vista de Gestão de Desempenho, o importante é processar esta informação, de forma a validar os Níveis de Serviço acordados com cada utilizador.

O processamento desta informação de gestão apresenta características diferentes, consoante o perfil da pessoa a quem se destina. No caso de um cliente específico, só lhe interessa a sua própria informação. Porém, para a empresa fornecedora de serviço a situação tem de ser avaliada em função do seu universo de clientes. Ainda na

perspectiva da empresa, a situação terá diferentes abordagens, consoante se trate de um operador da rede ou do gestor da empresa. Tipicamente, o operador quer avaliar tendências de funcionamento, com o objectivo de antever situações que conduzam a um mau desempenho, ou, quando essas situações surgem, descobrir as suas causas. Por seu lado, o gestor quer perceber se o desempenho da rede corresponde a uma aproveitamento convenientemente dos recursos, ou qual o efeito de um mau desempenho nos lucros e na imagem da empresa.

Imagine-se então, que no exemplo em estudo existe um troço de rede que está a funcionar com problemas de transmissão. Na perspectiva de Gestão de Desempenho estes problemas traduzem-se por um aumento gradual da percentagem de perdas. Quais são as diferentes visões que é possível ter em relação a este problema ? Para um dado cliente, este problema só se torna significativo quando o seu Nível de Serviço deixa de ser cumprido. Porém, para a empresa fornecedora de serviço a gravidade da situação tem de ser avaliada em função da quantidade de clientes afectados e do efeito que o problema tem na facturação da empresa. Neste caso concreto, seria de esperar que o operador da rede usasse a informação da existência contínua de perdas num dado troço de rede para averiguar o seu efeito no serviço oferecido e pesquisar a causa do problema em antecipação aos clientes, que não deveriam ser afectados de forma significativa. Nesta situação cabe ao gestor contabilizar os clientes afectados pelas perdas de informação, compensá-los e avaliar os prejuízos sofridos pela empresa.

Este exemplo está muito longe de ser um estudo completo sobre o problema da avaliação de qualidade de serviço na Internet. Serve apenas para ilustrar o facto de que existem diferentes perspectivas sobre o mesmo problema, ou seja, é possível dividir a gestão de uma forma completamente ortogonal às Afs, tendo em consideração os níveis de responsabilidade associados às tarefas de gestão. A esta divisão é usual chamar de **Modelo de Responsabilidade** ou **Pirâmide de Gestão**.

Vamos falar um pouco sobre ela, mas antes disso, está na hora da pausa habitual para o café….

5.2 Modelo de Responsabilidade ou Pirâmide de Gestão

A Pirâmide de Gestão foi um conceito que surgiu na British Telecom e que foi adoptado como fazendo parte do conjunto de normas que descreve a Arquitectura de Rede de Gestão de Telecomunicações, vulgarmente conhecida por Arquitectura TMN (Telecommunication Management Network). Bom, mas isto é assunto para falarmos lá bem mais à frente. Voltemos novamente à nossa pirâmide, que se encontra representada na Figura 1.

De acordo com esta figura, a gestão divide-se em 4 níveis diferentes – **Gestão de Elemento de Rede**, **Gestão de Sistema e de Rede**, **Gestão de Serviço** e **Gestão de Negócio** – com um grau de abstracção crescente, à medida que se sobe na pirâmide.

Este tipo de estruturação baseia-se no princípio de que as funções realizadas num dado nível pressupõem a realização de um conjunto de funções nos níveis inferiores.

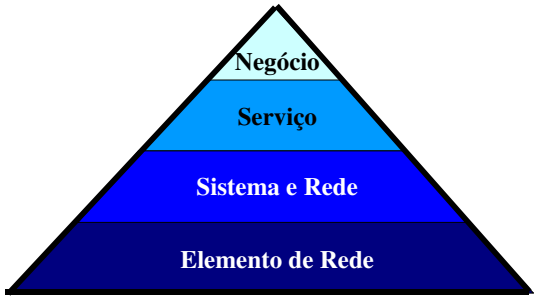


Figura 1: Modelo de Responsabilidade ou Pirâmide de Gestão

Passemos agora a identificação das funções de cada nível. A interface com os recursos a gerir situa-se no nível mais baixo da hierarquia, a Gestão de Elemento de Rede, que engloba a gestão individualizada dos vários elementos que compõem as redes, tais como, equipamentos de transmissão, comutadores ou *routers*. Ao nível de Rede e Sistema a gestão incide sobre a interligação dos equipamentos, nomeadamente nas questões relacionadas com o encaminhamento, o controlo de congestão ou o planeamento de capacidade, contendo ainda as questões relacionadas com os próprios sistemas distribuídos, tais como, gestão de memória e sistemas de ficheiros, distribuição de SW etc.. A Gestão ao Nível de Serviço está relacionada com a oferta de serviço aos Clientes, englobando aspectos relacionados com a avaliação da Qualidade de Serviço, custos envolvidos, oferta e disponibilidade de serviço etc. Por último, no topo da hierarquia situa-se a gestão do negócio, que inclui todos os aspectos relacionados com o investimento, com as perspectivas de mercado, satisfação de cliente etc.

A pirâmide de gestão tem-se revelado um conceito muito útil quando se pretende identificar as funcionalidades de gestão que são relevantes para um dado sistema em rede, do ponto de vista de responsabilidade dos actores do processo de gestão. Uma tarefa deste tipo pode ser realizada com duas perspectivas diferentes, isto é, começando pela base ou pelo topo da pirâmide. Por muito estranho que possa parecer não é indiferente a escolha do método de análise. Se se iniciar pela base vai-se identificar um conjunto de funcionalidades de gestão que existem nos elementos que compõem a infra-estrutura da rede e, a partir daí, derivar as funcionalidades que é possível obter nos níveis mais elevados da hierarquia. Ao invés, se se partir do topo, vão-se identificar as funcionalidades que se pretendem obter nos níveis mais elevados para derivar os requisitos que devem ser satisfeitos nos níveis mais baixos. De uma forma muito simplista pode-se dizer que a grande diferença reside na escolha entre o que se PODE e o que se DEVE ter. As modernas tendências apontam no sentido de se efectuar este tipo de análise a partir do topo.

6 Resumo final

O processamento cooperativo for causado pelo aumento da competitividade entre organizações que conduziu à redução de custos, melhoria de serviço e reutilização de recursos. Neste contexto, os sistemas em rede sofrem uma grande evolução, sendo caracterizados hoje em dia pela diversidade de tecnologias e serviços e pela capacidade de integração.

A necessidade de efectuar uma Gestão Integrada de redes e serviços apareceu também associada ao factor competitividade. Para finalizar, este tipo de gestão é uma tarefa complexa que deve ser repartida em tarefas mais simples.

Os três casos que foram estudados servem para demonstrar dois aspectos distintos da gestão: (i) a diversidade de funções a realizar para gerir um Sistema Distribuído é muito elevada e (ii) existem funções de gestão que se repetem em Sistemas com características e objectivos muito diferentes.

As funções de gestão agrupam-se em conjuntos denominados de Áreas Funcionais. A classificação das funções em Afs que se utilizou foi definida no âmbito da Arquitectura de Gestão OSI e engloba: a Gestão de Configuração, a Gestão de Faltas, a Gestão de Desempenho, a Gestão de Segurança e a Gestão de Contabilidade e Taxação.

Também é possível decompor as funções de gestão de acordo com níveis de responsabilidade. Este tipo de decomposição está associado ao Modelo de Responsabilidade ou Pirâmide de Gestão e engloba quatro níveis: Elemento de Rede, Rede e Sistema, Serviço e Negócio.

Estes dois tipos de estruturação são utilizados hoje em dia no desenho de soluções de gestão para os sistemas em rede. Sendo o objectivo deste curso o estudo da Gestão de Redes e de Sistemas Distribuídos, serão estudados, com particular ênfase, os dois níveis inferiores da pirâmide de Gestão, essencialmente nos aspectos relacionados com a Gestão de Desempenho, de Faltas e de Configuração. Os restantes níveis e AFs serão abordados, de forma não sistemática, sempre que tal se justifique.

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[1]: Capítulo 1 e 3

[4]: Capítulos 1 e 3

Capítulo II - Arquitectura genérica de um sistema de gestão.

Palavras chave: Gestor, Agente, Protocolo de Gestão, MIB, Objecto Gerido, Monitorização, Controlo, Arquitectura de Gestão, Modelo de Informação, Modelo de Comunicação, Modelo de Organização e Modelo Funcional, Gestão de Delegação, Domínios de Gestão

1 Introdução

Foram-vos apresentados alguns conceitos importantes, que vão ser utilizados ao longo deste curso: Gestão Integrada, Áreas Funcionais e Pirâmide de Gestão ou Modelo de Responsabilidade.

Vão-vos ser colocados agora alguns desafios importantes, que vos permitirão perceber como se estrutura um sistema de gestão, em termos de arquitectura geral.

O PRIMEIRO DESAFIO….

O primeiro desafio que vos é colocado consiste em tentarem descobrir como é que deve ser a arquitectura genérica de um sistema de gestão de rede. Eis algumas pistas que vos podem ser úteis: (i) identifiquem os elementos que constituem o sistema de gestão; (ii) definam a interacção entre eles; (iii) identifiquem que tipos de funcionalidades podem executar e que tipo de informação trocam entre si e. Utilizem-nas e cheguem à vossa solução, antes de prosseguirem com a leitura deste texto.

2 Arquitectura genérica de um sistema de gestão

2.1 Elementos que constituem um sistema de gestão

Há 7 elementos principais num sistema de gestão:

- **Gestor(es)**: Um programa de aplicação, ou conjunto de programas, que centraliza(m) a informação de gestão recolhida nos vários componentes a gerir, executa(m) acções de gestão sobre esses componentes e realiza(m) a interface com o operador da rede. O Gestor pode ser dividido em dois módulos diferenciados: o módulo de efectua a interface com o Operador da Rede, **Aplicação de Gestão**, e o módulo que executa as funções de gestão e efectua a interface com o Agente, **Entidade Gestora**.

- **Operador(es) de Rede:** pessoa(s) responsáveis pela gestão da rede, que interage(m) com o sistema de gestão através da aplicação Gestor. As suas funções são monitorizar e controlar o funcionamento da rede.

- **Agente(s)**: Um programa responsável pela recolha local de informação de gestão, de cada recurso a gerir. Normalmente, cada equipamento a gerir tem embebido o seu próprio agente.

- **MIB ou Repositório de Informação de Gestão:** Cada Agente possui um repositório de Informação de Gestão, a MIB. A informação existente na MIB contém a representação dos recursos geridos pelo Agente, na perspectiva da gestão. Isto significa que nem toda a informação existente num dado recurso é

relevante para a gestão, ou seja, só a informação relevante é que é mapeada na MIB.

- **Objecto Gerido:** Cada elemento da MIB chama-se Objecto Gerido (OG). Um OG pode representar um recurso ou um conjunto de recursos. Em termos de realização prática os OGs podem ser implementado através da metodologia de programação orientada-a-objectos ou não, dependendo do tipo de gestão que se está a adoptar.

- **Serviço de Transporte de Informação de Gestão:** Um serviço de transporte de informação de gestão, que suporte a comunicação entre o Gestor e os Agentes. Este serviço é realizado tendo por base um protocolo de gestão, suportado numa dada pilha de protocolos.

- **Recursos geridos:** Conjunto de equipamentos, programas, serviços etc… que estão mapeados numa MIB, isto é, que são representados em termos de OGs.

2.2 Interacção entre os elementos constituintes dum sistema de gestão

As interacções existentes realizam-se entre o Operador de Rede e o Gestor, entre o Gestor e o Agente e entre o Agente e os Recursos Geridos.

- **Operador de Rede ↔ Gestor**: O Operador de Rede interage com o Gestor através de uma interface pessoa-máquina, que pode ser gráfica ou do tipo comando-linha, quer para enviar comandos, quer para receber respostas a esses comandos, quer ainda para receber mensagens emitidas pelo sistema de gestão, constituídas por notificações de alarmes ou de outros tipos de acontecimentos.

- **Gestor ↔ Agente**: A interacção entre o Gestor e o Agente realiza-se através do protocolo de comunicação definido para o efeito, isto é, o Protocolo de Gestão. Cada mensagens transferidas entre o Operador da Rede e o Gestor é convertida numa dada mensagem do protocolo de gestão, utilizando a estrutura pré-definida. Por sua vez, esta mensagem poderá ser novamente convertida, num formato adequado á transmissão pelas camadas inferiores da pilha de protocolos.

- **Agente ↔ Recurso Gerido:** A interacção entre o Agente e os Recursos Geridos efectua-se através dos OGs existentes na MIB. O Agente pode efectuar a leitura de informação nela contida e, nalguns casos, efectuar a sua escrita. Uma operação de leitura corresponde a uma acção de monitorização sobre o recurso físico representado pelo OG e não envolve alterações nos recursos. Uma operação de escrita corresponde a uma acção de configuração do recurso físico representado pelo OG. A alteração do valor é realizada na MIB, cabendo ao SW existente no recurso físico detectar essa alteração e efectuar a alteração de configuração correspondente.

2.3 Funcionalidades dum sistema de gestão

Num sistema de gestão podem ser executadas dois tipos de funcionalidades: Monitorização e Controlo.

- **Monitorização**: A função de monitorização permite ao sistema de gestão observar e analisar o estado e o comportamento do sistema e está associada às Afs de Gestão de Falhas, Desempenho e Contabilidade.

- **Controlo**: A função de controlo permite ao sistema de gestão modificar parâmetros e executar acções e está associada às Afs de Gestão de Configuração e Segurança.

2.4 Informação transferida num sistema de gestão

Basicamente, existem três tipos diferentes de informação que podem ser transferidos: estática, dinâmica e estatística.

- **Informação estática**: que caracteriza a configuração actual do sistema gerido, sendo modificada raramente.

- **Informação dinâmica**: que caracteriza o estado dos vários elementos da rede e contém informação sobre acontecimentos que tenham sido detectados.

- **Informação estatística**: que caracteriza o sistema ao longo do tempo, efectuando medidas agregadas a partir da informação dinâmica.

Existem dois métodos distintos de transferir informação entre o Gestor e os Agentes: o polling e o reporte de acontecimentos (eventos).

- **Polling**: o método de polling funciona com base numa sequência periódica de perguntas (formuladas pelo Gestor) e respostas (devolvidas pelo Agente).

- **Reporte de Eventos**: o método de reporte de eventos é o Agente que tem a iniciativa de enviar uma mensagem ou Gestor, periodicamente ou em consequência de uma dada excepção.

Os dois métodos podem coexistir num sistema de gestão. Porém, pode acontecer que um deles seja predominante, relativamente ao outro.

2.5 Visão geral da arquitectura de um sistema de gestão

Na Figura 2 sumariza-se a visão geral da arquitectura de um sistema de gestão.

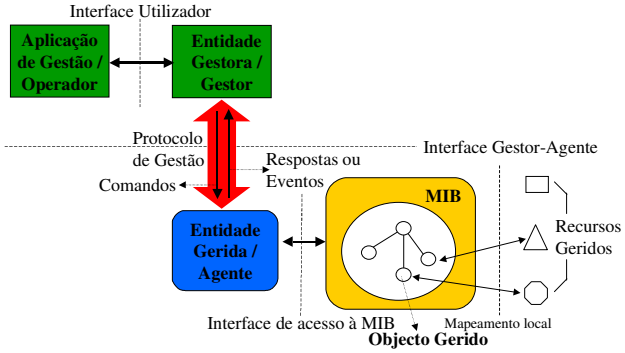


Figura 2: Modelo de um sistema de gestão genérico

O SEGUNDO DESAFIO….

Já ficaram com uma ideia da arquitectura genérica de um sistema de gestão. O desafio que vos é colocado agora é o de analisar, mais em detalhe, o caso da monitorização. O mais relevante neste contexto é identificar para cada AF o tipo de informação que deve ser recolhido e o tipo de processamento a realizar. Utilizem estas sugestões para construirem a vossa própria solução e completem-na de acordo com o texto que se segue. Relativamente ao controlo, fica reservado para as sugestões de leitura !!

3 Função de Monitorização

3.1 AF de Gestão de Desempenho

Na AF de Gestão de Desempenho a Monitorização recolhe um conjunto variável de indicadores. Os mais relevantes são: Disponibilidade, Tempo de Resposta, Precisão, Débito e Utilização.

- **Disponibilidade**: representa a percentagem de tempo que um sistema, componente ou aplicação está acessível ao utilizador. Este indicador está relacionado com a própria fiabilidade dos componentes da rede e com a própria organização dos sistema, ou seja, depende das falhas que ocorram e do tempo que elas demoram a ser reparadas. É um indicador muito importante, na medida em que tem impacto directo na percepção que o utilizador tem do funcionamento do sistema.

- **Tempo de Resposta**: representa o tempo que um sistema demora a reagir a uma dada entrada ou executar uma dada tarefa. Um valor aceitável para este indicador depende muito do tipo de aplicação que se está a considerar. Tal como o anterior, o Tempo de Resposta também é muito importante na gestão pelo impacto que tem na percepção que o utilizador tem do funcionamento do sistema.

- **Precisão**: representa a percentagem de tempo em que não ocorrem erros na transmissão e entrega de informação. Pelo facto de existirem mecanismos de retransmissão, embebidos nos protocolos este indicador não é relevante para o utilizador, mas dá uma ideia do estado do sistema de transmissão, apontando possíveis fontes de ruído ou linhas em falha. Daí a sua importância.

- **Débito**: representa o ritmo de ocorrência de acontecimentos relativos a aplicações, tais como, nº de transacções de um determinado tipo durante um dado período de tempo, nº de chamadas activas numa rede de circuitos virtuais etc..

- **Utilização**: representa a percentagem de tempo que um recurso está a ser utilizado durante um dado período. Este indicador é normalmente utilizado na detecção de situações de congestão.

Estes indicadores são obtidos pelos Agentes e registados nas MIBs. Posteriormente são utilizados para obter medidas de desempenho, através da realização de estatísticas. Essas medidas estão submetidas a programas específicos, que se encarregam de efectuar a sua análise.

3.2 AF de Gestão de Faltas

Relativamente à AF de Gestão de Faltas, não existem indicadores diferenciados para representar uma falha, porque estas podem ser de natureza muito diversa.

Um sistema de Monitorização de Falhas deve ser capaz de detectar, reportar e diagnosticar as falhas. Na sua forma mais simples, o Agente mantêm um ficheiro de logs onde regista os eventos e os erros e o Gestor, periodicamente, efectua o polling ao Agente para pedir esses logs. Numa outra versão, o Agente detecta que determinadas situações acontecem, quando certos valores de limiar são atingidos, e envia um reporte de evento (alarme) ou Gestor, contendo a descrição da situação. A melhor solução consiste na possibilidade do Agente efectuar testes locais, isolar o erro e reportar ao Gestor o erro e o diagnóstico efectuado.

Na grande maioria das situações a dificuldade da Monitorização da Falhas está associada ao diagnóstico, porque as falhas têm causas múltiplas, existem muitas observações diferentes relacionadas com a mesma falha e porque, muitas vezes, existem procedimento locais de recuperação que podem afectar o diagnóstico da falha.

3.3 AF de Gestão de Contabilidade

Na AF de Gestão de Contabilidade a Monitorização recolhe um conjunto de indicadores, que variam com os requisitos da organização, pública ou privada, de grandes ou pequenas dimensões. Apesar desta variedade, é comum a manutenção, por cada cliente de um conjunto de informação que inclui: identificação do utilizador e do destinatário, o volume de informação transferido, nível de segurança e recursos utilizados.

Os procedimentos a realizar também dependem da organização.

Sugiro a habitual pausa para café, antes de entrarmos no último desafio de hoje.

O TERCEIRO (E ÚLTIMO) DESAFIO (POR AGORA)…

Já se analisou a arquitectura genérica de um sistema de gestão, considerando até, mais detalhadamente, o caso da função de Monitorização. Porém, dada a enorme diversidade de redes, serviços e aplicações existente, é de esperar que esta arquitectura genérica seja utilizada com diferentes variantes. O problema destas variantes reside na necessidade de partilhar informação e conhecimento, isto é, é preciso existir alguma normalização. Surge então um conceito novo, que é fundamental para o bom funcionamento da gestão, o conceito de **Arquitectura de Gestão**.

O que é isso de uma arquitectura de gestão ? Não é mais do que um conjunto de normas que indicam como é que a gestão se deve realizar. Então, qual é o desafio seguinte ? Pensar na arquitectura de gestão genérica que foi definida e identificar os

aspectos que devem ser objecto de normalização. Atenção, não se pretende que definam vocês a norma.: isso já foi feito por diversas organizações, conforme vocês terão oportunidade de vir a saber !!!

4 Arquitectura de Gestão

Numa Arquitectura de Gestão, existem quatro aspectos diferentes que devem ser objecto de normalização. Analisando a figura 2, existem dois aspectos fundamentais que facilmente se conseguem identificar, como fazendo parte deste conjunto: os aspectos relacionados com a comunicação entre o gestor e o Agente e os aspectos relacionados com a representação da informação de gestão. Pensando um pouco mais em detalhe, pode-se também concluir que a normalização das funções de gestão também pode ser considerada, na medida em que facilita a interacção entre sistemas de fabricantes diferentes. Talvez o aspecto menos evidente seja aquele que está relacionado com a normalização organizacional em que se definem métodos para estruturar organizações, na perspectiva de gestão.

Na literatura, estes aspectos são referenciados pelo nome de Modelos, existindo assim numa Arquitectura de Gestão: um **Modelo de Comunicação**, **um Modelo de Informação**, um **Modelo Funcional** e um **Modelo Organizativo**.

4.1 Modelo de Informação

O Modelo de Informação contém a definição da estrutura dos Objectos Geridos que irão constituir a MIB. Na sua forma mais genérica, adaptada á metodologia orientada-a-objectos, cada OG é descrito por um conjunto de parâmetros que inclui: a identificação do objecto, as suas características, as operações que pode executar, os eventos que pode gerar e as relações com outros objectos. Em Modelos de Informação mais simples, cada OG é representado apenas por um parâmetro ou conjunto de parâmetros, agrupados sob a forma de tabela.

Por exemplo, se se considerar um OG que representa um router, que tipo de informação se pode associar a cada parâmetro ? Uma solução possível é a que se representa em seguida, mas pode haver outras ! Qual é a vossa ?

Nome	**Router**	**Vossa solução….**
Identificação	Endereço IP de uma das interfaces do router	
Características	Estatícas : localização Dinâmicas: estado, número de pacotes enviados, erros etc…	
Operações	Reset	
Eventos	Alarmes: erros em excesso Eventos: alteração de estado	
Relações com outros OGs		

Tabela 2: Modelo de Informação para a representação de um *router*

4.2 Modelo de Comunicação

No Modelo de Comunicação definem-se os aspectos da comunicação entre o Agente e o Gestor. Esta definição inclui:

- A selecção entre polling ou reporte de eventos como método de transferência de informação preferencial (ao único)

- A definição do Serviço e Protocolo de Comunicação de Gestão a utilizar entre o Agente e o Gestor. Neste contexto são definidas a sintaxe e a semântica das estruturas de dados de comunicação e é incluído o Protocolo de Gestão na Arquitectura de Protocolos de Comunicação.

4.3 Modelo Funcional

O Modelo Funcional não está presente em todas as Arquitecturas de Gestão. A sua especificação inclui a definição exacta das funções de gestão a realizar em cada AF. Por definição exacta entende-se a especificação de um modelo por cada função, que inclua uma descrição funcional e uma API que possibilite o acesso à função. Existem no entanto, especificações de funções menos detalhadas. A definição dos OGs necessários a cada função depende da estrutura do Modelo de Informação. Com base nestas especificações é possível criar bibliotecas de funções de gestão, que podem ser partilhadas por diversos Agentes num contexto de **Gestão por Delegação.**

4.4 Modelo de Organização

O Modelo de organização também não está presente em todas as Arquitecturas de Gestão. Através deste modelo estabelecem-se estruturas que permitem enquadrar o sistema de gestão na organização. A sua definição inclui:

- A definição do tipo de organização do sistema de gestão: hierárquico, centralizado, multiparto etc…

- A definição do tipo de cooperação: gestor-agente ou entidades pares com tarefas flexíveis e recíprocas.

- A definição da estrutura da rede a gerir, em termos de **Domínios de Gestão**, que podem ser organizados numa perspectiva física, lógica, organizativa, ou de policiamento.

5 Resumo final

A arquitectura genérica da gestão baseia-se nos princípios da arquitectura cliente-servidor. Em termos genéricos, existe um Gestor, que centraliza a informação e interage com o operador da rede, um conjunto de Agentes, que recolhem informação de gestão dos sistemas geridos e Repositórios de Informação de Gestão, vulgarmente designados por MIBs. A troca de informação de gestão está associada a procedimentos de monitorização e controlo sobre os recursos geridos, realizando-se através de protocolos de gestão normalizado que efectuam a comunicação entre o Gestor e os Agentes.

A informação de gestão pode ser de transferida preferencialmente pelo método de polling ou de reporte de eventos. A informação transferida e as funcionalidades associadas dependem das AFs, podendo no dividir-se em informação estática, dinâmica e estatística.

Para que seja possível a coexistência de produtos de diferentes fabricantes os sistemas de gestão genéricos devem ser objecto de normalização em algumas das suas características. Esta normalização chama-se Arquitectura de Gestão englobando os seguintes aspectos: modelo de informação, modelo de comunicação, modelo funcional e modelo de organização.

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

1]: Capítulo 3 e 4

[3]: Capítulos 1 e 2

[4]: Capítulo 1

Módulo II

Arquitecturas de Gestão

Capítulo I - Arquitectura de Gestão Internet: SNMPv1

Palavras chave: ARI, MIB, ASN.1 BER, Objectos escalares. Tabelas, GetRequest, GetNextRequest, SetRequest, Trap. MIB-II

1 Introdução

A INTRODUÇÃO TEM DE SER ADAPTADA, PORQUE A ESTRUTURA DO CURD FOI ALTERADA NOVAMENTE !

2 Evolução histórica do SNMP

2.1 O seu aparecimento

Pode-se considerar que na fase inicial do desenvolvimento da arquitectura TCP/IP não existia qualquer tipo de gestão. Nesta altura, todo o trabalho de investigação e desenvolvimento se concentrava no desenho e realização de uma rede de pacotes capaz de suportar a partilha de recursos e a inter-operação. Neste contexto, uma arquitectura de rede que teve a sua génese nos finais da década de 60, através da rede ARPANet, chega aos finais de década de 70, praticamente, sem capacidades de gestão.

Durante esta fase, pode-se considerar que a única ferramenta de gestão disponível era o **Internet Control Message Protocol** (ICMP), que permitia a transferência de mensagens de controlo entre Forwarders e hosts. Através das mensagens de Echo e Reply era possível monitorizar a acessibilidade e através das mensagens de Time-stamp e Time-stamp Reply podia medir-se o atraso.

O programa Ping é baseado neste protocolo e ainda hoje se utiliza com o objectivo de detectar situações de degradação de desempenho ou de congestão, ou ainda de isolar áreas da rede em que se detecte um mau funcionamento, etc…

O crescimento da Internet , que se começou fazer sentir no início dos anos 80, fez com que o número de hosts e de sub-redes aumentasse muito rapidamente. Neste novo contexto tornou-se evidente a necessidade de incluir ferramentas de gestão, por um lado, com outras funcionalidade que o Ping não oferecia, e por outro lado, com mecanismos de comunicação normalizados, que pudessem ser utilizados em diferentes situações.

Em 1987, surge então, um primeiro protocolo, **o Simple Gateway Management Protocol** (SGMP), cujo objectivo era realizar a monitorização de gateways. Rapidamente se compreendeu que a tarefa de gestão tinha de se estender o outro tipo de equipamentos e assim surgem três novas propostas de protocolos de gestão: o **High Level Entity Management System** (HLMS), que era uma generalização do Host Management Protocol; o **Simple Network Management Protocol** (SNMP), que era uma versão melhorada do SGMP e o **Common Management Information Protocol** over TCP/IP que representava uma tentativa de incorporar a arquitectura de gestão OSI , na arquitectura de rede TCP/IP.

Pela sua simplicidade, o SNMP foi escolhido como solução de gestão de curto prazo e o CMOT, pela sua ligação à arquitectura OSI, foi escolhido como solução de longo prazo. Pensava-se na época (1988) que o TCP/IP tinha os dias contados e que, naturalmente, as redes tenderiam a adoptar a para a arquitectura de protocolos OSI.

Para reforçar esta teoria, o IAB também definiu que o modelo de informação a utilizar pelos dois protocolos deveria se comum. Neste contexto, estando a diferença centrada no protocolo de comunicação e no software de suporte, a transição seria mais fácil. Rapidamente se chegou a conclusão que utilizar o modelo de informação OSI, na gestão SNMP era impraticável: neste modelo, a representação da informação de gestão segue a metodologia orientada-a-objectos; cada objecto gerido tem um conjunto de atributos, operações e notificações associadas; definem-se propriedades entre objectos que são especificas da metodologia orientada-a-objectos, nomeadamente a herança, o polimorfismo e o alomorfismos. Isto implica um grau de complexidade que não se adequada a um desenvolvimento rápido, como se pretendia realizar com o SNMP. As duas soluções começam então a divergir.

2.2 O aparecimento do SNMPv1

Em 1990 surge então a primeira versão do SNMP, através da publicação de um conjunto de RFCs onde se definia o modelo de organização, de comunicação e de informação. Neste contexto, foi especificado que a Gestão SNMP se baseava numa arquitectura cliente-servidor, na qual o cliente era designado por Gestor e o servidor por Agente. A função do Gestor era recolher informação de gestão dos Agentes, efectuando o polling. Pelo facto do SNMP correr sobre UDP/IP, esta recolha envolvia um processo de comunicação sem ligação. A informação de gestão a recolher em cada gente, estava organizada numa Management Information Base (MIB). Em oposição à Gestão OSI, o modelo de informação da MIB era caracterizado pela simplicidade: não existia realmente o conceito de objecto, mas sim variáveis simples, a que estavam associadas algumas características básicas, como o tipo e os atributos de leitura escrita.

O sucesso do SNMPv1 traduziu-se pela existência de um número crescente de fabricantes que passaram a incluir facilidades de Gestão SNMP nos seus equipamentos, através do desenvolvimento de MIBs e de Agentes. Em reflexo desta situação surgiram inúmeros RFCs que se destinavam a definir MIBs para suporte de diferentes tipos de equipamentos e tecnologias.

2.3 O aparecimento do SNMPv2

Apesar do grande sucesso do SNMP, a sua introdução no mercado veio demonstrar que o protocolo apresentava limitações quando se pretendia utilizar na gestão de redes complexas, quer pela sua dificuldade de suportar a comunicação com um grande número de agentes, quer pela dificuldade de realizar a transferência de grandes volumes de informação, ou ainda de realizar comunicações seguras. Esta última limitação, apresentava proporções tão graves que fez com que alguns fabricantes não implementassem a mensagem que possibilitava a realização de acções de controlo, o SET. Desta forma, o SNMP ficava reduzido à capacidade de monitorização.

Neste novo contexto, duas situações poderiam surgir: ou se efectuava a transição, inicialmente prevista para o CMOT, ou se tentava melhorar o SNMP.

A primeira solução era, na altura, inviável: as normas OSI para gestão de redes tinham demorado mais tempo do que o previsto a serem desenvolvidas e, consequentemente, não existiam produtos de gestão OSI disponíveis no mercado. Para reforçar esta situação, o modelo de informação que tinha acabado por ser adoptado para a gestão SNMP era totalmente diferente do modelo de informação OSI, o que dificultava fortemente a transição entre as duas arquitecturas.

Neste contexto, começam a surgir novas versões do protocolo SNMP. A primeira versão, denominada de SNMP seguro, surgiu em 1992 e tinha como objectivo resolver os problemas de segurança do SNMP.

Como nesta nova versão, não existia qualquer solução para as limitações de desempenho referidas anteriormente, foram desenvolvidas respostas para estas limitações, ainda em 1992, no âmbito de uma outra propostas, conhecida como **Simple Management Protocol** (SMP). Existiam quatro objectivos básicos no SMP:

- Facilitar a transferência de informação com recursos arbitrários e não apenas recursos de rede, estendendo a comunicação a outros tipos de agentes e a dois gestores.
- Possibilitar a transferência de informação em bloco.
- Utilizar os mecanismos de segurança e de privacidade desenvolvidos pelo SNMP seguro.
- Ser compatível com o SNMP, podendo correr numa arquitectura TCP/IP ou OSI.

Depois da publicação dos dois protocolos, foi decidido que deveria ser desenvolvida uma única versão do SNMP que englobasse os aspectos de segurança e de funcionalidade. O SMP é então adoptado como base para a concepção de um novo SNMP, o SNMPv2. O primeiro conjunto de RFCs foi publicado em 1993.

O enorme sucesso da primeira versão do SNMP, o SNMPv1, deu origem a que a nova versão tivesse uma reduzida aceitação no mercado. Os vendedores encontraram assim pouca motivação para a migração para a nova versão e, nomeadamente, os novos mecanismos de segurança limitaram ainda mais a divulgação desta nova versão, dada a sua extensão e complexidade.

Durante um processo de revisão do SNMPv2, que ocorreu durante 1994-1995, as tentativas de simplificação destes mecanismos foram abortadas, por falta de consenso e eles foram abandonados. O SNMPv2 com mecanismos de segurança embebidos passou a chamar-se de SNMPv2p e a versão sem segurança SNMPv2c.

2.4 O aparecimento do SNMPv3

A falta de segurança é uma grande deficiência das versões de SNMP existentes no mercado. No sentido de remediar esta situação, nos últimos anos têm surgido novas

tentativas de incorporar mecanismos de segurança no SNMP, que se traduzem nos esforços de produção de uma nova versão, deste protocolo, o SNMPv3.

Para além dos aspectos de segurança, o SNMPv3 inclui mecanismos de compatibilização com as versões anteriores do protocolo.

2.5 Monitorização Remota RMON

Com a MIB-II, os gestores das redes podem obter informação sobre cada elemento da rede, individualmente, através da informação medida à entrada e à saída desse elemento. Porém, não é possível obter informações sobre a rede, como um todo. Este tipo de informação pode ser obtido através de monitores de rede, que também se denominam de sondas. O monitor funciona como um espião que inspecciona cada pacote que circula na rede e partir dessa informação, produz informações estatísticas. Adicionalmente pode armazenar os pacotes, ou parte deles. Também pode utilizar filtros que limitam o número de pacotes contados ou capturados.

A especificação mais importante que foi adicionada às recomendações SNMP foi a especificação de Monitorização Remota, RMON-I ou RMON-II, através da qual se introduz a possibilidade de gestão de inter-redes. Basicamente, em RMON define-se uma MIB, que fornece ao gestor informação de inter-funcionamento, sem introduzir qualquer espécie de alteração às especificações existentes.

Com o objectivo de efectuar uma gestão inter-rede, deverá existir um monitor RMON por sub-rede. O monitor pode ser um dispositivo isolado, responsável pela captura e análise de tráfego, ou pode ser executado por um dispositivo, que para além destas funções é responsável por outras.

Para efeitos de gestão estes monitores comunicam com um gestor central, denominado de monitor remoto.

3 Arquitectura de Gestão SNMPv1

3.1 O Modelo de Organização

O SNMPv1 baseia-se numa arquitectura Cliente-Servidor, em que a estação gestora, vulgarmente conhecida por Gestor, é o Cliente e o Agente é o Servidor.

O Gestor é o responsável por todas as aplicações e pela interface com o utilizador, mantendo uma base de dados com toda a informação relevante sobre as redes que gere.

O Gestor comunica com o Agente para aceder à informação de gestão, utilizando um protocolo de comunicação, o SNMP. O Agente possui a informação de Gestão agrupada numa MIB, que caracteriza o comportamento dos objectos geridos. No contexto Internet, os objectos geridos não são realmente objectos, mas sim variáveis, uma vez que não possuem propriedades do tipo herança e encapsulamento. Assim sendo, as variáveis de Gestão Internet podem-se comparar aos atributos dos objectos da Gestão OSI.

No âmbito de uma acção de monitorização desencadeada pelo Gestor, o Agente pode ler a informação da MIB. No âmbito de uma acção de controlo, o Agente pode escrever informação na MIB, alterando desta forma o comportamento dos recursos que está a ser gerido.

As MIBs Internet definem a estrutura, o significado e o método de identificação dos objectos geridos que podem fazer parte da MIB de um Agente. A MIB do Agente contém apenas os objectos geridos que são suportados pelo nó que está a ser gerido. Por exemplo, uma MIB dum Router suporta o conjunto de objectos responsáveis pela gestão do protocolo IP, enquanto que uma MIB duma bridge não suporta este tipo de objectos. De uma forma simples pode-se dizer que, a MIB Internet define os "tipos de objectos" e a MIB do Agente define as entidades objecto que são suportadas por uma dado Agente na realidade.

Os objectos da MIB Internet estão estruturados numa hierarquia, em árvore: as folhas dessa árvore são os objectos geridos e os outros nós são conjuntos de objectos, agrupados funcionalmente. Estes conjuntos representam, um pouco, o papel das Classes de Objectos na Gestão OSI. Contudo, não têm associado nenhum mecanismo de herança.

Existe um conjunto de objectos que foram normalizados na MIB-II e que devem ser suportados por qualquer Agente. Adicionalmente, os vendedores são livres de definir MIBs proprietárias, para gerir de forma mais eficiente, ou fornecer informação adicional sobre os seus equipamentos.

Em qualquer das situações, a identificação dos objectos é efectuada utilizando a mesma Árvore de Registo que se usa no OSI e no ITU-T. Para tal, foi definida uma sub-árvore, denominada de **Árvore de Registo Internet** (ARI), que se inicia no objecto cujo identificador é "1.3.6.1" ("iso.org.dod.internet"), na qual a responsabilidade de numeração de objectos está atribuída ao IETF. Pelo facto de se utilizar uma Árvore de Registo única, é possível especificar informação multi-fornecedor, de forma não ambígua.

3.2 O Modelo de Informação

3.2.1 Árvore de Registo Internet

Na sua organização original, a ARI possui quatro ramos diferentes, sob os quais de definiam todos os objectos de gestão Internet. São eles:

- **Directory** : o primeiro nó, (1), que está reservado para uso do OSI .
- **Mgmt**, que está reservado para objectos de gestão normalizados, isto é, definidos em RFCs aprovados pelo IAB.
- **Experimental**, que inclui os objectos de gestão, utilizados em experiências que se realizam na Internet.
- **Private**, que inclui os objectos proprietários, definidos pelos fabricantes.

Actualmente mgmt contém um único nó, mib-2, referente à MIB normalizada no RFC 1213 (MIB-II). Extensões a esta MIB são definidas através da adição de sub-árvores. Um exemplo desta situação é o RMON, que é definida como a 16ª sub-árvore da MIB-II.

No terceiro nó (3),. Estes objectos estão normalmente associados a aspectos dependentes da tecnologia (ATM, FDDI, X.25 etc..) e, muito embora sejam fortes candidatos a normalização, não chegam a mudar de ramo na ARI.

Por último, existe actualmente um único ramo a sair da sub-árvore private, que tem ínicio do nó enterprise. Dentro desta sub-árvore existe uma ramo alocada a cada fabricantes de equipamentos.

3.2.2 MIBs proprietárias

As MIBs proprietárias dão aos fabricantes a possibilidade de gerir, através de SNMP, aspectos específicos dos seus produtos, na medida em que são criados objectos geridos que representam estes aspectos e que estes objectos podem ser conhecidos do gestor. Um dos problemas desta aproximação reside exactamente aqui: o gestor só é capaz de gerir o que conhece e "não tem obrigação" de conhecer todas as MIBs proprietárias de todos os fabricantes. Como resolver então estas situação ?

A solução consiste em carregar a estrutura das MIBs privadas no Gestor. Para que tal seja possível os fabricantes incluem uma versão em texto e uma especificação formal das suas MIBs. O Gestor deverá carregar e compilar a especificação formal, incluindo-a na sua biblioteca, de forma a poder gerir a MIB do fabricante. Daí a existência de compiladores e browsers de MIBs, como uma das ferramentas mais comuns no âmbito de gestão de redes Internet.

Subsistem ainda algumas dificuldades neste processo, inerentes ao facto de existirem diversas versões de SNMP, que não são totalmente compatíveis ao nível do Modelo de Informação. Nestas circunstâncias a única solução é a conversão manual !

3.2.3 Estrutura da MIB

Os objectos da MIB Internet estão estruturados numa hierarquia, em árvore: as folhas dessa árvore são os objectos geridos e os outros nós são conjuntos de objectos, agrupados funcionalmente.

Associado a cada tipo de objecto na MIB, existe um identificador do tipo ASN.1 - **OBJECT-IDENTIFIER** - que se utiliza para atribuir um nome e identificar o tipo de objecto. Como o valor associado a OBJECT-IDENTIFIER é definido hierarquicamente, a convenção de nomes também serve para identificar a estruturação dos tipos de objectos.

Analisemos então o exemplo representado na Figura 3, que descreve um estrato do grupo IP da MIB-II. O tipo de objecto ipInReceives é identificado por: 1.3.6.1.2.1.4.3. Com base neste identificador é possível concluir que ipInReceives:

- faz parte da ARI, porque se inicia com 1.3.6.1.

- pertence ao ramo Mgmt, mais precisamente à MIB-II, porque a seguir a 1.3.6.1 possui a sequência .2.1,

- dentro da MIB-II é o terceiro objecto do grupo ip, porque termina a sua identificação coma sequência 3.4.

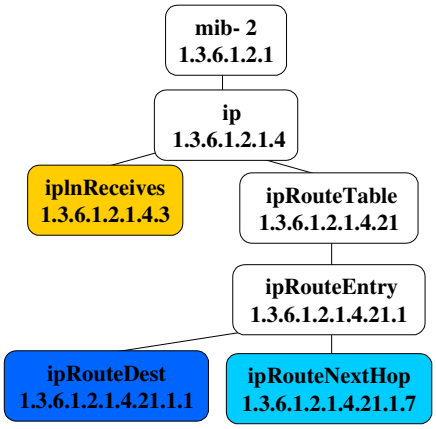


Figura 3: Estrutura de uma parte da MIB-II

3.2.3.1 Sintaxe dos objectos

Cada objecto é definido formalmente de forma a especificar: o tipo de dados, as formas e valores possíveis e a relação com os outros objectos da MIB. Formalmente este objectos, e toda a estrutura da MIB, são definidos em ASN.1. De forma a garantir a simplicidade, apenas um sub-conjunto das funcionalidades ASN.1 são implementadas.

Neste contexto, podem ser utilizados os seguinte tipos de dados:

- Tipos universais – **integer**, **octectstring**, **null**, **object-identifier**, **sequence** e **sequence-of** – que são independentes da aplicação.

- Tipos dependentes da aplicação, que para o caso do SNMP, engloba: **networkaddress**, **ipaddress**, **counter**, **gauge**, **timeticks** e **opaque**. Actualmente, em networkaddress só se pode especificar um ipaddress, que é um campo de 32 bits. Counter representa um contador circular de 32 bits, não negativo. Gauge serve para implementar um contador up-down de 32 bits, que fica bloqueado quando atinge o valor máximo, até que seja feito o reset. Timeticks serve para contar tempo, em centésimos de segundo. Opaque serve para passar dados arbitrários.

Pensem em aplicações concretas para estes tipos de dados.

3.2.3.2 Definição de objectos

Definição de objectos escalares

A MIB define um conjunto de objectos, em que cada objecto tem um tipo e um valor. Cada tipo de objecto está definido sintacticamente e corresponde à representação dum dado tipo de recurso gerido. Quando se instância um recurso gerido, está-se a instanciar um tipo de objecto associando-lhe um valor.

Existe uma macro genérica que se utiliza para definir os tipos de objectos e que se contém, basicamente, a seguinte informação: nome do tipo, sintaxe, tipo de acesso, estado e descrição textual. Por exemplo, no caso do objecto ipInReceives, a macro que o define é a que se representa na Figura 4.

```
ipInReceives OBJECT TYPE
   SYNTAX Counter
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
       ´The total number of input
       datagrams received from
       interfaces… .´
::= {ip 3}
```

Figura 4: Macro de definição do tipo de objecto ipInReceives

O campo do nome identifica o tipo de objecto. No campo sintaxe pode-se utilizar qualquer um dos tipos ASN.1, anteriormente definidos. Em acesso indica-se o tipo de operações que se pode realizar sobre um objecto deste tipo. Os valores possíveis são: **read-only**, **read-write**, **write-only** ou **not-accessible**. Os três primeiros tipos de acesso só são válidos para as folhas da ARI. Todos os outros nós são not.-accessible, o que significa que não se pode realizar nenhuma acção de monitorização ou de controlo sobre eles. O campo status especifica duas coisas diferentes: a obrigatoriedade deste tipo de objecto, se o seu valor for **mandatory** ou **optional,** ou o seu estado desactualizado, se o seu valor for deprecated. Esta indicação serve para referenciar tipos de objectos que só se encontram presentes por razões de compatibilidade com versões anteriores. O campo de descrição é opcional, mas contém informação importante, sobre o que se pretende representar com aquele tipo de objecto.

Definição de tabelas

O Modelo de Informação SMI suporta apenas tabelas bidimensionais, em que as entradas são objectos escalares. A definição de tabelas envolve a utilização dos construtores SEQUENCE e SEQUENCE-OF e a utilização dum novo campo na macro object-type, o campo **IndexPart**.

Consideremos novamente a Figura 3, que para além do objecto escalar ipInReceives, anteriormente descrito, também contém a representação gráfica da tabela ipRouteTable. A tabela é constituída por um conjunto de linhas iguais, sendo uma

dada linha referenciada por ipRouteEntry. Os campos que constituem cada linha, localizam-se debaixo de ipRouteEntry, nas folhas da ARI, e são os únicos objectos da tabela acessíveis para leitura ou escrita. Para identificar cada uma das linhas da tabela, existe um conjunto de campos de pesquisa. No caso concreto existe apenas um destes campos: o ipRouteDest.

A identificação dos tipos de objectos que constituem a tabela realiza-se de forma idêntica à que se utiliza para identificar os tipos de objectos escalares e que foi exemplificada através de ipInReceives.

Antes de exemplificar a macro de definição desta tabela façam uma pausa e construam a vossa própria versão. Utilizem como modelo a macro da Figura 4 e baseiem-se no que anteriormente foi dito.

Comparem então a vossa versão com a que se representa na Figura 5.

```
ipRouteTable OBJECT-TYPE
SYNTAX SEQUENCE OF IpRouteEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
<< Tabela de encaminhamento IP >>
::= { ip 21}
```

```
IpRouteEntry OBJECT-TYPE
SYNTAX IpRouteEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
<< Rota para um dado destino >>
INDEX {ipRouteDest }
::= { ipRouteTable 1}
```

```
IpRouteEntry ::=
SEQUENCE {
ipRouteDest IpAddress, ...
ipRouteNextHop IpAddres, ..
}
```

```
ipRouteDest OBJECT-TYPE
SYNTAX IpAddress
ACCESS read-write
STATUS mandatory
DESCRIPTION
<< Endereço de destino da rota >>
::= {ipRouteEntry 1}
```

```
ipRouteNextHop OBJECT-TYPE
SYNTAX IpAddress
ACCESS read-write
STATUS mandatory
DESCRIPTION
<< Endereço seguinte da rota >>
::= {ipRouteEntry 7}
```

Figura 5: Macro de definição do tipo de objecto ipRouteTable

Analisando a Figura 5, verifica-se que o tipo de objecto ipRouteTable é constituído por ume série de entradas, definidas por SEQUENCE-OF ipRouteEntry. Cada objecto do tipo ipRouteEntry identifica uma linha da tabela, cujo indíce é dado no campo INDEX, como sendo o objecto do tipo ipRouteDest. Cada linha da tabela é constituída por um conjunto de objectos escalares, conjunto esse que é definido através de ipRouteEntry ::= SEQUENCE {…} . Por fim, cada um dos tipos de objectos escalares que constituem a tabela é definido de forma idêntica a ipInReceives.

Resta relembrar que, tal como já tinha sido anteriormente referido, nem a tabela ipRouteTable como um todo, nem as suas linhas ipRouteEntry isoladamente estão

acessíveis para leitura ou escrita. Estas operações só se podem realizar sobre os objectos escalares que a constituem.

Codificação da informação para transferência

Uma arquitectura de comunicação tem dois componentes principais: a componente de transferência de dados- que no caso presente é constituída pela pilha de protocolos UDP/IP/protocolos dependentes da rede – e a componente de aplicação, que neste caso é representada pelo protocolo SNMP. Quando se transita entre estas duas componentes há uma diferença significativa na forma como a mesma informação é analisada: do ponto de vista da comunicação, a informação é constituída apenas por uma sequência de octectos, que é codificada nas SDUs para ser transferida entre níveis adjacentes e em PDUs para ser transferida entre o mesmo nível de dois sistemas diferentes; do ponto de vista da aplicação é constituída por um conjunto de informação estruturada, de forma a ter significado para o utilizador. Esta situação encontra-se representada na Figura 6.

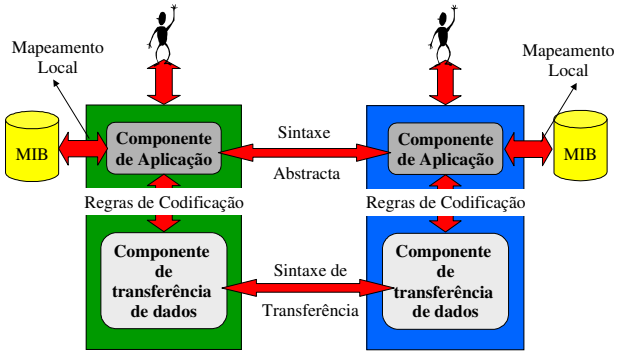


Figura 6: A sintaxe abstracta e a sintaxe de transferência

A informação ao nível da aplicação é então codificada de uma forma abstracta, que lida simultaneamente com tipos de dados e com valores. É através desta notação abstracta que as entidades de nível de aplicação comunicam. No caso concreto, a notação abstracta que se utiliza é a notação ASN.1, que já foi abordada anteriormente. Dentro de cada sistema, a informação pode ser mapeada de uma outra forma que seja adequada ao armazenamento local ou à visualização do operador. Ao nível de transferência pode-se utilizar códigos de representação de caracteres, tais como o ASCII.

A conversão entre os dois formatos efectua-se através de regras de codificação que representam cada valor e cada tipo. Este tipo de codificação tem duas vantagens:

- Existe uma representação comum da informação que é transferida entre os sistemas
- Internamente, cada aplicação pode ter a sua própria representação da informação.

No caso da Arquitectura de Gestão Internet o esquema de codificação utilizado é o **Basic Encoding Rule.** (BER).

Basic Encoding Rule

O esquema de codificação BER atribui um triplo a cada componente ASN.1, constituído por (tipo, dimensão, valor). Este triplo é habitualmente referenciado por TLV (Type, Length, Value).

A representação resumida encontra-se descrita na Figura 7-

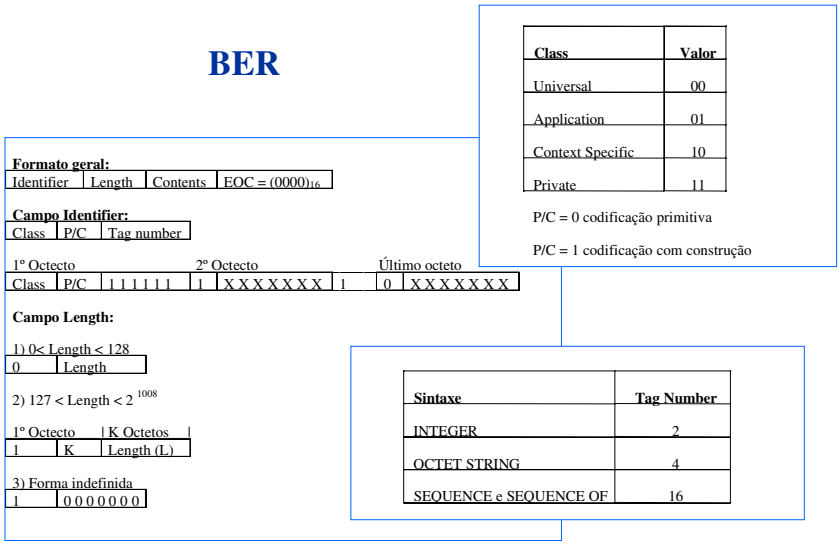


Figura 7: A codificação BER

Efectuemos nova pausa, desta vez para reflectir sobre uma questão essencial neste processo de comunicação: o que é preciso para que duas entidades comuniquem e se entendam ? É uma questão do dia a dia, de resposta simples portanto...

Em primeiro lugar é preciso que conheçam ambas o assunto em discussão, isto é, que utilizem a mesma representação de objectos. Em segundo lugar, é preciso que falem a mesma linguagem, isto é, que utilizem o mesmo protocolo de comunicação.. Ou seja, no contexto da comunicação vai ser necessário definir duas coisas: (i) como se identificam os objectos que representam os recursos geridos; (ii) qual o protocolo de comunicação que se utiliza. Passemos então ao estudo da identificação dos objectos e deixemos os aspectos do protocolo para a próxima aula.

3.3 O Modelo de Comunicação

3.3.1 Identificação de objectos

Conforme foi referido na aula anterior cada tipo de objecto possui um identificador único que se define em função da posição de objecto da ARI. No contexto da comunicação entre um Gestor e um Agente o acesso efectua-se relativamente a um objecto gerido, isto é, a uma instânciação de um dado tipo de objecto. Vamos então

analisar em seguida, como se efectua a identificação de um objecto escalar e de uma tabela.

3.3.1.1 Identificação de objectos escalares

A instanciação de um objecto do tipo á origem a um único objecto. Assim sendo, não há qualquer ambiguidade quando se pretende aceder à instância do objecto, podendo-se incluisivamente, utilizar o identificador do tipo de objecto. Porém, por razões de compatibilidade com as tabelas, optou-se por distinguir as duas representações. Assim sendo:

A representação de uma instância de um objecto escalar é constituída pelo identificador do tipo de objecto, concatenado com um 0.

Analisemos então o exemplo do objecto ipInReceives, representado na figura 16. De acordo com a sua posição na ARI, como o identificador do tipo de objecto vale 1.3.6.1.2.1.4.3, a identificaao da instância valerá 1.3.6.1.2.1.4.3.**0.**

3.3.1.2 Identificação de objectos colunares (elementos de tabelas)

A identificação das instâncias dos objectos das tabelas, normalmente denomindaos de objectos colunares, o processo é mais complicado, uma vez que o tipo de objecto associado dá origem a várias instâncias diferentes: uma tabela é normalmente composta por várias linhas, com formatos idênticos. Isto significa que o identificador que se obtém da ARI não é suficiente para determinar o objecto. O que é que falta então ? Um campo, ou mais campos, de INDEX, que sejam obrigatoriamente diferentes em cada uma das linhas, sendo portanto utilizados para as distinguir. Assim sendo:

A representação de uma instância de um objecto colunar é constituída pelo identificador do tipo de objecto, concatenado o valor dos campos de INDEX da tabela.

Retomemos então o exemplo da tabela ipRouteTable, cuja localização na ARI foi descrita na Figura 3: a tabela é constituída por um conjunto de campos, dos quais se representaram apenas o campo de INDEX, ipRouteDest e um outro campo, ipRouteNextHop. Considere-se ainda a instânciação da tabela descrita da Figura 8.

ipRouteDest	…	ipRouteNextHop	…
9.1.2.3	…	99.0.0.3	
10.0.0.51	…	89.1.1.42	
10.0.0.77	….	89.1.1.42	

Figura 8: Instanciação da tabela ipRouteTable

De acordo com o que anteriormente foi descrito, para referenciar a instancia do objecto ipRouteDest existente na segunda linha da tabela é necessário concatenar o identificador do objecto na ARI (1.3.6.1.2.1.4.21.1) com o valor associado à instância

do campo de INDEX. (10.0.0.51). Ou seja, a instância identifica-se através da seguinte sequência: 1.3.6.1.2.1.4.21.1.1. **10.0.0.51**.

Se se pretender referenciar a instância do outro campo, ipRouteNextHop, a única diferença reside no identificador deste campo na ARI. Uma vez que ele ocupa a posição 7, relativamente a ipRouteEntry, o seu identificador na ARI será dado por.**7**, obtendo-se então a seguinte identificação para a sua instância: 1.3.6.1.2.1.4.21.1.**7**. 10.0.0.51.

É uma representação de díficil leitura e muito sujeita a erros. Vamos então arranjar uma forma mais simples de o fazer. Comecemos então por arranjar uma referência única para cada linha da tabela, isto é, o identificador do tipo de objecto ipRouteEntry.

x = 1.3.6.1.2.1.4.21.1

Para referenciar um tipo de objecto especifico, basta concatenar o seu próprio identificador. Por exemplo, os tipos de objectos anteriormente referidos obtém-se da seguinte forma:

- Tipo: ipRouteDest → x.ipRouteDest ↔ x.1
- Tipo: ipRouteNextHop → x.ipRouteNextHop ↔ x.7

Por último, as instâncias dos objectos obtém-se por concatenação do valor do campo de INDEX, associado à instância, isto é:

- Instância: ipRouteDest → x.ipRouteDest.<valor de ipRouteDest> ↔ x.1.10.0.0.51
- Instância: ipRouteNextHop → x.ipRouteNextHop.<valor de ipRouteDest> ↔ x.7.10.0.0.51

3.3.2 Protocolo de comunicação

3.3.2.1 Tipos e formatos das mensagens SNMP

Para transferir informação de gestão entre um Gestor e um Agente utiliza-se o protocolo de comunicação SNMP, sobre uma piha de protocolos baseada em UDP/IP. Existem cinco tipos diferentes de mensagens SNMPv1 que é possível transferir entre um Gestor e um Agente:

- **GetRequest** e **GetNextRequest** são PDUs geradas pelo Gestor, no âmbito de uma acção de monitorização.
- **SetRequest** é uma PDU gerado pelo Gestor no âmbito de uma acção de controlo a executar sobre o Agente.
- **GetResponse** é uma PDU gerada pelo Agente em resposta a uma acção de monitorização ou controlo, desencadeada pelo Gestor, isto é, em resposta a um GetRequest, GetNextRequest ou SetRequest.
- **Trap PDU** é uma PDU gerada assíncronamente pelo Agente, para reportar eventos ao Gestor.

Conforme se representa na Figura 9, cada mensagem SNMP é constituída por:

- **Version**: número de versão de SNMP utilizada,
- **Community name**: nome de comunidade utilizado para efeitos de controlo de acesso.
- **PDU SNMP**: Unidade de Protocolo de Dados (PDU- Protocol Data Unit), pertencente a um de cinco tipos diferentes: GetRequest, GetNextRequest, SetRequest, GetResponse e Trap.

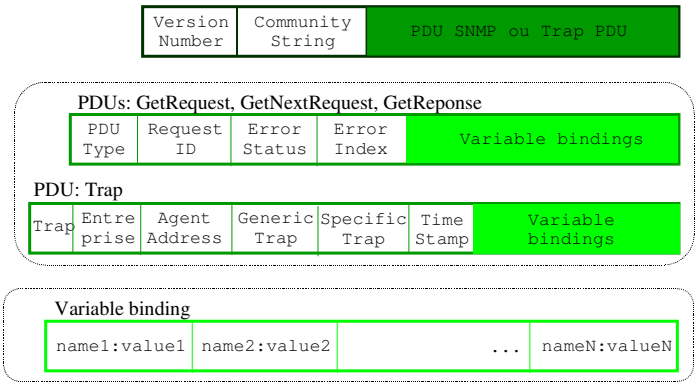


Figura 9: Formato das mensagens SNMPv1

Dos cinco tipos de PDUs utilizadas, apenas a TRAP possui um formato diferente das restantes, nesta versão de SNMP. Em qualquer uma delas existe um campo comun, denominado de variable-binding, onde se coloca a lista de instâncias sobre a qual se pretende realizar a acção definida pelo tipo de PDU e os valores associados.

Relativamente aos tipos GetRequest, GetNextRequest, SetRequest e GetResponse, o significado de cada um dos restantes campos é o que a seguir se indica na Tabela 3.

Nome	Descrição
PDU Type	Tipo de PDU. Valores: GetRequest, GetNextRequest, SetRequest, GetResponse e Trap
Request Id	Identificador do pedido que serve para associar um pedido a uma resposta
Error Status	Indicador de que ocorreu uma excepção no tratamento do pedido. Valores: noError(0), tooBig(1), noSuchName(2), badValue(3), readOnly(4) genError(5)
Error Index	Informação adicional sobre a causa do erro
Enterprise	Tipo de objecto que gerou o TRAP (sysObject)
Agent Address	Endereço do objecto que gerou o TRAP
Generic Trap	Identificador genérico do TRAP Valores: coldStart(0), warmStart(1), linkDown(2), linkUp(3), authenticationFailure(4), egpNeighbourLoss(5), enterpriseSpecific(6)
Specific Trap	Código específico do TRAP
Time Stamp	Tempo entre o último reset ao equipamento e a geração do TRAP (sysUpTime).
Variable Bindings	Lista de instâncias de objectos e valores associados

Tabela 3: Descrição dos campos das SNMP-PDUs

Esta mesma informação encontra-se definida no standard em notação ASN.1, utilizando a representação descrita na Figura 10.

```
RFC 1157-SNMP DEFINITIONS ::= BEGIN
IMPORTS
   ObjectName,ObjectSyntax,NetworkAddress,IPAddress,TimeTicks
   FROM RFC1155-SMI;

Message ::= SEQUENCE {version INTEGER {version-1(0)} - version 1
            community OCTET STRING, - community name
            data ANY       - if trivial authentication is used

PDUs ::= CHOICE {get-request        GetRequest-PDU,
                 get-next-request   GetNextRequest-PDU
                 get-response       GetResponse-PDU
                 set-request        SetRequest-PDU
                 trap               Trap-PDU}

GetRequest-PDU     ::= [0] IMPLICIT PDU
GetNextRequest-PDU ::= [1] IMPLICIT PDU
GetReponse-PDU     ::= [2] IMPLICIT PDU
SetRequest-PDU     ::= [3] IMPLICIT PDU



PDU :== SEQUENCE {
   request-id INTEGER,
   error-status INTEGER {   - sometimes ignored
      noError(0), tooBig(1), noSuchName(2),
      badValue(3), readOnly(4), genErr(5)},
   error-index INTEGER     - sometimes ignored
   variable-bindings VarBindList
}
Trap-PDU ::= [4] IMPLICIT SEQUENCE {
    enterprise OBJECT-IDENTIFIER,
    agent-addr NetworkAddress
    generic-trap INTEGER {
       coldStart(0), warmStart(1), linkDown(2),linkUp(3),
       authenticationFailure(4), egpNeighborLoss(5),
       enterpriseSpecific(6)}
    specific-trap INTEGER,
    time-stamp TimeTicks
    variable-binding VarBindList
}


VarBind ::= SEQUENCE {
    name ObjectName,
    value ObjectSyntax
}
VarBindList ::= SEQUENCE OF VarBind

END
```

Figura 10: Notação ASN.1 para a descrição das mensagens SNMPv1 (RFC 1157)

3.4 O Modelo de Comunicação

3.4.1 Transmissão ou recepção de uma mensagem SNMP

Independentemente do tipo de mensagem SNMP há um conjunto comum de procedimentos a realizar na emissão e na recepção.

3.4.1.1 Procedimentos de Transmissão

1. Construir a PDU utilizando a notação ASN.1

2. Opcionalmente, enviá-la para um serviço de autenticação, juntamente com os endereços de origem e destino e com o nome de comunidade.

3. Construir uma mensagem SNMP, adicionando à PDU (autenticada ou não) o campo de versão e o nome da comunidade.

4. Codificar a PDU, utilizando o esquema de codificaao BER, de forma a que seja enviada para o serviço de transporte.

3.4.1.2 Procedimentos de Recepção

1. Construir a mensagem SNMP em ASN.1 a partir de BER

2. Validar a versão do protocolo SNMP utilizado, e retirar a PDU SNMP da mensagem.

3. Opcionalmente, enviá-la para um serviço de autenticação, juntamente com os endereços de origem e destino e com o nome de comunidade.

   - Se a autenticação falhar a PDU é ignorada e é gerado um TRAP.

   - Se a autenticação não falhar, enviar a PDU em ASN.1. e…

4. Analisar os campos da PDU e se o seu preenchimento estiver correcto….

5. Executar a acção pedida, tendo em conta o perfil de comunidade identificado.

3.4.1.3 Manipulação de objectos

A manipulação de objectos refere-se à execução das operações de monitorização ou de controlo sobre estes objectos.

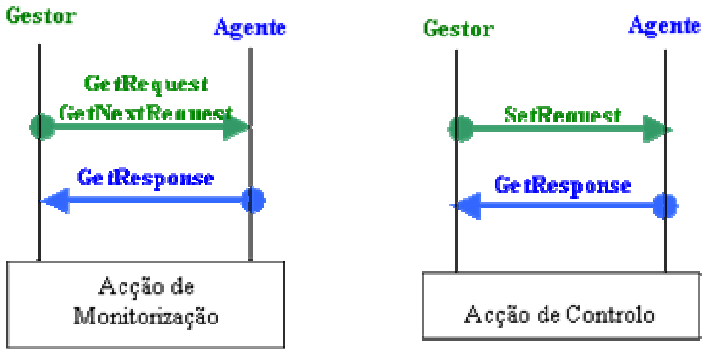


Figura 11: Acções de Monitorização e Controlo em SNMPv1

Conforme se pode analisar na Figura 11, a monitorização pode estar associada à sequência de mensagens GetRequest e GetReponse, ou à sequência de mensagens

GetNextRequest, GetResponse. Na primeira situação o Gestor sabe exactamente o identificador do objecto que pretende manipular, na segunda não sabe.

Acção de Monitorização

Quando o Gestor envia uma mensagem GetRequest deverá preencher o número do pedido (Request ID) e a lista de identificadores de objectos que pretende monitorizar, preenchendo com o valor NULL, os valores associados (variable-binding). Ao receber esta mensagem, depois de efectuar um conjunto de validações, o Agente identifica os objectos definidos no campo variable–binding e lê os valores que lhe estão associados na sua MIB. Com base nos resultados obtidos, constrói uma mensagem SNMP do tipo GetResponse com o mesmo número de pedido, onde inclui no campo variable-binding o valor associado a cada objecto pedido. Caso a mensagem enviada fosse do tipo GetNextRequest, o Agente iria seleccionar o próximo objecto da ARI, efectuando uma análise lexicográfica.

Em qualquer dos casos, a operação realizada é uma **operação atómica**, isto é, só se realiza se o Agente conseguir ler os valores de todos os objectos definidos em variable-binding.

Que situações podem levar a que a operação não se possa realizar ? Por exemplo, se um dos objectos não estiver incluído na visão da MIB, correspondente à comunidade identificada na mensagem, a operação não se pode realizar, sendo devolvida ao Gestor uma indicação de noSuchName.

EXEMPLO:

Exemplifiquemos então a realização de uma acção de monitorização sobre o objecto escalar ipInReceives, utilizando uma linguagem simbólica, onde apenas se inclui a informação relevante:

GetRequest (x.ipInReceive.0)

GetResponse (x.ipInReceives.0 = 25)

Em alternativa o Gestor poderia enviar uma mensagem do tipo:

GetNextRequest (x.ipInReceive)

E o Agente deveria responder com o objecto que se encontra na posição seguinte em termos léxicos, isto é:

GetResponse (x.ipInReceives.0 = 25)

Acção de Controlo

O controlo está associado à sequência de mensagens SetRequest e GetReponse. A diferença relativamente à monitorização reside no facto dos campos que contém os valores dos objectos, na lista de variable-binding serem preenchidos pelo Gestor. Na resposta o Agente deve indicar o valor do campo actualizado.

Tal como em relação às acções de monitorização, o controlo também é uma operação atómica.

EXEMPLO:

Utilizando o exemplo anterior, uma acção de controlo sobre o mesmo objecto nã conduz ao efeito pretendido, uma vez que ipInReceives só é acessível para leitura. Assim sendo, quando o Gestor envia:

SetRequest (x.ipInReceive.0=35)

O Agente devolve uma indicação de erro:

GetResponse (x.ipInReceives.0 = noSuchName)

No caso do campo variable-binding incluir outras variáveis, pelo facto da actualização de ipInReceives não se puder realizar, nenhuma das restantes variáveis é actualizada.

Consideremos agora, um objecto que tenha permissões de leitura/escrita, por exemplo: ipForwarding. A actualização do seu valor corresponde à transferência do seguinte conjunto de mensagens:

SetRequest (x.ipForwarding.0 = 1)

GetResponse (x.ipForwarding.0 = 1)

3.4.1.4 Manipulação de tabelas

Considere-se agora, a situação em que se pretende aceder a uma tabela. O acesso só se pode realizar às células da tabela, uma vez que, são estas que se localizam nas folhas da ARI. O acesso a cada elemento efectua-se com base no seu identificador e no valor dos campos de INDEX correspondentes à sua linha da tabela. Mas estes já foram focados anteriormente, quando se estudou a identificação de instâncias.

Onde reside então a dificuldade de realizar o acesso a tabelas ? A dificuldade reside, essencialmente, no facto de não se saber, à partida, o número de linhas que compõem a tabela, nem tão pouco o valor dos campos de INDEX.

Acção de Monitorização

Assim sendo, a realização de uma acção de monitorização implica a utilização da mensagem SNMP GetNextRequest.

Para se efectuar a leitura de uma tabela completa, o procedimento a realizar é o seguinte:

- Inicialmente, o Gestor envia uma mensagem GetNextRequest, contendo a identificação dos tipos de objectos que se quer aceder: Esta mensagem deverá conter obrigatoriamente os campos de INDEX

- Agente devolve uma mensagem do Tipo GetResponse, contendo os valores dos objectos que em termos léxicos se seguem aos que foram indicados na mensagem enviada pelo Gestor, isto é, os campos respeitantes à primeira linha da tabela.
- Utilizando os valores dos campos de INDEX, que foram devolvidos pelo Agente, o Gestor envia uma nova mensagem GetNextRequest, identificando os objectos correspondentes à 1ª linha.
- Mais uma vez, o Agente devolve uma mensagem do Tipo GetResponse, contendo os valores dos objectos que em termos léxicos se seguem aos que foram indicados na mensagem enviada pelo Gestor. Caso a tabela não tenha acabado, estes objectos são os que se encontram na próxima linha da tabela e possuem o mesmo identificador (de tipo) que os que foram referenciados pelo Gestor. Caso a tabela já tenha acabado, os objectos que se seguem em termos léxicos já não fazem parte da tabela e possuem um identificador (de tipo) diferente daquele que o Gestor enviou.
- Quando o Gestor recebe uma mensagem GetResponse, compara os identificadores (de tipo) que enviou com os que recebeu, para descobrir se a informação recebida ainda faz parte da tabela.

Para o controlo, torna-se necessário saber previamente os valores dos campos de INDEX, correspondentes às linhas em que se encontram as células da tabela que se pretendem alterar. Ou seja, é necessário realizar previamente uma acção de monitorização.

EXEMPLO:

Exemplifiquemos novamente, os conceitos através da tabela ipRoutetable anteriormente descrita, na figura 21. Considere-se então que o Gestor pretende efectuar a leitura da tabela completa, sem saber o seu contéudo e a sua dimensão. Considere-se também que o Agente possui a árvore representada na Figura 12.

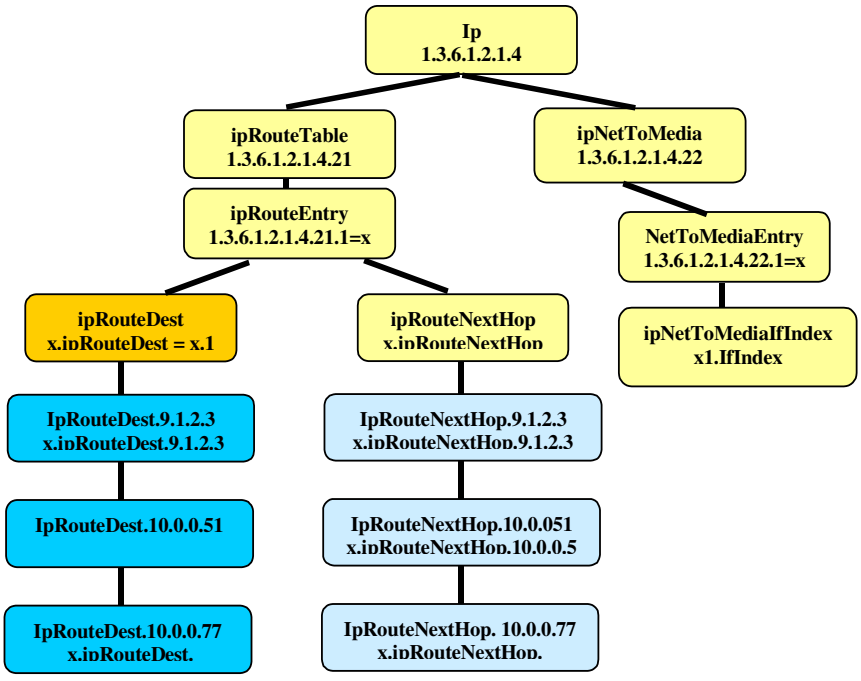


Figura 12: Extracto da árvore existente no Agente

Acção de Controlo

A realização de uma acção de controlo implica a utilização o conhecimento do valor dos campos de INDEX. Assim sendo, o Gestor envia uma mensagem SetRequest, na qual identifica as células que quer actualizar através do campo de INDEX que lhe está associado. Se o Agente conseguir efectuar as actualizações pedidas, devolve o novo valor das células na mensagem GetResponse.

Remoção de linhas

A remoção de linhas não pode ser realizada de forma dinâmica, nesta versão de SNMP. A única acção que se pode realizar é invalidar a linha que se pretende remover, desde que a tabela possua um campo de **Type.** Se for esse o caso, para invalidar a entrada na tabela o Gestor deverá enviar um SetRequest com esse campo a Invalid.

3.5 Estruturação da MIB-II

A MIB-II define uma estrutura de objectos geridos, que deverá ser incluída, pelo menos parcialmente, em qualquer equipamento que suporte SNMP. Os requisitos principais que estiveram na base da sua concepção foram:

- Conter apenas os objectos essenciais às áreas de gestão de faltas e de configuração. Este facto justificará que, opções complicadas, que envolvam por exemplo uma forte componente de segurança, não se possam realizar.

- Não introduzir redundância de objectos, mas também não limitar o número de objectos a definir (como era o caso da MIB-I que estava limitada a 100 objectos). Não considerar no entanto objectos particulares de cada implementação.

De uma forma breve: só existem na MIB-II os objectos que, na época, foram considerados essenciais.

Para simplificar a estruturação da informação, os objectos foram divididos em grupos, de acordo com as suas características funcionais. Consideram-se então 10 grupos diferentes:

- **System**: que contém a informação do sistema gerido
- **Interfaces**: que contém informação de cada interface que o sistema possui em cada sub-rede.
- **At**: que possibilita a tradução de endereços IP em endereços da sub-rede, para cada uma das interfaces.
- **IP**: que contém toda a informação relevante para a gestão, relativamente à implementação e execução do protocolo IP.

- **ICMP**: que contém toda a informação relevante para a gestão, relativamente à implementação e execução do protocolo ICMP.
- **TCP**: que contém toda a informação relevante para a gestão, relativamente à implementação e execução do protocolo TCP.
- **UDP**: que contém toda a informação relevante para a gestão, relativamente à implementação e execução do protocolo UDP.
- **EGP**: que contém toda a informação relevante para a gestão, relativamente à implementação e execução do protocolo EGP.
- **SNMP**: que contém toda a informação relevante para a gestão, relativamente à implementação e execução do protocolo SNMP.
- **Dot3**: que contém toda a informação referente aos mecanismos de transmissão e protocolos de acesso ao meio de cada interface.

Passemos então ao estudo de cada um destes grupos. De forma a efectuar um abordagem sistemática, para cada um dos grupos vai ser feita uma breve descrição do seu conteúdo. Apenas para alguns dos grupos, a sua constituição detalhada pode ser encontrada numa tabela. Esta descrição vai servir de base a um conjunto de exemplos. Sempre que precisem da saber mais detalhes, é só ir estudar os RFCs.

3.5.1 Grupo System

Este grupo fornece informação sobre o sistema gerido, permitindo identificar o tipo de equipamento, a sua localização, o responsável pela sua operação e o tempo que está a funcionar. Conforme se pode observar, por análise da Tabela 4, é um grupo constituído unicamente por variáveis escalares.

Objecto	Acesso	Descrição
sysDescr	RO	Descrição dos sistema: HW, SO etc..
sysObjectID	RO	Identificador que o vendedor atribui ao sistema do ponto de vista de gestão
sysUpTime	RO	Tempo desde que o sistema foi reinicializado
sysContact	RW	Identificação do responsável pelo sistema
sysName	RW	Nome administrativo do sistema
sysLocation	RW	Localização física do sistema
SysServices	RO	Nível de serviços oferecidos pelo sistema (1)

Notas: (1): Por cada camada da pilha de protocolos que seja suportada é colocado um 1 na posição correspondente.

Tabela 4: Constituição do grupo System da MIB-II

EXEMPLO:

Consideremos então o exemplo da máquina Hobbit, onde se encontra instalado o TeMIP. Na perspectiva do grupo system, só interessa saber que a máquina é uma TRU64 Station, com sistema operativo UNIX, fabricada pela Compaq, registada como WS com um dado identificador na parte da ARI atribuída à Compaq, que está na Laboratório X da RNL e que está sobre a responsabilidade da equipa de gestão da

RNL, da Alameda. Com base nesta descrição, o grupo system deve incluir a seguinte informação.

sysDescr = ´HW=TRU64 SO=UNIX´; sysObjectID = <valor obtido da ARI>;

sysUpTime = <Tempo desde último reset >; sysContact = 'RNL-Alameda';

sysName = ´hobbit´; sysLocation = ´lab X´; sysServices = 1001000

3.5.2 Grupo Interfaces

O grupo de interfaces contém informação genérica sobre a interface física do sistema. Este grupo contém informação de configuração e informação de desempenho, constituída por um conjunto de estatísticas relativas a eventos que ocorrem na interface.

O grupo é constituído por uma variável escalar, que indica o número de interfaces existentes no sistema e uma tabela, em que cada linha descreve uma dessas interfaces. A tabela é indexada pelo campo ifIndex.

Objecto	Acesso	Descrição
ifNumber	RO	Número total de interfaces.
ifTable	NA	Tabela de interfaces
ifEntry	NA	Linha da tabela de interfaces
ifIndex	RO	Chave identificadora da interface
ifDescr	RO	Informação sobre a interface: fabricante, nome do produto, versão de HW
ifType	RO	Tipo da interface (definido em função dos protocolos de nível físico).
ifMTU	RO	Tamanho da maior PDU que a interface pode enviar/receber
ifSpeed	RO	Estimativa da capacidade da interface
ifPhyAddress	RO	Endereço de nível físico da interface
ifAdminStatus	RW	Estado da interface: UP (1), DOWN(2) e TESTING(3)
ifOperStatus	RO	Estado operacional da interface: UP (1), DOWN(2) e TESTING(3)
ifLastChange	RO	Valor de sysUpTime em que ocorreu a alteração de estado operacional
ifIn<params>	RO	{ Estatísticas de desempenho de nível 2, à entrada da interface }
ifOut<params>	RO	{ Estatísticas de desempenho de nível 2, à saída da interface }
ifSpecific	RO	Referência à especificação da MIB associada ao tipo específico da interface

Tabela 5: Constituição do grupo Interfaces da MIB-II

3.5.3 Grupo at

O grupo Address Translation fornece conversão de endereços entre endereços de nível de rede e endereços físicos. Este grupo consiste numa única tabela - **atTable** - em que cada linha corresponde a uma interface física do sistema. A tabela é indexada por atIfIndex, cujo valor deve ser coincidente com o valor de ifIndex, que está associado à interface, e por atNetAddress.

Normalmente o endereço de rede é o endereço IP dessa interface do sistema. O endereço de nível físico depende do tipo de tecnologia. Por exemplo, se a interface

estiver associada a uma rede local, o endereço físico é um endereço MAC; se a interface estiver associada a uma rede de comutação de pacotes X.25, o endereço físico é o endereço X.121.

Esta tabela só existe por razões de compatibilidade com a MIB-I. Na MIB-II, a tabela de conversão de endereços está associada a cada grupo de protocolo de nível de rede.

Objecto	Acesso	Descrição
AtTable	NA	Tabela de conversão de endereços
atEntry	NA	Linha da tabela de conversão de endereços
atIfIndex	RW	Índice de identificação da interface
atPhyAddress	RW	Endereço físico
atNetAddress	RW	Endereço de rede

Tabela 6: Constituição do grupo at da MIB-II

3.5.4 Grupo IP

O grupo de ip contém informação sobre a implementação e operação do protocolo IP num dado sistema. Este grupo contém informação de configuração e informação de desempenho, constituída por um conjunto de estatísticas relativas a eventos que ocorrem neste protocolo.

O grupo é constituído por um conjunto de variáveis escalares, que contêm informação de configuração do protocolo e estatísticas de desempenho de pacotes IP e três tabelas:

- a tabela de endereços – **ipAddrTable** – que contém informação de endereçamento de nível de rede que é indexada por uma única chave correspondente ao índice da interface representada nessa linha da tabela;
- a tabela de encaminhamento – **ipRouteTable** – que contém informação genérica sobre o encaminhamento e que é indexada, exclusivamente, pelo endereço de destino.
- a tabela de conversão de endereços – **ipNetToMedia** – que contém a informação de conversão de endereços do protocolo IP, de uma forma semelhante à que foi descrita no grupo at, sendo desta forma endereçada por duas chaves com significados semelhantes.

Objecto	Acesso	Descrição
IpForwarding	RW	Indica se o sistema funciona como gateway (1) ou não (2)
IpDefaultTTL	RW	Valor por omissão do campo Time-to-Live dos pacotes IP.
ipIn<params> [1]	RO	{ Estatísticas de desempenho do protocolo IP à entrada da interface }
ipOut<params> [2]	RO	{ Estatísticas de desempenho do protocolo IP à saída da interface }
ipReasms<params> [3]	RO	{ Estatísticas de desempenho referentes à reassemblagem de pacotes IP}
ipFrags<params> [4]	RO	{ Estatísticas de desempenho referentes à fragmentação de pacotes IP }
IpRoutingDiscards	RO	Estatísticas sobre o número de entradas de encaminhamento descartadas
IpAdTable	NA	Tabela de endereços
IpAdEntry	NA	Informação de endereços referente a uma entidade do sistema
ipAdEntIAddr	RO	Endereço IP da entidade
ipAdEntIfIndex	RO	Índice que identifica univocamente a interface da entidade
ipAdEntNetMask	RO	Máscara associada ao endereço IP da entidade
ipAdEntBCastAddr	RO	LSB do endereço de difusão da interface
ipAdEntReasmMaxSize	RO	Dimensão máxima dos datagramas que recebidos para reassemblagem
ipRouteTable	NA	Tabela de encaminhamento
ipRouteEntry	NA	Informação de endereço referente à rota para um dado destino
ipRouteDest	RW	Endereço de destino
ipRouteIffIndex	RW	Índice da interface associada ao próximo nó (NextHop) para o destino
ipRouteMetric<1 a 4>	RW	Métrica de encaminhamento
ipRouteNextHop	RW	Endereço IP do próximo nó
ipRouteType	RW	Tipo de encaminhamento
ipRouteProto	RO	Identificador do protocolo de encaminhamento
ipRouteAge	RW	Tempo que decorreu desde a última actualização à tabela de encaminhamento
ipRouteMask	RW	Máscara a ser aplicada ao endereço antes da comparação com IpRouteDest
ipRouteInfo	RO	Informação específica do protocolo de encaminhamento (outras MIBs)
ipNetToMedia	NA	Tabela de tradução de endereços
ipNetToMediaEntry	NA	Informação de conversão de endereços referente a um endereço IP
ipNetToMediaIfIndex	RW	Índice da interface associada ao endereço IP
ipNetToMediaPhysAddr	RW	Endereço de nível físico
IpNetToMediaNetAddr	RW	Endereço IP
IpNetToMediaType	RW	Tipo de mapeamento

Notas:
[1] : {ipInReceives, ipInHdrErrors, ipInAddrErrors, ipInForwDatagrams, ipInUnknownProtos, ipInDiscards, ipInDelivers}
[2] : {ipOutRequest, ipOutDiscards, ipOutNoRoutes}
[3] : {ipInReasmTimeOut, ipReasmReqds, ipReasmOk, ipReasmFail}
[4] : {ipFragOk, ipFragFail, ipFragCreates}

Tabela 7: Constituição do grupo IP da MIB-II

Relativamente à tabela de encaminhamento existe dois aspectos extra que vale a pena referir:

1. Diferentes domínios administrativos podem utilizar diferentes protocolos de encaminhamento: em ipRouteTable só existe informação de natureza genérica, que, por esse facto, pode ser encontrada em qualquer um destes protocolos. A informação de detalhe, específica de cada uma deles, pode ser encontrada através

desta tabela, utilizando o apontador para MIBs que contêm informações de protocolos de encaminhamento específicos, nomeadamente Exterior Gateway Protocol (EGP), Hello Protocol, Routing Information Protocol (RIP), Intermediate-System to Intermediate-System Protocol (IS-IS), End-System to Intermediate-System Protocol (ES-IS) etc..

2. A tabela ipRouteTable só tem como chave de pesquisa o endereço de destino. Assim sendo, para um dado destino só é possível escolher uma rota, o que impede, entre outras coisas, o balanceamento de carga. Para resolver estas limitações, esta tabela de encaminhamento foi substituída por **ipForwardTable**, que possui uma chave composta por dois campos: ipForwardDest e ipForwardNextHop. Nesta nova tabela é possível seleccionar rotas por saída e por endereço de destino.

3.5.5 Grupo ICMP

O grupo icmp contém informação sobre a implementação e operação do protocolo ICMP num dado sistema. Este grupo têm de estar incluído em todos os sistemas que possuam o grupo IP, uma vez que os dois protocolos têm de existir simultaneamente num dado sistema. Este grupo contém informação unicamente informação de desempenho, constituída por um conjunto de variáveis escalares, que descrevem estatísticas relativas a eventos que ocorrem neste protocolo.

3.5.6 Grupo TCP

O grupo tcp contém informação genérica sobre a implementação e operação do protocolo TCP num dado sistema. Este grupo contém informação de desempenho, constituída por um conjunto de estatísticas relativas a eventos que ocorrem neste protocolo.

O grupo é constituído por um conjunto de variáveis escalares, que indicam o contém indicações sobre o número de ligações em diversos estados, e uma tabela – **tcpConnTable** -, em que cada linha descreve uma ligação TCP. A tabela é indexada pelos quatro campos que identificam uma ligação TCP: tcpConnLocalAddress, tcpConnLocalPort, tcpConnRemoteAddress, tcpConnRemotePort.

3.5.7 Grupo UPD

O grupo udp contém informação genérica sobre a implementação e operação do protocolo UDP num dado sistema. Este grupo contém informação de desempenho, constituída por um conjunto de estatísticas relativas a eventos que ocorrem neste protocolo.

Para além de estatísticas referentes aos datagramas recebidos e enviados, o grupo contém uma tabela – **udpTable** -, em que cada linha descreve um UDP listener. A tabela é indexada pelos dois campos que o identificam: udpLocalAddress, udpLocalPort.

3.5.8 Grupo EGP

O grupo egp contém informação genérica sobre a implementação e operação do protocolo EGP num dado sistema. Este grupo contém informação de desempenho, constituída por um conjunto de estatísticas relativas a eventos que ocorrem neste protocolo.

Para além de estatísticas referentes às mensagens EGO enviadas e recebidas, o grupo contém uma tabela – **egpNeighTable** -, em que cada linha descreve uma ligação com um sistema autónomo vizinho. A tabela é indexada pelo campos que identifica o sistema vizinho: egpNeighAddr.

4 Resumo final

FALTA FAZER !!

Nesta fase, um pouco de prática será útil para clarificar os conceitos. Uma pequena pausa antes de continuar…um pouco de café e…depois mãos à obra !!

PROBLEMAS PROPOSTOS

Problema 1:

Considere a seguinte tabela de uma MIB genérica:

a) Indique justificando, qual a chave de pesquisa na tabela.

b) Represente, a parte, da árvore de registo, respeitante a este extracto de MIB.

```
ClientTable OBJECT-TYPE
     SYNTAX      SEQUENCE      OF
ClientEntry
     ACCESS not-accessible
     STATUS mandatory
:== {exemplo 1}
ClientEntry OBJECT-TYPE
     SYNTAX ClientEntry
     ACCESS not-accessible
     STATUS mandatory
     INDEX {ClientCode}
:== {ClientTable 1}
ClientEntry :== SEQUENCE {
     ClientCode INTEGER,
     ClientName OCTECT STRING,
     ClientIpAddress IpAddress,
     ClientInTraff INTEGER,
     ClientOutTraff INTEGER,
}
ClientCode OBJECT-TYPE
     SYNTAX INTEGER
     ACCESS READ-ONLY
     STATUS mandatory
:== {ClientEntry 1}
```

```
ClientName OBJECT-TYPE
     SYNTAX OCTET_STRING
     ACCESS READ-ONLY
     STATUS mandatory
:== {ClientEntry 2}
ClientIpAddress OBJECT-TYPE
     SYNTAX IpAddress
     ACCESS READ-ONLY
     STATUS mandatory
:== {ClientEntry 3}
ClientInTraffic OBJECT-TYPE
     SYNTAX INTEGER
     ACCESS READ-WRITE
     STATUS mandatory
:== {ClientEntry 4}
ClientOutTraffic OBJECT-TYPE
     SYNTAX INTEGER
     ACCESS READ-WRITE
     STATUS mandatory
:== {ClientEntry 5}
```

Problema 2

Suponha que pretende desenvolver um serviço, muito simples, de atendimento automático de chamadas telefónicas. Este serviço deve ser activado sempre que:

- O telefone do cliente se encontre ocupado.
- O cliente esteja ausente, o que significa que o telefone toca mais do que um dado número de vezes pré-estabelecido.
- A chamada for bloqueada, por problemas internos da rede e não atingir o telefone do cliente.

Por cada vez que acede ao serviço, o Cliente pode remover ou não as chamadas memorizadas. Porém, por questões de dimensionamento da rede, cada cliente não pode ter mais de 30 chamadas memorizadas. Assim sendo, quando uma esta situação limite se atinge, é removida a primeira chamada do Cliente.

Pretende-se que o cliente tenha acesso a estatísticas referentes ao serviço que lhe está a ser disponibilizado. Estas estatísticas devem permitir contabilizar:

- As situações que originaram o atendimento automático;
- Número de chamadas memorizado
- A duração total das chamadas memorizadas

a) Considerando que dispõem de SNMPv1, desenhe uma MIB para representar a informação de todos os Clientes do serviço. Suponha que cada Cliente é identificado por um campo denominado ClienteId.

Problema 3

Indique a sequência de mensagens SNMPv1 e o conteúdo das mensagens, associado às seguintes situações:

a) Leitura da tabela de encaminhamento, ipRouteTable, de um router. Considere que a tabela tem apenas 4 linhas.

b) Alteração do valor de NextHop referente ao endereço de destino D2.

c) Remoção da linha da tabela de encaminhamento correspondente ao endereço de destino D2.

Problema 4

Suponha a existência da seguinte informação associada à tabela tcpConnTable.

a) Indique, justificando, qual a sequência de mensagens SNMP que utilizaria para efectuar a leitura das duas primeiras linhas da tabela.

TcpConnState	tcpConnLocalAddress	tcpConnLocalPort	tcpConnRemAddress	tcpConnRemPort
1	193.164.2.16	15	193.120.2.1	11
2	193.163.5.16	12	193.164.2.1	12
12	193.164.2.16	13	193.120.7.1	17

Problema 5

Suponha que possui um router com Agente SNMPv1 e MIB-II e que pretende obter os seguintes indicadores de desempenho:

- Nº médio de pacotes IP perdidos no router
- % de pacotes recebidos por protocolo de nível de rede

Indique os objectos da MIB que escolhia para os obter e como efectuava os seus cálculos.

Problema 6

Imagine um SLA em que se específica:

- Ritmo médio de transferência
- Percentagem máxima de pacotes perdidos

Qual a informação da MIB-II que se pode utilizar para avaliar se essas garantias estão a ser cumpridas ?

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[1]: Capítulo 6

[2]: Capítulo 7

[3]: Capítulo 4, 5 e 7.

Capítulo II - Arquitectura de Gestão Internet: SNMPv2

Palavras chave:

1 Introdução

Com a apresentação da MIB-II, terminou o estudo da Arquitectura de Gestão Internet, correspondente à 1ª versão do protocolo SNMP.

Como se pode caracterizar, em traços gerais, esta arquitectura ? Em primeiro lugar, trata-se de uma Arquitectura de Gestão simples, quer na perspectiva do Modelo de Informação, isto é, da estrutura de informação de gestão, quer na perspectiva do Modelo de Comunicação, ou seja, mensagens de gestão e funcionalidades associadas, quer ainda na perspectiva de segurança. Esta simplicidade tem duas consequências fundamentais: primeiro, possibilitou a sua rápida divulgação em termos comerciais, o que fez com se tornasse uma solução de gestão generalizada; segundo, limitou a sua aplicabilidade a cenários demasiado complexos, ou com requisitos de segurança.

Conforme foi abordado quando se efectuou um breve historial da Gestão Internet, a 2ª versão do SNMP surgiu exactamente para resolver as falhas da 1ª versão, nomeadamente, no que se refere à estruturação, transferência e segurança da informação de gestão. Comecemos então por identificar em que áreas é que se registaram as modificações à Arquitectura de Gestão original:

- Os aspectos de estruturação foram contemplados através de alterações ao Modelo de Informação.

- Os aspectos relativos à transferência de informação foram abordados através de alterações ao Modelo de Comunicação, que permitiram executar novas funcionalidades.

- Os aspectos de segurança também foram inicialmente considerados, muito embora não tenham sido totalmente normalizados, por falta de consenso.

Vamos efectuar hoje o estudo do SNMPv2 e do SNMPv3 para posteriormente avaliar em que medida os objectivos que estiveram nas suas géneses foram alcançados.

2 Arquitectura de Gestão SNMPv2

2.1 Modelo de Informação

O Modelo de Informação proposto na nova versão do protocolo SNMP, SMIv2, difere, significativamente, do modelo de SMI da versão 1. Para além de serem introduzidos **novos grupos na ARI**, a própria estruturação da informação de gestão é diferente. Desta forma, para garantir a operação entre um equipamento com uma MIB desenvolvida segundo SMIv2 e um Agente ou um Gestor que suportem SNMPv1 é necessário efectuar conversão de informação, o que acarreta perda de funcionalidade.

O novo modelo de informação, baseia-se assim em quatro conceitos fundamentais, que se traduzem em avanços significativos em termos de modelo objectos. Estes conceitos são:

- **Definição de objectos**
- **Tabelas conceptuais**
- **Definição de Notificações**
- **Módulos de Informação**

2.1.1 Alterações à ARI

Com o aparecimento do SNMPv2 surgiram alterações na ARI, que se traduziram, essencialmente, na adição de novos grupos. Neste contexto, por baixo do nó estrutural internet surge a sub-árvore, que aglutina as características de segurança, iniciada no nó estrutural **security** (5 na ARI) e a sub-árvore, que contém as características da versão 2 do SNMP, e que se inicia no nó **SNMPv2** (6 na ARI).

O nó SNMPv2 origina, por sua vez, três sub-árvores: **domain**, **proxys** e **snmpModules**. O nó domain contém informação referente às diferentes versões do protocolo SNMPv2; o nó proxys contém---- e nó snmpModule contém----.

SnmpModules também dá origem a três novas sub-árvores: **snmpMIB**, **snmpM2M**, **PartyMib**. snmpMIB, no seu essencial, é uma substituição do grupo snmp da MIB-II. Esta substituição deriva do facto do SNMPv1 e do SNMPv2 serem incompatíveis. snmpM2M ---- e PartyMIB---

Paralelamente, foram inseridos novos valores em alguns grupos da MIB-II, nomeadamente no grupo system.

2.1.2 Estruturação da Informação de Gestão

2.1.2.1 Definição de objectos

Em SMIv2, na definição de objectos existem alterações, que se traduzem pela existência de novos tipos ASN.1 e de uma estrutura mais completa para a macro que especifica os objectos.

Definiu-se a possibilidade de utilização de novos tipos de objectos, através da possibilidade de especificação de novos tipos ASN.1 no campo SINTAX. Estes novos tipos podem ser simples, como por exemplo, **Unsigned32** e **Counter64**, ou podem depender da aplicação, como é o caso dos **endereços OSI** (NSAP Address).

A própria Macro que define um objecto foi alterada, contendo nesta nova versão a seguinte informação: **SYNTAX**, **UnitsParts**, **MAX-ACCESS**, **STATUS**, **Description**, **ReferPart**, **IndexPart** e **DefValPart**. Saliente-se que em SMIv1, um objecto era caracterizado através do campo SYNTAX, Descrição e, no caso das tabelas, INDEX.

A Figura 13 representa uma macro de definição de um objecto em SM1v2.

Figura 13: Macro de definição de objecto em SMIv2

```
snmpInPkts OBJECT-TYPE
   SYNTAX Counter32
   MAX-ACCESS read-only
   STATUS current
   DESCRITION
       ´The total number of messages delivered to the SNMP
       entity from the transport service´
:== {snmp 1}
```

2.1.3 Comparação da definição de objectos em SMIv1 e SMIv2

As diferenças existentes na definição de um objecto em SMIv1 e SMIv2 encontram-se sumarizadas na Tabela 8. Por análise desta tabela é possível concluir existem duas versões significativamente diferentes do modelo de informação SNMP, o que vai acarretar graves problemas de interfuncionamento.

Campo	SMIv1	SMIv2
UNITS		Descrição textual que identifica as unidades associadas ao objecto. É útil para objectos que representam medidas
MAX_ACCESS (1)	Define o tipo de acesso que o objecto suporta. Pode assumir os seguintes valores: not-accessible, read-only, read-write e write-only	Define o nível máximo de acesso que o objecto suporta, independentemente de critérios de policiamento administrativos. Pode assumir os seguintes valores: not-accessible, accessible-for-notify, read-only, read-write e read-create
STATUS	Define se o objecto é de uso obrigatório ou opcional. Pode assumir os seguintes valores: mandatory e optional.	Define se o objecto existe ou se é um objecto histórico, que já não se utilizas, ou está em vias disso. Pode assumir os seguintes valores: current, obsolete e deprecated.
ReferPart	Descrição textual que se utiliza para referir um outro objecto, de outra MIB (?)	Descrição textual que se utiliza para referir um outro objecto, de outra MIB.
IndexPart	Campos utilizados para a indexação de tabelas (INDEX).	Campos utilizados para a indexação de tabelas, que podem ser de dois tipos: - INDEX: para a definição das chaves de pesquisa de uma tabela. - AUGMENTED: para aumentar o número de colunas duma tabela.
DefValPart	Valor por omissão do objecto	Valor por omissão do objecto.

Notas: (1) – em SMIv1 este campo chamava-se ACCESS:

Tabela 8 : Campos ASN.1 da macro de definição de objectos em SMIv1 e SMIv2

2.1.4 Tabelas conceptuais

Em qualquer das versões do Modelo de Informação SNMP uma tabela é definida como uma SEQUENCE OF entry, isto é uma sequência de linhas. Cada linha

conceptual é uma SEQUENCE de objectos escalares. Todos os objectos escalares têm de estar presentes, o que significa que não é possível atribuir-lhe os valores default e optional, e são definidos utilizando a MACRO de definição de objectos.

Em SNMPv2 é possível utilizar dois tipos de tabelas diferentes: tabelas em que não é permitido criar nem remover linhas e tabelas em que estas acções se podem realizar. O primeiro tipo de tabelas é útil quando se pretende representar informação estática, por exemplo as interfaces de um Router; o segundo tipo serve para representar informação dinâmica, como seja o número de ligações TCP activas.

Também é possível adicionar, dinamicamente, colunas a tabelas de qualquer um dos tipos anteriormente definidos. Esta opção é útil quando se pretendem incorporar extensões a uma tabela de base, que dependem da configuração do equipamento.

2.1.4.1 Adição de colunas

Cada linha conceptual é indexada através de um conjunto de chaves, que são identificadas através da clausula INDEX, que serve assim para definir a linha conceptual de base.

Se se substituir a clausula **INDEX** por **AUGMENT** não se está a definir uma nova tabela, mas sim a possibilitar a extensão de uma tabela pré-existente, por adição de novas colunas. A grande vantagem desta facilidade reside na possibilidade de aumentar a tabela, sem ser necessário escrever novamente a sua definição.

Os novos objectos são denominados de extensão à linha conceptual. É usual que eles façam parte de uma MIB diferente, por exemplo de uma empresa. Porém, passam a fazer parte da referida tabela, sendo processados exactamente da mesma forma que o objectos que constituem a linha conceptual de base. O objecto referido por AUGMENT tem de ser a linha da tabela conceptual de base da tabela.

EXEMPLO

Consideremos então a tabela de encaminhamento, definida em RFC 1354, ipForwardTable. Esta tabela foi concebida para substituir ipRouteTable, que apresentava como inconveniente o facto de só possibilitar a existência de uma rota por cada destino, uma vez que só possui um campo de INDEX. IpForwardTable descreve-se então da seguinte forma:

```
IpForwardTable OBJECT-TYPE
       SYNTAX SEQUENCE OF IpForwardEntry
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Tabela de encaminhamento IP >>
::= { ip 21}
IpForwardEntry OBJECT-TYPE
       SYNTAX IpForwardEntry
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Rota para um dado destino >>
       INDEX    {ipForwardDest, ipForwardNextHop }
::= { ipForwardTable 1}
IpForwardEntry ::=
       SEQUENCE {
```

```
       ipForwardDest IpAddress, ...
       ipForwardNextHop IpAddres, ..
}
ipForwardDest OBJECT-TYPE
       SYNTAX IpAddress
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION << Endereço de destino da rota >>
::= {ipForwardEntry 1}
…
ipForwardNextHop OBJECT-TYPE
       SYNTAX IpAddress
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION << Endereço seguinte da rota >>
::={ipForwardEntry7}
```

Supondo que se utiliza uma nova versão do protocolo de encaminhamento, que inlclui duas métricas dinâmicas, que conduzem à adição de duas colunas à tabela: uma referente à métrica de atraso e outra relacionada com a ocupação das filas de espera à saída da interface. Como se define a extensão à tabela ? Façam a habitual pausa para pensar, construam a vossa solução antes de prosseguirem.

Eis então a solução do problema

```
moreIpForwardTable OBJECT-TYPE
       SYNTAX SEQUENCE OF moreEntry
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Extensão a ipForwardTable >>
::= {B}
moreEntry OBJECT-TYPE
       SYNTAX moreEntry
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Adição de novas colunas a ipForwardTable >>
       AUGMENTS {ipForwardEntry}
::= {moreIpForwardTable 1}
moreEntry :==  SEQUENCE {ipDelayMetric, ipQueueLenMetric}
ipDelayMetric OBJECT-TYPE
       SYNTAX INTEGER
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION << Metrica de atraso >>
::= {moreEntry 1}
ipQueueLenMetric OBJECT-TYPE
       SYNTAX INTEGER
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION << Metrica de tamanho da fila de saída >>
::= {moreEntry 2}
```

2.1.4.2 Criação e remoção de linhas

O método escolhido pelo SNMPv2 para a criação/remoção de linhas baseia-se na na introdução desta capacidade na própria descrição da tabela e na utilização das mensagens tradicionais de Set e Get.

Estrutura da tabela

O método baseia-se na existência de um objecto, com a cláusula de SYNTAX do tipo **RowStatus**, cujo valor de **MAX-ACCESS** seja **read-create**. Por convenção, este objecto chama-se coluna de estado.

EXEMPLO

Como exemplo, podemos analisar a MIB da aplicação Ping, definida no RFC --- Esta MIB contém uma tabela com a descrição dos Pings em curso. Para cada endereço IP que está a ser monitorizado existe uma linha da tabela com a informação referente ao tráfego de pacotes ICMP. Para além destes campos, de natureza estatística, a tabela possui um campo de estado, pingStatus, que por ser do tipo RowStatus, indica que é possível inserir ou remover linhas dinamicamente desta tabela.

Os campos que se podem criar dinamicamente caracterizam-se por ter uma valor de read-create no campo MAX-ACCESS. Os restantes campos não são significativos na perspectiva de criação/remoção de linhas da tabela.

```
pingTable OBJECT-TYPE
       SYNTAX SEQUENCE OF pingEntry
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Tabela com os ping requests >>
::= {ping 1}

pingEntry OBJECT-TYPE
       SYNTAX pingEntry
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Linha da tabela ping request >>
       INDEX {pingIndex}
::= {pingTable 1}

pingEntry :== SEQUENCE {
               pingIndex         Integer32,
               pingIPAddress     IpAddress,
               pingDelay         Integer32,
               pingsRemaining    Integer32,
               pingsTotal        Integer32,
               pingsReceived     Integer32,
               pingsRtt          Integer32,
               pingStatus        RowStatus,
               pingSize          Integer32
     }

pingIndex OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS not-accessible
       STATUS current
       DESCRIPTION << Chave da tabela de pings >>
::= {pingEntry 1}
```

```
pingIPAddress OBJECT-TYPE
       SYNTAX IpAddress
       MAX-ACCESS read-create
       STATUS current
       DESCRIPTION << Endereço IP para o qual os pacotes de eco ICMP são
           enviados>>
::= {pingEntry 2}

pingDelay OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-create
       STATUS current
       DESCRIPTION <<Tempo entre envio de pacotes de eco ICMP(mseg)>>
       DEFVAL {1000}
::= {pingEntry 3}
pingRemaining OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-create
       STATUS current
       DESCRIPTION <<Nº pacotes ICMP que falta enviar na sequência>>
       DEFVAL {5}
::= {pingEntry 4}

pingTotal OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-create
       STATUS current
       DESCRIPTION <<Nº total de pacotes ICMP a enviar na sequência>>
      DEFVAL {5}
::= {pingEntry 5}

pingReceived OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION <<Nº total pacotes ICMP recebidos na sequência >>
       DEFVAL {0}
::= {pingEntry 6}

pingRtt OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION  <<Round-Trip-Delay do ultimo eco ICMP. Este objecto é
           criado quando se recebe a 1ª resposta ao eco ICMP>>
::= {pingEntry 7}

pingStatus OBJECT-TYPE
       SYNTAX RowStatus
       MAX-ACCESS read-create
       STATUS current
       DESCRIPTION <<Estado da linha da tabela>>
::= {pingEntry 8}

pingSize OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-create
       STATUS current
       DESCRIPTION <<Dimensao do eco ICMP>>
       DEFVAL {64}
::= {pingEntry 9}
```

Figura 14: MIB da aplicação PING que permite a criação/remoção de linhas

Mecanismos de criação de linhas

Existem dois mecanismos possíveis para a criação de linhas: **createAndWait** e **createAndGo**. Basicamente, no primeiro mecanismo o Gestor inicia a criação da linha, o Agente atribui valores aos objectos que tiverem definidos valores por omissão, mas aguarda que o Gestor preencha os valores dos campos, para os quais não possua nenhum valor por omissão, ou que actualize os que ele preencheu, antes de considerar que a coluna está realmente "activa". No segundo método, o Gestor inicia a criação da linha e pede informação sobre os objectos que não têm valores por omissão, atribui-lhes valores e, se a operação tiver sucesso, a coluna é criada e fica automaticamente no estado "activo".

Analisemos, em seguida, cada um destes mecanismos de forma mais detalhada.

Mecanismo createAndWait

A criação de linhas em tabelas através do método createAndWait realiza-se em 6 etapas:

1. Gestor indica ao Agente que pretende criar uma nova linha da tabela, com um dado valor para o(s) índice(s).

2. Se for possível, o Agente cria a linha e atribui a cada um dos seus objectos o respectivo valor por omissão. Se todos os objectos read-create possuirem valores por omissão, a linha é colocada no estado **notInService**; caso contrário, fica no estado **notReady**.

3. O Gestor envia um GetRequest para determinar o estado dos objectos do tipo read-create.

4. O Agente responde: (i) para cada objecto que possua um valor por omissão, com este valor; (ii) noSuchInstance para cada objecto que não tenha valor por omissão e (iii) noSuchObject para cada objecto que está definido na MIB, mas não é suportado pelo Agente.

5. O Gestor devolve um SetRequest, em que preenche o valor dos objectos que foram referenciados com noSuchInstance e, adicionalmente, pode alterar o valor por omissão dos objectos, para os quais este valor tenha sido referenciado.

6. Assim que todos os objectos read-create tenham sido criados, o Gestor envia um SetRequest para colocar o estado da coluna **Active**.

EXEMPLO:

Consideremos novamente a MIB da aplicação Ping, que num dado instante contém apenas a informação representada na Tabela 9.

Exemplifiquemos então, quais os procedimentos a realizar quando se pretende adicionar a linha representada na Tabela 10, utilizando o mecanismos anteriormente descrito.

Index	IpAddress	Delay	Remaining	Total	Received	Rtt	Status
1	128.2.13.21	1000	0	10	9	3	Active

Tabela 9: Conteúdo inicial da tabela pingTable

Index	IpAddress	Delay	Remaining	Total	Received	Status
2	128.2.13.99	1000	20	20	0	Active

Tabela 10: Linha que se pretende adicionar à tabela pingTable

De acordo com as etapas anteriormente descritas, o procedimento de adição de linhas executa-se da seguinte forma:

1. Gestor envia pedido de criação de linha com indíce 2

   ```
   SetRequest (pingStatus.2 = createAndWait)
   ```

2. Agente responde com indicação de que não possui informação suficiente para criar a linha pedida

   ```
   SetResponse (pingStatus.2 = notReady)
   ```

3. Gestor envia informação em falta

   ```
   GetRequest (pingIpAddress.2 , pingDelay.2, pingRemaining.2,
               pingTotal.2, pingStatus.2, pingSize.2)
   ```

Os campos pingReceived e pingRtt não são enviados porque são do tipo read-only e portanto não são significativos para efeitos de criação de linhas. O campo pingIndex não é enviado porque é do tipo not-accessible.

4. O Agente responde com os valores por omissão dos campos que conhece, com a indicação de noSuchInstance para os objectos que não tenham valor por omissão e com noSuchObject para os objectos que não estão suportados na sua MIB.

   ```
   GetResponse ((pingIpAddress.2 = noSuchInstance), (pingDelay.2 = 1000),
                (pingRemaining.2 = 5), (pingTotal.2 = 5),
                (pingReceived.2 = 0), (pingStatus.2 = notReady)
                (pingSize.2 = = noSuchObject))
   ```

5. Gestor envia os valores dos campos que não tinham valores por omissão ou dos campos que devam ser criados com um valor diferente do valor por omissão e coloca a linha no estado activo.

   SetRequest ((pingIpAddress.2 = 128.2.13.99), (pingRemaining.2 = 20),
   (pingTotal.2 = 20), (pingStatus.2 = active))

Mecanismo createAndGo

A criação de linhas em tabelas através do método createAndGo realiza-se em 3 etapas:

1. Gestor indica ao Agente que pretende criar uma nova linha da tabela, com um dado valor para o(s) índice(s) e envia um GetRequest, para identificar quais dos objectos read-create são noSuchInstance.

2. O Gestor envia um SetRequest e atribui valores aos objectos read-create que não tenham valores por omissão, podendo alterar os valores dos outros objectos read-create.

3. Se a operação Set tiver sucesso, a coluna é criada e colocada no estado Active.

Fica à vossa responsabilidade a realização de um exemplo ilustrativo da aplicação deste segundo mecanismo.

Mecanismos de remoção de linhas

A remoção de uma coluna é iniciada, pelo Gestor, através do envio de um SetRequest com o valor da coluna de estado igual a **destroy.** Se a operação tiver sucesso, o Agente remove a linha da tabela.

2.1.5 Descrição de Notificações

Em SNMPv2 passou a existir uma **MACRO** para definir a informação enviada por uma entidade quando uma situação de excepção ocorre, isto é, para definir **notificações**. Um exemplo desta MACRO está representado na Figura 15.

```
linkUp NOTIFICATION-TYPE
   OBJECTS (ifIndex,ifAdminStatus,
            ifOperStatus)
   STATUS current
   DESCRIPTION
      << A linkUp trap signifies that the SNMPv2 entity, acting  in
      the agent role, has detected that the ifOperStatus object for
      one of its communication links has transitioned out of the down
      state >>
:: = {snmpTraps 4}
```

Figura 15: Macro de definição de Notificações

2.1.6 Módulo de Informação

O SNMPv2 introduz o conceito de módulo de informação, que serve para especificar um grupo de definições relacionadas. Existem três tipos de módulos:

- **Módulos das MIBs**, que contém MACROS de definição de objectos e de notificações
- **Especificações de compatibilidade** para os módulos da MIB, que utilizam as MACROS MODULE-COMPLIANCE, OBJECT-GROUP e NOTIFICATION-GROUP.
- Especificações de **capacidades de implementação** do Agente, que utilizam a AGENT-CAPABILITIES MACRO.

1

2.1.6.2 Módulo das MIBS

O Módulo das MIBs contém as MACROS que especificam os objectos e as notificações e é caracterizado por um cabeçalho de módulo, com a estrutura que a seguir se indica na Figura 16.

```
snmpMIB MODULE-IDENTITY
   LAST-UPDATED ´9511090000Z´
   ORGANIZATION ´IETF SNMPv2 Working Group'
   CONTACT INFO
       ´Marshall T. Rose
       Postal: …
       Tel: …
       Email: …
   DESCRIPTION
       ´The MIB module for SNMPv2 entities´
   REVISION   ´9304010000Z´
   DESCRIPTION
       ´The initial version of this MIB a was published
       in RFC 1450´
:: = {snmpModules 1}
```

Figura 16: MACRO de definição de Módulo

2.1.6.3 Especificações de compatibilidade

As especificações de compatibilidade servem para definir os limites mínimos que as implementações têm de respeitar para se poderem considerar SNMPv2 compatíveis. Os OBJECT-GROUP e NOTIFICATION-GROUP servem para definir conjuntos de objectos e notificações, respectivamente. Cada uma destas MACROS encontra-se exemplificada na Figura 17, na Figura 18 e na Figura 19.

```
snmpBasicCompliance MODULE-COMPLIANCE
   STATUS current
   DESCRIPITON
      ´The compliance statements for SNMPv2 entities which implements
     SNMPv2  MIBs´
   MODULE
     MANDATORY-GROUPS {snmpGroup,…snmpBasicNotificationsGroup}
     GROUP snmpComunityGroup
     DESCRIPTION
        ´This group is mandatory for SNMPv2 entities which support
         community-based authentication´
::= {snmpMIBCompliance 2}  ….
```

Figura 17: MACRO de definição de Módulo de Compatibilidades SNMP

```
snmpGroup OBJECT-GROUP
   OBJECT {snmpInPkts, snmpInBadVersion, snmpInASNParseErrs,
           SnmpSilentDrops, snmpProxyDrops, snmpEnableAuthenTraps}
   STATUS current
   DESCRIPTION
     << Conjunto de objectos que fornece a instrumentação e controlo básicos
de uma entidade SNMPv2>>
:== {snmpMIBGroup 8}
```

Figura 18: MACRO de definição de Grupo de Objectos

```
snmpBasicNotificationGroup NOTIFICATION-GROUP
   OBJECT {coldStart, authenticationFailure}
   STATUS current
   DESCRIPTION
     << As duas notificações que uma entidade SNMPv2 é obrigada a
implementar >>
:== {snmpMIBGroup 8}
```

Figura 19: MACRO de definição de Grupo de Notificações

2.2 Modelo de Comunicação

A Arquitectura de Gestão SNMPv2 fornece três tipos diferentes de acesso à informação de gestão:

- Comunicação entre um Gestor e um Agente do tipo pedido resposta
- Comunicação entre um Agente e um Gestor não confirmada.
- Comunicação entre dois Gestores do tipo pedido resposta

Os dois primeiros tipos de comunicação já eram possíveis de realizar em SNMPv1, através das mensagens existentes. Porém, a nível de eficiência foram introduzidas alterações significativas, quer através da utilização de novas mensagens SNMP, quer através do aumento da eficiência da operação de cada uma das mensagens já existentes.

O último tipo é uma característica desta nova versão do protocolo. Ao permitir-se a comunicação entre Gestores está-se a aumentar o âmbito de aplicação desta

Arquitectura de Gestão, uma vez que se possibilita a sua aplicação numa rede de estrutura hierárquica, possivelmente com dimensões superiores.

Detalhemos então um pouco mais estes conceitos.

2.2.1 Tipos e formatos das mensagens SNMP

Em SNMPv2 existem 7 tipos de mensagens, representadas na Figura 20, que partilham, basicamente o mesmo formato. Destas 7 mensagens, 6 são dedicadas à comunicação entre o Gestor e os Agentes e uma à comunicação entre Gestores.

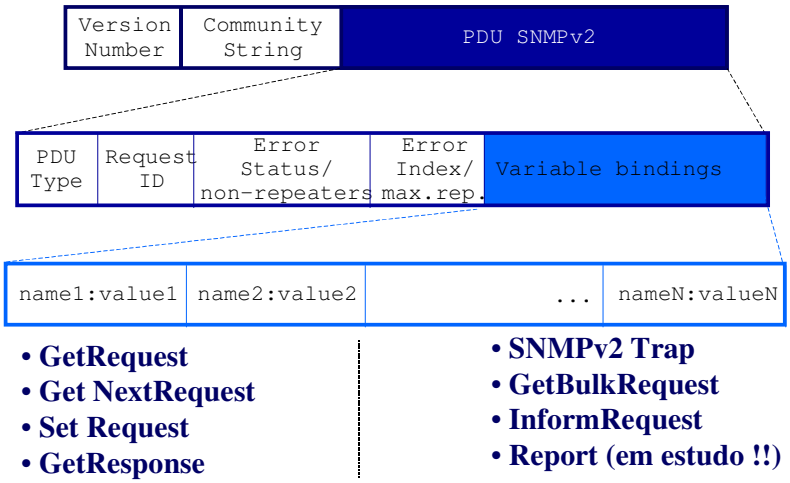


Figura 20: MACRO de definição de Grupo de Objectos

Relativamente à comunicação entre um Gestor e um Agente as mensagens que existem são:

- GetRequest, GetNextRequest e GetResponse são PDUs que já existiam com o mesmo formato e semântica em SNMPv1. A diferença fundamental reside no facto das **operações** que lhe estão associadas já **não** serem **atómicas**. Assim sendo, as acções de monitorização realizam-se independentemente sobre cada objecto definido na mensagem SNMP enviada pelo Gestor, pelo que o facto da acção não se poder realizar sobre um dado objecto não afecta o procedimento global, sendo o erro reportado no campo value associado ao objecto em causa.

- SetRequest também já existia com o mesmo formato e semântica em SNMPv1. Tal como nesta versão do protocolo, a acção de controlo é uma operação atómica. Caso contrário, quando não se pudesse realizar em todos os objectos requisitados, poder-se-ia obter uma configuração incoerente no sistema.

- GetBulkRequest é uma PDU específica de SNMPv2 , gerado pelo Gestor, com o objecto de receber do Agente um grande volume de informação, sem ter de enviar mais mensagens SNMP. Com esta PDU o Gestor pode indicar quantos objectos do mesmo tipo é que pretende receber.

- **Trap PDU** é uma PDU que já existia em SNMPv1, mas que possui nesta nova versão do protocolo um formato diferente. Este novo formato, compatível com o das restante PDUs SNMP torna o processamento mais fácil.

A transferência de informação entre Gestores realiza-se através dum novo tipo de PDU – **InformRequest** – cujo formato é idêntico aos das restantes PDUs SNMP.

2.2.1.1 Utilização de GetBulkRequest

O GetBulkRequest permite que um Gestor obtenha grandes volumes de informação, enviando apenas uma mensagem para o Agente. Para que tal seja possível, a lista de objectos definida em GetBulkRequest (campo variable-binding) inclui dois conjuntos distintos de objectos:

- **N** objectos que vão dar origem à selecção do próximo objecto em termos léxicos e cujo valor se especifica no campo **non-repeaters**, da mensagem SNMP
- **R** objectos que vão dar origem à selecção de múltiplos objectos sucessores, em que o número de sucessores pretendidos é dado através do campo **max-repetitions** (M).

Em resposta a uma mensagem GetBulkRequest, contendo (L+R) objectos, o Agente deverá responder com uma mensagem GetResponse, contendo (L+R*M) objectos. Pode acontecer no entanto que a quantidade de informação a incluir na resposta a devolver seja demasiadamente elevada, ou demore demasiado tempo a ser coleccionado pelo Agente. Em qualquer uma destas circunstâncias, o Agente deverá repartir a informação da resposta por várias mensagens.

O processamento de situações de excepção é efectuado de forma semelhante ao processamento que é efectuado em SNMPv1. Assim sendo, quando não existe mais nenhum objecto da Mib view é retornado o valor **EndOfMibView** na posição correpondente ao objecto que se segue ao último existente; quando a leitura duma tabela termina são devolvidos os sucessores léxicos da tabela. Esta mensagem é particularmente útil em dois contextos diferentes: browsing duma MIB ou leitura de uma tabela.

Consideremos então o seguinte exemplo:

EXEMPLO

Para clarificar os conceitos, consideremos novamente o extracto do grupo ip da Mib-II, que tem sido objecto de estudo ao longo destas aulas, e a leitura do objecto ipInReceives e de três linhas da tabela ipRouteTable, que estão representadas na figura 21.

1. Gestor envia uma mensagem GetBulkRequest com a indicação de que pretende ler o campo ipInReceives e os campos ipRouteDest e ipRouteNextHop correspondente a três linhas da tabela ipRouteTable:

   GetBulkRequest ((non-repeaters=2), (max-repeaters = 2),
   x0.ipInReceives, x.ipRouteDest, x.ipRouteNextHop)

   Em que: x0 =1.3.6.1.2.1.4 e x = x0.ipRouteTable.ipRouteEntry

2. O Agente responde com o valor de ipInReceives e com três linhas da tabelas, devolvendo apenas uma única mensagem GetResponse.

GetResponse ((x0.ipInReceives = 25),
(x.ipRouteDest.9.1.2.3=9.1.2.3, x.ipRouteNextHop.9.1.2.3=99.0.0.3),
(x.ipRouteDest.10.0.0.51=10.0.0.51, x.ipRouteNextHop.10.0.0.51=89.1.1.42)
(x.ipRouteDest.10.0.0.77=10.0.0.77, x.ipRouteNextHop.10.0.0.77=89.1.1.42)

3 Resumo final

PROBLEMAS PROPOSTOS

Problema 7

1- [2 val] Uma empresa de transportes públicos (combóios) tem um conjunto de equipamentos ligados em rede, com capacidades de gestão SNMPv2. Cada equipamento está localizado numa estação e por cada combóio que chega regista:

- A identificação do combóio
- O número de passageiros que entraram
- O número de passageiros que sairam.

Esta informação está representada em cada equipamento na MIB seguinte:

```
TrainTable OBJECT-TYPE
      SYNTAX SEQUENCE OF TrainEntry
      ACCESS not-accessible
      STATUS current
:== {exemplo 1}
TrainEntry OBJECT-TYPE
      SYNTAX TrainEntry
      ACCESS not-accessible
      STATUS current
      INDEX {TrainCode}
:== {TrainTable 1}
TrainEntry :== SEQUENCE {
      TrainCode INTEGER,
      InPassengers INTEGER,
      OutPassengers INTEGER
}
TrainCode OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS READ-ONLY
      STATUS current
:== {TrainEntry 1}
InPassengers OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS READ-ONLY
      STATUS current
OutPassengers OBJECT-TYPE
      SYNTAX INTEGER
      ACCESS READ-ONLY
      STATUS current
:== {TrainEntry 3}
```

Posteriormente, os responsáveis da empresa chegaram à conclusão que a informação que o equipamento fornecia era pouca para caracterizar o serviço que ofereciam aos seus clientes. Assim sendo, resolveram comprar uma nova versão do equipamento que, para além dos parâmetros anteriores, permitia registar, sob a forma de inteiros:

- A hora de chegada do combóio,
- A hora de partida do combóio.

Descreva, justificando, a extensão à MIB que deve ser desenvolvida para incluir estes novos parâmetros.

Problema 8

Indique a sequência de mensagens que deve ser transferida entre um Gestor e um Agente, para efectuar a leitura da seguinte tabela, utilizando SNMPv2, da forma mais eficiente possível.

Número	Nome	Apelido	Data Nascimento
234561	João	Silva	550901
276532	Pedro	Silva	901202
442310	Luís	Mendonça	990307

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[1]: Capítulo 6

[2]: Capítulo 7

[3]: Capítulo 4, 5 e 7.

Capítulo III - Arquitectura de Gestão Internet: SNMPv3

Palavras chave:

1 Introdução

1.1 Identificação de requisitos

A terceira versão da Arquitectura de Gestão SNMP – SNMPv3 – surgiu essencialmente pelos enormes problemas de segurança não resolvidos nas versões anteriores. A experiência obtida com a desenvolvimento da segunda versão foi fulcral para a toda a filosofia de base do SNMPv3. Neste contexto, houve algumas preocupações fundamentais que estiveram na base da sua especificação, e que por terem sido ignoradas na segunda versão, tiveram como consequência a sua ineficácia na resolução dos problemas de segurança e a sua difícil aceitação no mercado.

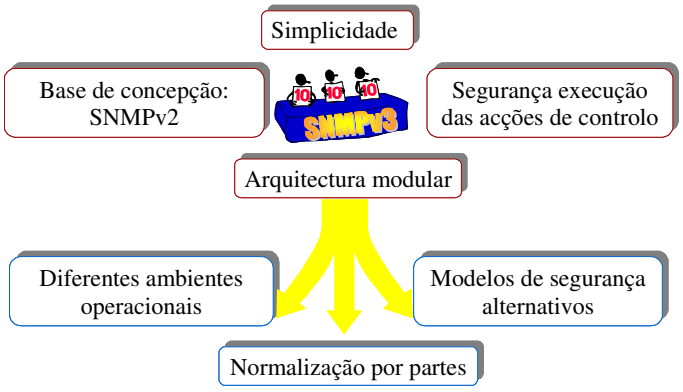


Figura 21: requisitos e características fundamentais de SNMPv3

Assim sendo, foram adoptados os seguintes princípios, que se ilustram na Figura 21, durante a especificação da arquitectura de gestão SNMPv3:

1. Utilizar como base o trabalho existente, para o qual existe experiência de implmentação e consenso sobre o seu valor acrescentado.
2. Colmatar a deficiência de segurança das versões anteriores, que impossibilita a real utilização de opções de configuração via mensagens SNMP SetRequest.
3. Conceber uma arquitectura modular, que fosse possível implementar em diversos ambientes operacionais, desde sistemas simples, com funcionalidades mínimas, até sistemas complexos, capazes de gerirem redes de grandes dimensões.
4. Permitir que esse arquitectura pudesse ser utilizada sem interferência do processo de normalização, de forma a que a sua utilização não estivesse sujeita à obtenção de consenso global em torno de toda a arquitectura.
5. Permitir a utilização de modelos alternativos de segurança.
6. Manter o SNMP o mais simples possível (chave do sucesso das versões anteriores !)

7. fosse progressivamente aceite como norma, podendo as partes normalizadas serem utilizadas

Com base nestes requisitos, foram tomadas as seguintes opções de desenvolvimento:

1. **Arquitectura**: tem de ser definida com base em limites conceptuais, que se traduzam directamente nos documentos de especificação. Neste contexto, definem-se **sub-sistemas**, que descrevem os serviços fornecidos por partes especifícas da arquitectura; as interfaces (abstractas) entre os serviços são definidas através de **primitivas de serviço** e os **serviços** que são fornecidos pelos sub-sistemas.
2. **Documentos auto-suficientes**: a especificação de uma dada parte do SNMP deve ser contida, tanto quanto possível, num documento único, que engloibará as funções, as variáveis da MIB, etc..Desta forma, assegura-se que qualquer alteração que se venha a realizar numa parte do SNMPv3, não tem qualquer impacto nos documentos que especificam outras partes da arquitectura.
3. **Configuração remota**: o sub-sistema de Segurança e Controlo de Acessos adiciona um novo conjunto de parâmetrso de configuração ao SNMP. Para alé, disso, o sistema de segurança implica a troca frequenta de chaves entre entidadase SNMP. Para que estas tarefas sejam viáveis num ambiente operacional de grande dimensão, deverá ser possível efectuar a configuração remota.
4. **Complexidade controlada**: em equipamentos simples, os recursos associafas à gestão SNMP têm de ser necessariamente reduzidos. Simultâneamente, pode haver necessidade de configurações mais complexas, que necessitem de mais funcionalidades e que possam dispender mais recursos na gestão SNMP. A especificação da arquitectura SNMPv3 deve permitir fazer face a estes dois tipos de situações antagónicas.
5. **Ataques à segurança**: o sub-sistema de Segurança deve ser capaz de proteger o sistema de gestão contra os principais tipos de ataques, nomeadamente: modificação de informação, masquerade, modificação do fluxo de mensagens, disclosure. A protecção contra a negação do serviço e a análise de tráfego não está incluída na especificação.

Ante de proceder com o estudo do SNMPv3, relembremos então alguns conceitos importantes de segurança...

1.1.1 Enquadramento da segurança no cenário de gestão

Existem vários tipos de ataques à segurança duma rede com objectivos diferenciados, que em termos de gestão podem ter consequências a diferentes níveis. Os que normalmente se destacam neste contexto são:

- **Modificação da informação**: uma entidade não autorizada pode alterar as mensagens de gestão em trânsito, modificando por exemplo os parâmetros de configuração, ou mesmo a informação de contas.
- **Mascaragem**: uma entidade pode realiza operações para as quais não tenha permissões, se assumir a identificação de uma outra entidade, autorizada para o efeito.

- **Modificação da sequência de mensagens**: as mensagens SNMP podem ser atrasadas, re-ordenadas ou duplicadas (uma vez que o protocolo é sem ligação), dando origem à realização de operações não permitidas. Por exemplo, uma mensagem de *reboot* pode ser copiada, com o objectivo de ser re-enviada noutro instante de tempo.
- **Espionagem**: uma entidade não autorizada pode observar as trocaas entre o gestor e os agentes e aprender os valores dos objectos geridos e dos eventos que dão origem a notificações. Com base neste procedimento pode, por exemplo, aprender a sequência de acções necessária à alteração de *passwords* e reconhecer as *passwords* do sistema.

Existem dois outros tipos de ataques de segurança – negação de serviço e análise de tráfego – que não são objecto de protecção em SNMPv3. A negação de serviço impede as trocas entre um gestor e um agente e, em muitas situações não se consegue distinguir duma falha na rede. Para além disso, quando ocorrem, afecta todo o tipo de operações, pelo que deve ser resolvido ao nível da segurança global do sistema e não apenas ao nível da gestão. Relativamente à análise de tráfego, muitas trocas de informação de gestão têm um padrão previsível, em que os comandos e as respostas são emitidos de forma regular, pelo que não vale a pena prevenir este tipo de ataques.

2 Arquitectura SNMPv3

2.1 Entidades SNMP

A arquitectura de gestão SNMPv3 é constituída por um conjunto de entidades SNMP distribuídas e que cooperam entre si. Cada entidade implementa um sub-conjunto das funcionalidades SNMP, podendo funcionar como Gestor, Agente ou uma combinação de ambos. Dentro de cada entidade existem módulos que interagem entre si para fornecer serviços.

Na Figura 22 representa-se um esquema genérico dum entidade SNMP constituída por uma **máquina SNMP** (*SNMP engine*) e um **conjunto de aplicações**.

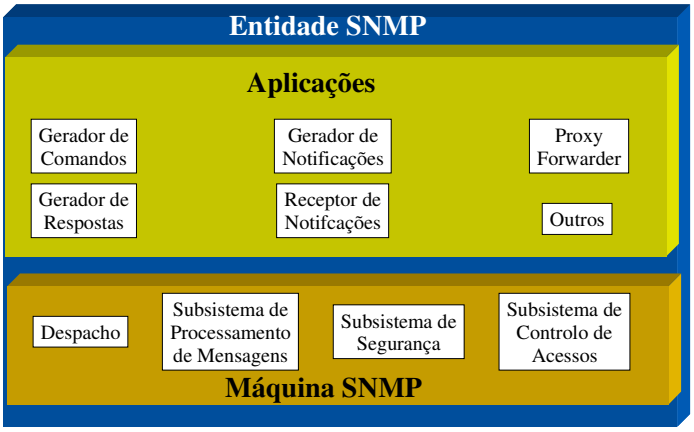


Figura 22: Entidade SNMP

A máquina SNMP é responsável por receber e enviar as mensagens SNMP, autenticar, cifrar e decifrar, e controlar o acesso aos objectos geridos. Estas funcionalidades são fornecidas como serviços a uma ou mais aplicações.

2.1.1 Aplicações SNMP

Conforme se pode ainda constatar na referida figura, ao nível das aplicações, para além de outros módulos não definidos, pode existir o seguinte conjunto:

- **Gerador de Comandos**, inicia a geração de mensagens de dados SNMPv1 ou SNMPv2 do tipo `Get`, `GetNext`, `GetBulk` e `Set` e processa as respostas a um pedido efectuado.

- **Gerador de Respostas**, recebe mensagens de dados SNMPv1 ou SNMPv2 do tipo `Get`, `GetNext`, `GetBulk` e `Set` destinadas para o sistema local e processa-as, executando a operação pedida, realizando os procedmentos de controlo de acesso necessários e gerando a mensagem de resposta a ser enviada para o emissor.

- **Gerador de Notificações**, monitoriza um sistema para determinados eventos e condições e gera mensagens assíncronas baseadas nesses eventos e condições, que no caso dum Agente se traduzem em `Traps` PDus e no caso dum Gestor em `InformRequest` PDUs. Este módulo tem de incluir um conjunto de mecanismos que permitam identificar os destinatários das mensagens assíncronas, a versão de SNMP a utilizar e os mecanismos de segurança a incluir.

- **Receptor de Notificações**, processa as mensagens assíncronas `InformRequest` e `Traps` PDUs que são recebidas, gerando uma mensagem `Response`, no caso da recepção da InformRequest PDU.

- ***Proxy Forwarder*** , recebe e envia mensagens SNMP.

2.1.2 Máquina SNMP

Ao nível da máquina SNMP, os módulos que podem existir são:

- **Despacho**, permite o suporte concorrencial a múltiplas versões de mensagens SNMP na máquina SNMP, sendo responsável por: (i) aceitar para transmissão PDUs SNMP das aplicações; (ii) receber da rede PDUs SNMP e entregá-las às aplicações; (iii) entregar as PDUs destinadas à rede ao Sub-sistema de Processamento de Mensagens para serem convertidas em mensagens; (iv) entregar as mensagens destinadas às aplicações ao Sub-sistema de Processamento de Mensagens para serem convertidas em PDUs e (v) enviar e receber mensagens SNMP através da rede.

- **Sub-sistema de Processamento de Mensagens**, prepara as mensagens para enviar e extrai os dados das mensagens recebidas.

- **Sub-sistema de Segurança,** fornece serviços de segurança, tais como autenticaçãoe garantia de privacidade, podendo suportar diferentes modelos de segurança.

- **Sub-sistema de Controlo e Acessos,** fornece serviços de autorização que uma aplicação pode utilizar para verificar os direitos de acesso, podendo ser requisiado sempre que se pretenda efectuar muma acção de monitorização ou controlo, ou sempre que se pretenda gerar uma notificação.

2.1.3 Estutura do Gestor e do Agente

Conforme se pode constatar da análise da Figura 23 e da Figura 24, a existência destes módulos depende, entre outras coisas, da entidade ser um Gestor, um Agente ou ambos.

Assim sendo, ao nível da máquina SNMP as funcionalidades do despacho, do sub-sistema de processamento de mensagens e do sub-sistema de segurança existem quer no Gestor quer no Agente. O sub-sistema de Controlo de Acessos existe apenas no Agente, uma vez que este é a entidade que possui a MIB.

Em qualquer dos casos, o despacho subdivide-se no processamento das PDUs, no processamento das mensagens e no mapeamento para o transporte que difere consoante a tecnologia de rede utilizada para o transporte da informação de gestão. Relativamente ao sub-sistema de processamento de mensagens podem existir diferentes componentes para lidar com as diferentes versões de SNMP. O Sub-sistema de Segurança especifica apenas o modelo de segurança baseado em utilizador, embora possa incluir outros.

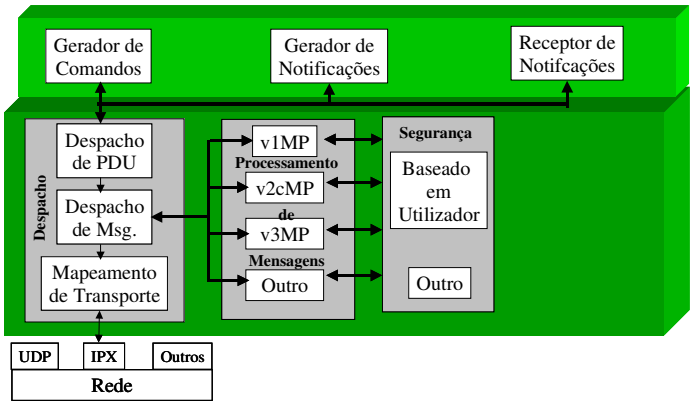


Figura 23: Estrutura dum Gestor SNMPv3

Ao nível da aplicação as diferenças são mais pronunciadas. Um Gestor pode conter um gerador de comandos e um gerador e um receptor de notificações, enquanto que um Agente contém um receptor de comandos, um gerador de notificações e um proxy, para comunicar com recursos que não possam conter um Agente. Adicionalmente, o Agente tem de possuir mecanismos que lhe permitam manipular as MIBs.

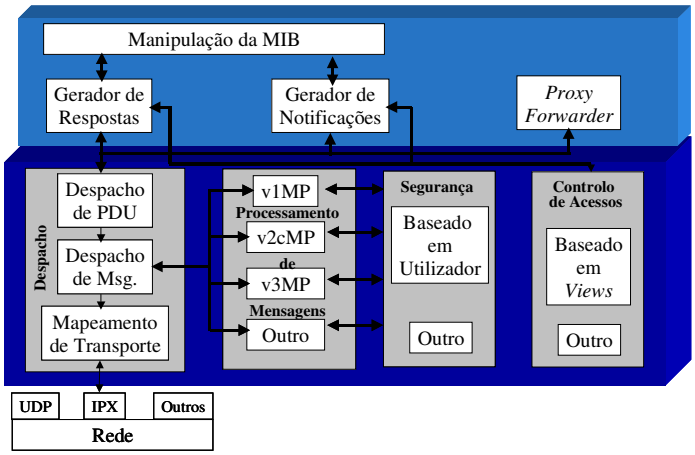


Figura 24: Estrutura dum Agente SNMPv3

2.2 Interfaces abstractas

Os serviços entre os vários módulos SNMP são definidos através de **Primitivas**, que especificam a função a ser executada e **Parâmetros**, que são utilizados para transferir dados e informação de controlo. Uma interface que se define utilizando primitivas e parâmetros chama-se de **Interface de Serviço Abstracta**.

Consideremos então o exemplo representado pela seguinte expressão que serve para ilustrar o conceito em análise:

```
codigo_funcao = pedido_servico (IN x, IN y, OUT z, OUT w)
```

A primitiva chama-se pedido_servico e possui quatro parâmetros: x e y são parâmetros de entrada, cujos valores são fornecidos pela entidade que evoca a primitiva; z e e são parâmetros de saída, que serão fornecidos após a execução da primitiva. O valor de retorno, codigo_funcao representa outro tipo de parâmetro, denominado de valor da primitiva.

Nos RFCs que definem SNMPv3 estão especificadas as interfaces de serviço que são necessárias para implementar SNMPv3. Em vez de efectuar uma abordagem exaustiva das várias interfaces, vamos encerrar este assunto com um exemplo de aplicação, onde se ilustra a sua utilização efectiva.

EXEMPLO

O exemplo consiste na construção dum diagrama temporal ilustrativo da situação em que um Gestor está a fazer polling a um Agente, enviando-lhe uma PDU SNMP e recebendo deste a resposta.

Façamos uma breve pausa (porque está na hora do café !!!) e aproveitemos para responder às seguintes questões:

- Quais são os módulos envolvidos?
- Qual a sequência de acções ?
- Quais as interfaces a definir ?

Do ponto de vista do Gestor, uma acção de polling envolve o módulo de Geração de Comandos, o Sub-sistema de Processamento de Mensagens, o Sub-sistema de Segurança e o Despacho. No Agente, a mesma acção engloba o módulo de Recepçã de Comandos, o Sub-sistema de Processamento de Mensagens, o Sub-sistema de Segurança, o Sub-Sistema de Controlo de Acessos e o Despacho.

Em termos de sequência de acções, a situação do Gestor é a seguinte:

1. Inicialmente a aplicação gera uma PDU que contem a indicação da acção de polling a realizar e do Agente a que se destina. Essa PDU é construída pelo Gerador de Comandos e enviada para o Despacho.

2. O Despacho identifica a versão do protocolo SNMP suportada pelo Agente e encaminha a PDU para a parte do Sub-sistema de Processamento de Mensagens encarrega dessa versão do protocolo.

3. Após o Sub-sistema de Processamento de Mensagens proceder à construção da mensagem, esta é enviada para o módulo de segurança, para que os mecanismos de segurança sejam incluídos antes de se efectuar a transmissão para a rede.

Conforme se representa na Figura 25, no final de cada uma destas fases define-se uma interface entre dois módulos, correspondendo à evocação de uma primitiva.

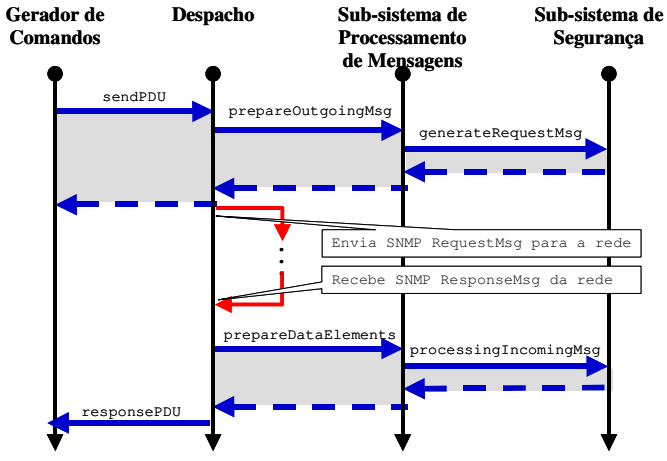


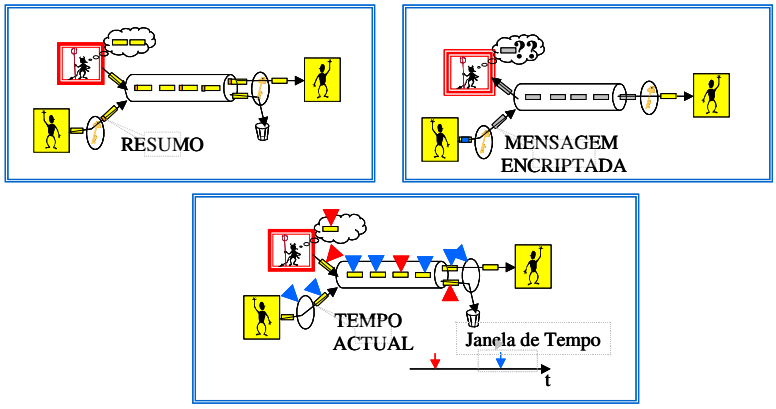
Figura 25: Diagrama temporal do polling: caso do Gestor

Neste exemplo, definem-se então as seguintes interfaces: sendMsg, processResponsePDU, (Gerador de comandos ↔ Despacho) prepareOutgoingMsg, prepareDataElements, (Despacho ↔ Sub-sistema de Processamento de Mensagens) generateRequestMsg e processIncomingMsg (Sub-sistema de Processamento de Mensagens ↔ Sub-sistema de Segurança). Enquanto a função a executar por uma dada primitiva não estiver concluída, não podem ser executadas outras tarefas. Esta situação está representada na pelas setas azuis e pelas zonas a sombreado cinzento.

2.2.1 Modelo de Segurança

O principal valor acrescentado do SNMPv3 é a existência de mecanismos de segurança e controlo de acessos, que podem ser utilizados de forma flexível, adaptando-se assim aos requisitos de cada sistema. Existe um modelo de segurança, que é o que se vai descrever em seguida, que é o modelo de segurança baseado em utilizador.

PARA COMPLETAR



Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

Capítulo IV - Monitorização remota.

Palavras chave: MIB-RMON, Alarmes, Eventos, Filtragem

1 Introdução

Terminou a apresentação das duas primeiras versões do protocolo SNMP. Conforme se pode constatar, a segunda versão traduziu-se em duas melhorias significativas: (i) eficiência de transmissão de grandes volumes de informação e (ii) estruturação da informação de gestão. Hoje, vamos regressar ao SNMPv1 para introduzir um dos conceitos mais interessantes desta arquitectura de gestão: a monitorização remota, mais vulgarmente conhecida por RMON.

Com a divulgação do SNMP e da MIB-II rapidamente se chegou à conclusão que a informação de gestão que estava disponível era claramente insuficiente para fazer face aos problemas leventados pela comunicação entre sub-redes diferentes: a MIB-II só fornecia informação de tráfego dentro de um sistema e era necessário conhecer o tráfego que circulava na rede.

Assim sendo, surgiu a ideia de desenvolver para esse efeito um monitor espião, cuja tarega era inspeccionar o tráfego qure circulava na rede local. Este monitor, ou sonda RMON, como é vulgarmente conhecido, pode operar apenas numa rede de meio partilhado o que, à data do seu aparecimento, servia inteiramente para satisfazer as necessidades de gestão.

Vamos estudar agora alguns dos conceitos RMON. O assunto é muito extenso, pelo que a abordagem será efectuada apenas ao nível introdutório. No entanto, os conceitos fundamentais serão introduzidos, de forma a que futuramente possam ser explorados com sucesso, se de tal necessitarem !!

2 Objectivos da Monitorização Remota

De acordo com RFC 1657, a Monitorização Remota possui 5 objectivos fundamentais:

- **Operação off-line**, com a finalidade de libertar o Gestor central da tarefa de efectuar o polling contínuo e periódico aos Agentes, o Monitor tem de ser capaz de coleccionar informação de faltas e de configuração continuamente, independentemente de estar ou não a ser interrogado pelo Gestor.

- **Detecção e relato de problemas**: o Monitor pode, passivamente, reconhecer certas situações de erro e de congestão, tem por base o tráfego que observa. Quando estas situações surgem o Monitor deve ser capaz de as armazenar, num log, e de tentar notificar o gestor da sua ocorrência.

- **Monitorização pro-activa**: se Monitor tiver recursos suficientes pode, continuamente, executar diagnósticos e armazenar, em logs, a informação de desempenho da rede. Em caso de falha, pode notificar o Gestor, enviando-lhe, conjuntamente com a informação da falha, a informação de diagnóstico.

- **Informação de valor-acrescentado**: o Monitor pode efectuar análise sobre os dados que colecciona, aliviando o gestor desta responsabilidade. Por exemplo, o Monitor pode identificar os hosts que geram mais tráfego.

- **Múltiplos gestores**: numa configuração inter-rede podem existir diferentes gestores. Os Monitores devem ser capazes de interagir, simultaneamente, com diversos gestores.

2.1 Controlo de um Monitor Remoto

Um Monitor remoto pode ser realizado num dispositivo dedicado, ou num dispositivo que também é capaz de implementar outras funções. Com recursos dedicados, é possível realizar funções mais complexas do que as que se esperam dum agente que suporte, unicamente, MIB-II.

Para gerir um Monitor remoto RMON de forma eficiente, a MIB-RMON contém capacidades que servem para suportar uma extensa gama de funções de controlo, a serem executadas pelo gestor. Estas capacidades incluem: **configuração** e **invocação de acção**.

2.2 Capacidade de configuração

Um Monitor remoto RMON precisa de ser configurado para coleccionar dados. Através da configuração é possível indicar o tipo e estrutura dos dados a coleccionar.

A MIB-RMON permite a realização desta tarefa de configuração da seguinte forma:

- A MIB está organizada em grupos funcionais.

- Em cada grupo pode existir uma, ou mais, **tabelas de controlo** e uma, ou mais, **tabelas de dados**.

- Uma tabela de controlo é do tipo de leitura/escrita e contém parâmetros que descrevem os dados, da tabela de dados.

- Uma tabela de dados é do tipo leitura.

No instante inicial, o gestor configurar o Monitor remoto, para receber determinados tipos de dados, por adição de novas linhas na tabela de controlo, ou modificação das linhas existentes. A informação é armazenada na tabela de dados, de na linha correspondente da tabela de dados.

2.3 Tabela de Controlo

Cada linha da Tabela de Controlo descreve uma função a ser realizada pelo Monitor remoto. As linhas da Tabela de Controlo são criadas dinâmicamente, através da utilização das mensagens usuais SNMP, envolvendo o preenchimento do conjunto de quatro campos que caracterizam uma Tabela de Controlo (ver Figura 26)

- **rmlControlIndex**, que identifica a linha da Tabela de Controlo e o conjunto de linhas da Tabela de Dados que lhe está associado.

- **rmlControlParameter**, que identifica o conjunto de parâmetros que especificam a funcionalidade a ser executada pelo Monitor Remoto, no âmbito de uma dada acção de monitorização.

- **rmlControlOwner**, que identifica o Gestor responsável pela função de monitorização, inicialmente o que criou a entrada na tabela. É possível negociar este campo, e substituir o Gestor que lá está referenciado por outro. O acesso à informação de monitorização remoto também pode ser efectuado por outros Gestores. Porém, quando o Gestor responsável decide remover a função, a informação deixa de estar automaticamente acessível para os outros.

- **rmlControlStatus**, que define o estado de criação da linha. Este variável assume o valor definido na máquina de estados representada na Figura 27.

```
rmlControlTable OBJECT-TYPE
   SYNTAX SEQUENCE OF RmlControlEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION << Uma tabela de Controlo >>
::= {ex 1}

RmlControlEntry OBJECT-TYPE
   SYNTAX SEQUENCE OF RmlControlEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION << Define um parâmetro que controla um conjunto de entradas
da tabela de dados >>
   INDEX {rmlControlIndex}
::= {rmlControlTable 1}

RmlControlEntry ::= SEQUENCE {
   rmlControlIndex INTEGER,
   rmlControlParameter Counter,
   rmlControlOwner  OwnerString,
   rmlControlStatus RowStatus
}

RmlControlIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION << Identifica a entrada na tabela de controlo >>
::= {rmlControlEntry 1}

RmlControlParameter OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-write
   STATUS mandatory

   DESCRIPTION << Caracteriza as linhas da tabelas de dados associadas a
esta linha de tabela de controlo >>
::= {rmlControlEntry 2}

RmlControlOwner OBJECT-TYPE
   SYNTAX OwnerString
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION << Identifica a entidade que configurou esta linha da tabela
de controlo >>
::= {rmlControlEntry 3}

RmlControlStatus OBJECT-TYPE
   SYNTAX EntryStatus
```

```
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION << Estado desta linha da tabela de controlo >>
::= {rmlControlEntry 4}
```

Figura 26: Formato genérico da Tabela de Controlo

Figura 27: Máquina de estados da manipulação de linhas na Tabela de Controlo

2.4 Tabela de Dados

Um conjunto de linhas da Tabela de Dados contém informação sobre o resultado duma acção de monitorização, estando por isso associado a uma linha da Tabela de Controlo e sendo identificadas, globalmente, por um índice –**rmlDataControlIndex**. Cada uma destas linhas individualmente pode ser acedida por utilização adicional do índice rmlDataIndex. A estrutura desta tabela é a que se descreve na Figura 28.

```
rmlDataTable OBJECT-TYPE
   SYNTAX SEQUENCE OF RmlDataEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION << Uma tabela de Dados >>
::= {ex 1}


RmlDataEntry OBJECT-TYPE
   SYNTAX SEQUENCE OF RmlDataEntry
   ACCESS not-accessible
   STATUS mandatory
   DESCRIPTION << Uma entrada da tabela de dados >>
   INDEX {rmlDataControlIndex, rmlDataIndex}
::= {rmlDataTable 1}


RmlDataEntry ::= SEQUENCE {
   rmlDataControlIndex INTEGER,
   rmlDataIndex INTEGER,
   rmlDataValue Counter
}

RmlDataControlIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION << Identifica a tabela de controlo correspondente a um
conjunto de entradas na tabela de dados >>
::= {rmlDataEntry 1}

RmlDataIndex OBJECT-TYPE
   SYNTAX INTEGER
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION << Identifica uma entrada na tabela de dados, de entre o
conjunto de entradas referentes á mesma entrada da tabela de controlo >>
::= {rmlDataEntry 2}

RmlDataValue OBJECT-TYPE
   SYNTAX Counter
   ACCESS read-only
   STATUS mandatory
```

```
   DESCRIPTION << O valor retornado pela entrada >>
::= {rmlDataEntry 3}
```

Figura 28: Formato genérico da Tabela de Dados RMON

2.5 Invocação de acção

O SNMP não possui nenhum mecanismo que obrigue um Agente a realizar uma acção, uma vez que só pode ler e escrever variáveis. No entanto, através do Set é possível executar um comando. A este funcionalidade chama-se invocação de Acção.

Na especificação RMON, os objectos sobre os quais se realiza o Set, representam estados. A realização da acção ocorre quando o gestor origina uma transição de estados.

3 Grupos da MIB RMON-1

A MIB RMON 1 cobre essencialmente a tecnologia Ethernet (RFC 1757). Em RFC 1513 define-se informação referente a Token-Ring.

Esta MIB é constituída pelos seguintes grupos, que se utilizam para a obtenção de estatísticas

- **Statistics**: informação referente a estatísticas de baixo nível de cada subrede monitorizada (carga na rede, erros de checksum, colisões etc..).
- **History:** amostras estatísticas periódicas, obtidas a partir de informação do grupo anterior.
- **Host**: contadores de tráfego de estações pertencentes à mesma sub-rede. O Monitor descobre as estações que existem porque espia a sub-rede.
- **HostTopN**: estatísticas ordenadas, de acordo com o máximo valor de um dado parâmetro.
- **Matrix**: informação de erros e de utilização, representada matricialmente, para qualquer par origem-destino.
- **TokenRing**: semelhante ao grupo statistics, mas para redes Token-Ring.
- Os grupos resultantes, que se indicam em seguida, servem para definir alarmes e filtros.
- **Alarm**: define o intervalo de amostragem e os limites para os alarmes, associados a qualquer contadores ou inteiro, registado por RMON.
- **Filter**: permite observar pacotes que obedeçam a um determinado filtro. O feixe de pacotes que passa no filtro chama-se canal e o número de pacotes é contado. O valor de contador pode ser utilizado para disparar eventos, ou capturar pacotes.

- **Capture**: define o modo como a informação é transmitida para o gestor. Existe um buffer que guarda a informação que passa nos filtros

- **Event**: define uma tabela com todos os eventos que o Monitor sabe gerar. Os eventos são disparados por uma condição e podem disparar um acção(ex: produzir um registo de história ou gerar um TRAP).

Façamos uma pausa em tanta teoria. Já sabem….café !!

Vamos construir uma rede, que será utilizada ao longo desta aula para exemplicar as potencialidades do RMON. Consideremos então uma rede local, de tecnologia Ethernet, onde existe um Monitor RMON e um conjunto de 25 hosts. Este monitor está ligado à sub-rede pela interface 1, que em termos de MIB-II é representada pelo índice 2. Nesta rede existe um conjunto de situações problemáticas que foram detectadas, impiricamente:

- O tráfego é muito elevado.
- Há suspeitas, embora não confirmadas, de que um número muito reduzido de hosts contribui muito significativamente para essa situação.
- Em particular, existe um host que gera sistemáticamente muito tráfego para um destino em particular, sendo responsável pela deterioração do funcionamento da rede.

Em que medida é que o RMON pode ajudar a resolver esta situação ? Nesta fase é difícil saberem. Esta resposta vai sendo dada, há medida que os diversos grupos RMON forem descritos em mais detalhe.

3.1 Grupo statistics

A informação do grupo statistics é constituída por um conjunto de contadores que se encontra representado numa única tabela, sendo criada uma linha por cada interface monitorizada. Exemplo típico de contadores deste grupo são: etherStatsDropEvents, etherStatsOctets, etherStatsPkts etc..

EXEMPLO

Comecemos então a construção do nosso exemplo. Em primeiro lugar, para efectuar qualquer tipo de análise é necessário recolher estatísticas de tráfego, referente à sub-rede.

1. Assim sendo, o Gestor vai iniciar a activação duma entrada na tabela referente ao processo de monitorização, enviando:

```
SetRequest ((x.etherStatsStatus = create-request))  // Pedido de criação de linha
```

2. Seguidamente, o Gestor vai efectuar a programação do processo de monitorização, enviando:

```
SetRequest ((x.etherStatusIndex=1),  // Linha associada à monitorização da interface
            (x.etherStatsDataSource=2),    // Índice da interface na MIB-II
            (x.etherStatsOwner=´RNL´))     // Gestor responsável
```

3. Para finalizar, o Agente inicia o processo de monitorização, colocando as estatísticas e zero e enviando o resultado da activação para o Gestor, através de:

```
SetResponse ((x.etherStatsStatus = valid))   // Confirmação de criação de linha
```

3.2 Grupo History

A informação do grupo history é constituída por um conjunto de amostras estatisticas efectuadas sobre os dados recolhidos pelo grupo anterior, sendo obtida através de uma Tabela de Controlo e uma Tabela de Dados.

Na Tabela de Controlo especifica-se informação referente à interface, ao número de amostras a recolher e ao intervalo entre amostragens. Cada amostra obtida é armazenada numa linha da Tabela de Dados, que funciona como um buffer circular.

Para uma dada sub-rede podem ser efectuados vários processos de amostragem, desde que o intervalo entre amostragens seja diferente de processo para processo. Desta forma, em amostragens com intervalos mais curtos podem ser detectadas pequenas variações no tráfego e, em amostragens com intervalor maiores, vão ser encontradas as flutuações de longo termo.

EXEMPLO

As estatísticas recolhidas anteriormente não dão ideia ao gestor da evolução do sistema em períodos de tempo bem identificados. Para que tal seja possível é necessário utilizar as potencialidades deste grupo da MIB, efectuando a amostragem do comportamento do sistema durante determinados intervalos de tempo.

Considere-se então, que o Gestor quer avaliar o comportamento do sistema durante uma hora, recolhendo um total de 10 amostras. Neste contexto, o gestor inicia a criação duma linha da tabela de controlo, e durante a fase de parametrização fornece a seguinte informação ao Agente

```
SetRequest ((x.HistoryControlIndex=1),                 // Linha da Tabela de Controlo
            (x. HistoryControl DataSource=2),          // Índice da interface na MIB-II
            (x. HistoryControlOwner=´RNL´)             // Gestor responsável
            (x. HistoryControlBucketRequested=10)      // Número de amostras pretendido
            (x. HistoryControlInterval=10*60)          // Intervalo entre amostragens [seg]
```

Com esta função de monitorização activa, o Gestor tem acesso a informação do tráfego que circula numa sub-rede, em determinados instantes de tempo, podendo assim efectuar um avaliação da sua evolução temporal.

3.3 Grupo Host

A informação do grupo host é constituída por um conjunto estatisticas referentes a hosts específicos da rede local. O Monitor remoto aprende quais são os hosts que existem na sub-rede, por inspecção dos endereços MAC de origem e de destino de cada trama que analisa. Este grupo é constituído por três tabelas:

- **hostControlTable**, a Tabela de Controlo, onde se especifica a interface que se pretende monitorizar;

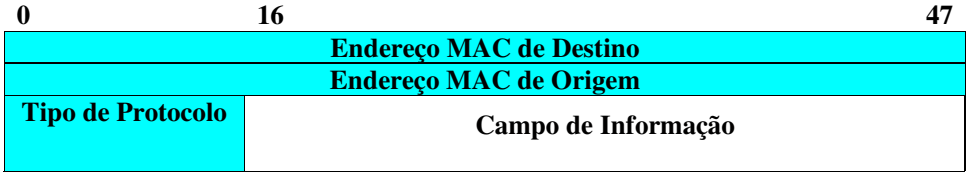
- **hostTable**, a Tabela de Dados que contém uma entrada por cada endereço MAC descoberto na interface. Por cada vez que um novo host é descoberto, é adicionada uma linha a esta tabela e o valor de hostControlTableSize é incrementado. Cada linha desta tabela é identificada pela índice da interface, que foi definido quando se criou a função de monitorização correspondente em hostControlTable, e pelo endereço MAC do host.

- **HostTimeTable**, que é outra Tabela de Dados e que contém exactamente a mesma informação da anterior, mas indexada pela interface e pelo instante de descoberta.

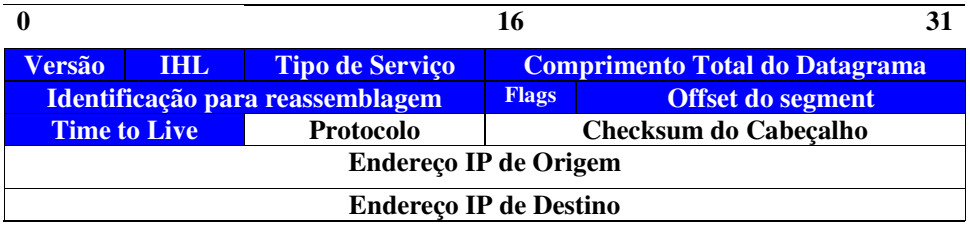
EXEMPLO

Regressemos novamente ao nosso exemplo: com as duas funções anteriores já é possível avaliar o tráfego na sub-rede e a sua evolução ao longo do tempo, em instantes de amostragem pré-definidos.

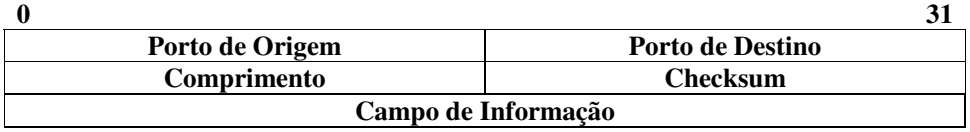
Cabeçalho das Tramas Ethernet



Cabeçalho dos Pacotes IP



Cabeçalho do Pacote UDP



Porém, nada se sabe sobre a distribuição do tráfego pelos hosts que o geraram. Esta informação é obtida exactamente a partir duma função de monitorização associada a este grupo da MIB RMON.

Assim sendo, vai-se programar a Tabela de Controlo para analisar o tráfego dos hosts na interface que o Monitor remoto possui nessa rede local, isto é, vais-e efectuar a seguinte programação:

```
SetRequest ((x.hostControlIndex=1),          // Linha da Tabela de Controlo
                    (x. hostControl DataSource=2),      // Índice da interface na MIB-II
              (x. hostControlOwner=´RNL´))      // Gestor responsável
```

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

Capítulo V - Arquitectura de Gestão OSI.

Palavras chave:

4 Introdução

Terminou o estudo da Arquitectura de Gestão SNMP, através de uma breve descrição das novas características do SNMPv3. Sumariamente, pode-se considerar que o SNMP se caracteriza pela simplicidade e por uma enorme divulgação no mercado, sendo a sua evolução pautada no sentido da eficiência da transmissão, da modularidade e da segurança. Pode-se ainda referir que a normalização das funcionalidade surgiu no âmbito do RMON, de forma não directamente relacionada com a evolução do protocolo.

Vamos iniciar agora o estudo da Arquitectura de Gestão OSI. Conforme terão oportunidade de avaliar, trata-se de uma arquitectura bastante completa e complexa, o que se traduz numa dificuldade significativa de implantação no mercado. Tem porém uma grande vantagem, que justifica o seu estudo – é uma Arquitectura de referência- que tem servido de base à evolução das restantes arquitecturas. Conforme terão oportunidade de verificar, a evolução que se efectuou na Arquitectura de Gestão SNMP e o próprio RMON são, em grande parte, inspirados na arquitectura OSI.

Antes de iniciar o estudo resta referir que a Arquitectura de Gestão OSI é a base da Arquitectura de Rede de Gestão de Telecomunicações, vulgarmente conhecida pela sua abreviatura em Inglês – TMN – Telecommunication Management Nerwork. Mas essa fica para a próxima aula.

Comecemos então o seu estudo. Conforme irão verificar um grande número de conceitos que hoje vos irão ser apresentados de uma forma abstracta, já são vossos conhecidos da Plataforma de Gestão TeMIP.

5 Arquitectura de Gestão OSI

A Arquitectura de Gestão OSI foi definida pelo ISO, como uma adenda ao Modelo de Referência de Protocolos OSI, através dum conjunto de normas, denominadas "Management Framework" (7498-4). Estas normas são idênticas as que são publicadas pelo ITU-T (antigo CCITT), na série X.700.

É uma arquitectura que engloba os quatros modelos que foram definidos:

- **Modelo de Informação**, que adopta integralmente uma aproxinação orientada-a-objectos, estabelecendo metodologias estruturadas de modelização da informação de gestão.
- **Modelo de Organização**, que se baseia numa arquitectura Gestor – Agente, em que os papeis podem ser permutáveis, em função, por exemplo, do recurso que se está a modelizar.
- **Modelo de Comunicação**, que está baseado no Modelo de Referência OSI, permitindo a transferência de informação de gestão ao nível da aplicação, entre camadas pares de sistemas diferentes ou entre camadas de um sistema e uma entidade de gestão.

- **Modelo Funcional**, divide a gestão nas cinco Áreas Funcionais que foram definidas durante as primeiras aulas, subdividindo cada área num conjunto de funções mais específicas.

A nível de Redes de Telecomunicações a Arquitectura de Gestão OSI é bastante popular. As razões técnicas que o justificam estão relacionada com a flexibilidade que o elevado nível de abstracção da especificação orientada-a-objectos possibilita e com o facto da abordagem top-down ser facilmente realizável, uma vez que as próprias funções de gestão são definidas como Classes de Objectos Geridos. Para além destas razões, existem razões de negócio, associadas ao facto desta Arquitectura ser normalizada internacionalmente, no âmbito dos organismos que regulamentam as próprias telecomunicações.

5.1 Modelo de Informação

5.1.1 Estruturação do Modelo de Informação

O Modelo de Informação, designado por Structure of Management Information encontra-se definido no seguinte conjunto de normas:

- **Management Information Model** (MIM), que define os conceitos fundamentais que devem se seguidos na definição de Objectos Geridos, constituindo assim a base de todas as outras recomendações SMI.
- **Definition of Management Information** (DMI), que contém um conjunto de definições referentes à Informação de Gestão.
- **Guideline for Definition of Management Objects** (GDMO), que define as técnicas de especificação de Objectos Geridos, fornecendo assistência prática aos projectistas que têm de definir Informação de Gestão. Todos os Objectos Geridos devem respeitar a recomendação GDMO. Esta garantia não assegura que respeitem o MIM, mas ajuda !
- **Requirements and Guidelines for Implementation Conformance Statement Proformas associated with Management Information** (MOCS), que define as regras e o formato de produção de testes de conformidade.
- **Generic Management Information** (GMI), que define os Objectos Geridos que são utilizados pelo modelo OSI, servindo para garantir compatibilidade entre as várias camadas do modelo.
- **General Relationship Model** (GRM), que define um modelo para estabelecer relações entre Objectos Geridos.

5.1.2 Características gerais

O Modelo de Informação é muito complexo, sendo realizado pela aproximação orientada-a-objectos. Os Objectos Geridos são instâncias de Classes de Objectos Geridos, que estão organizadas hierarquicamente. As características dos Objectos são visíveis descritos no **Managed Object Boundary**, de acordo com as propriedades da respectiva classe.

Conforme se representa na **Error! Reference source not found.**, de uma forma genérica, o Managed Object Boundary é constituído por:

- **Atributos**, que caracterizam as propriedades e o estado do Objecto Gerido
- **Operações,** que identificam as operações que se podem realizar sobre os Atributos ou sobre o Objecto Gerido como um todo.
- **Notificações**, que identificam os eventos assíncronos que podem ser gerados pelo Objecto Gerido.
- **Outras características**, que descrevem a Classe do Objecto Gerido informalmente (Comportamento), a identificam (Nome e Posição), definem os Pacotes que a compõem e as relações existentes com outras Classes de Objectos Geridos.

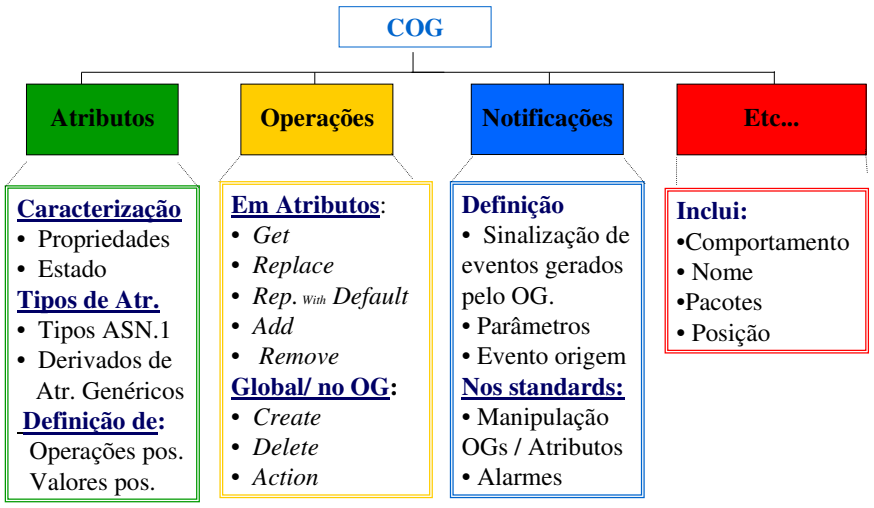


Figura 29: representação gráfica da estrutura de uma Classe de Objectos Geridos

5.1.2.1 Descrição dos Atributos

Conforme já foi referido, os Atributos são visíveis no Managed Object Boundary e caracterizam as propriedade e o estado do Objecto Gerido, reflectindo o comportamento do Objecto Gerido e a sua relação com o recurso gerido.

O tipo de um Atributo depende do Objecto Gerido que está a ser modelado. Pode ser um tipo ASN.1 simples, por exemplo um inteiro, ou pode ser um tipo derivado dum tipo genérico, nomeadamente um contador de qualquer tipo (counter ou gauge), ou um temporizador.

O valor dum Atributo pode ser armazenado, calculado a pedido ou obtido de outro sistema. Para um dado Atributo definem-se dois pares diferentes (valor, operação). O primeiro par – permitido - refere-se ao valor e à operação que podem estar genericamente associados; o segundo par – requerido - refere-se ao valor e à operação que podem ser associados ao Atributo num contexto particular.

Existem ainda Atributos de Grupo, que permitem operar os Atributos, globalmente, através de uma só operação.

EXEMPLO

Para clarificar este conceito consideremos então o seguinte exemplo: o número de cartas de interface que um router suporta pode ser considerado um Atributo da Classe

de Objectos Geridos router. Através do conjunto de valores requerido é possível limitar o número de cartas de interface, de acordo com a arquitectura do router.

5.1.2.2 Descrição das Operações

As Operações podem ser realizadas sobre os Atributos ou sobre os Objectos Geridos, sendo estas últimas denominadas de Acções.

São 7 as Operações que se podem realizar sobre os Atributos:

- **GET**- Leitura do valor do Atributo (pode ser de Grupo)
- **REPLACE** - Escrita do valor do Atributo
- **REPLACE WITH DEFAULT**: Escrita do valor do Atributo (de Grupo), que foi definido na especificação do Objecto Gerido.
- **ADD e REMOVE** – adicionar ou remover Atributos de um conjunto. Estes Atributos têm de ser do tipo conjunto de valores (***set-valued***).
- **SET-BY-CREATE** – Atribuir um valor a um Atributo quando o Objecto Gerido é criado.
- **NO-MODIFY** – Impedir alterações ao valor do Atributo, associadas ao refinamento por Subclasse.

São 3 as Operações que se podem realizar sobre os Objectos Geridos:

- **CREATE** – criação de um OG
- **DELETE** – remoção de um OG
- **ACTION** – qualquer outra operação que se realize sobre o Objecto Gerido, por exemplo Reset.

5.1.2.3 Descrição das Notificações

As Notificações são mecanismos de sinalização de Eventos Assíncronos, iniciados pelo Objecto Gerido, que ocorrem nos recursos geridos, sem que previamente se tenha recebido um pedido do Gestor. Estas Notificações podem ser específicas ou não do Objecto Gerido.

Nas normas definem-se Notificações associadas à criação e remoção de Objecto Geridos, à manipulação de Atributos e à detecção de Alarmes.

5.1.2.4 Descrição do Comportamento

A descrição do Comportamento contém a definição semântica dos Atributos, Operações e Notificações e das relações com os outros Objectos Geridos. As definições são, normalmente, efectuadas através de texto (linguagem normal). É possível utilizar linguagens de especificação (ex: SDL) desde que se altere a norma GDMO !!

5.1.2.5 Descrição do Nome

O descrição do Nome serve para identificar o Objecto Gerido.

5.1.2.6 Descrição dos Pacotes

Os pacotes servem para obter um nível de estruturação abaixo das Classes. Podem existir dois tipos de pacotes:

- **Pacotes Obrigatórios**, que têm de ser incluídos na Classe
- **Pacotes Condicionais**, que podem ou não fazer parte da referida Classe.

Por exemplo, o pacote pode ser definido em função do que se pretende gerir. Neste contexto, o pacote obrigatório poderia incluir informação de taxação e contabilidade e o pacote condicional, informação de desempenho.

5.1.2.7 Descrição da Posição

A posição indica o localização da Classe de Objectos Geridos na hierarquia de Classes, sendo indicada através da referência à Classe imediatamente Superior.

5.1.3 Propriedades

O Modelo de Informação está estruturado em Classes de Objectos Geridos, organizadas hierarquicamente. De acordo com DMI, **Top** é Superclasse de todas as Classes que forem definidas.

Este modelo possui duas propriedades fundamentais, que advêm do facto de ser orientado-a-objectos: a **Herança** e a **Compatibilidade**.

5.1.3.1 Propriedade da Herança

A propriedade de Herança define que uma Classe de Objectos Geridos pode ser definida como uma Subclasse de uma ou mais Classes. A este nível são definidas duas visões diferentes: especialização e extensão.

Na perspectiva da **especialização** uma Subclasse herda todas as suas propriedades da sua Superclasse, de forma a que, no nível mais interno, os atributos de modelização vão sendo cada vez mais específicos .

Na perspectiva da **extensão**, também se pode dizer que a Subclasse é uma extensão da Superclasse, podendo as propriedades desta Superclasse serem refinadas ou expandidas na Subclasse.

EXEMPLO

Por exemplo, a Subclasse Impressora Laser pode-se considerar uma especialização da Superclasse Impressora, sendo esta por sua vez, uma especialização da Superclasse Dispositivo.

Por outro lado, ao serem adicionadas novas propriedades à Superclasse Equipamento é possível modelizar um Comutador. A extensão das propriedades deste permite a modelização da entidade Carta de Interface.

Tipo de herança

Relativamente ao tipo de herança pode-se considerar que existe herança múltipla e estrita. Existe **Herança Múltipla** na medida em que uma Classe pode herdar características de várias Superclasses; existe **Herança Estrita** porque todos os Atributos, Operações e Notificações de uma Classe estão presentes nas suas Subclasses e não podem ser omitidos. No entanto, é possível efectuar algumas alterações, desde que se garanta a compatibilidade com a Superclasse Pai. Estas alterações permitem que:

- Os intervalos dos Atributos possam ser modificados dentro de certos limites.
- Possam ser adicionados parâmetros às Notificações e às Acções
- Os intervalos de valores possam ser alterados.
- Possam ser adicionadas, alteradas ou removidas restrições aos valores dos Atributos

Para terminar, vamos tentar descobrir em que condições se pode tirar partido da herança ? Não é nada disso que estão a pensar ! Lembrem-se que estamos a falar de gestão de redes e de sistemas distribuídos. Já sabem, pensem um pouco antes de continuarem com esta leitura.

A propriedade da herança pode ser útil em várias situações. Por exemplo, quando se pretende expandir definições das normas para incluir aspectos específicos do ambiente; quando numa nova versão de um Objecto Gerido se pretendem incluir aspectos da versão anterior, por causa da compatibilidade ou quando se pretendem incluir, para além das normas, aspectos de gestão proprietária.

5.1.3.2 Propriedade da Compatibilidade

A propriedades da Compatibilidade serve para resolver os problemas que ocorrem quando um dado Sistema de Gestão possui Objectos Geridos com Classes diferentes das que ele conhece. Assim sendo, para efectuar a sua Gestão é necessário identificar uma Classe com a qual esses Objectos sejam compatíveis.

Em que condições é que esta situação pode acontecer ? Por exemplo, a Aplicação de Gestão é dum fabricante e o Objecto Gerido está num sistema de outro fabricante, ou a Aplicação de Gestão é duma versão e o Objecto Gerido está num sistema com outra versão de gestão.

Se pensarem um pouco continuam a ter dúvidas sobre isto. Afinal de contas, esta é ou não uma Arquitectura de Gestão normalizada ? Se é, qual a razão para estas diferenças ?

As diferenças ocorrem pelo facto de existirem variações dentro de uma dada Classe de Objectos Geridos, motivadas pela existência de Pacotes Condicionais faz com que existam diversas instâncias de Objectos Geridos que satisfazem a definição da Classe. Os resultados das Notificações e as Acções tem de estar preparados para lidar com informação cuja presença é opcional. Podendo-se verificar mesmo que, uma dada Notificação pode ser definida com informação diferente, em diferentes Pacotes Condicionais.

Como gerir então estas diferenças ? A Arquitectura de Gestão OSI fornece duas soluções distintas para este problema:

- **Gestão Best-Effort**, em que o Gestor tem de estar preparado para ignorar informação extra que receba.

- **Alomorfismo**, em que só é enviada a informação que o Gestor conhece, porque o Objecto Gerido imita o Objecto Gerido que o Gestor conhece.

Gestão Best-Effort

A utilização deste tipo de Gestão pode conduzir a resultados inesperados no Gestor, nomeadamente: este dar ordens para modificar valores de Atributos e eles não serem alterados; receber Notificações estranhas ou receber atributos com valores para além dos limites definidos. Porém, se o Sistema gerido for flexível este pode enviar apenas as Notificações que o Gestor conhece.

Apesar destas limitações, pode ser uma boa opção para ambientes multivendedor.

Alomorfismo

Neste tipo de gestão, o sistema gerido actua como se o Objecto Gerido fosse uma instância da Classe C, conhecida pelo Gestor. Não existe actualmente nenhuma normalização relativamente à identificação dos objectos que possuem comportamento alomórfico e das Classes que imitam.

O procedimento actualmente utilizado é o seguinte:

- Um Gestor pede a criação de uma Classe e o sistema gerido cria um Objecto Gerido compatível; devolve a identificação da Classe que criou, colocando a Classe requerida no Atributo de Alomorfismo que herdou do Topo.

Relativamente às Notificações, não existe ainda um mecanismo para restringir as Notificações às da Classe Alomorfica, utilizando-se neste contexto a Gestão Best-Effort.

Condições de Compatibilidade

Resta então identificar as condições de compatibilidade.

De acordo com as normas, um Objecto Gerido é compatível com a classe C desde que satisfaça as seguintes condições:

- Possuir todos os Atributos definidos em C, podendo no entanto possuir Atributos adicionais.

- Suportar todas as Operações definidas sobre estes Atributos.
- Suporta todas as Acções definidas em C, podendo no entanto suportar Acções adicionais.
- Estender os parâmetros das Acções e das Notificações definidas em C, apenas quando a definição de C permite extensões.

Deve-se garantir ainda que:

- Para cada Atributo definido em C, os valores que ele toma não podem ir for a dos limites definidos.
- Comportamento dos Atributos, Acções e Notificações não pode entrar em conflito com o Comportamento definido em C.

5.1.4 Estruturas de Dados

A Gestão OSI baseia-se na existência de três árvores diferentes:

- **Árvore de Herança**, que contém as definições das Classes de Objectos Geridos, identificando as relações entre elas em termos de herança. **Árvore de Inclusão**, que também é chamada de **Árvore de Informação de Gestão** (MIT-Management Information Tree), é utilizada no processo de atribuição de nomes ao Objectos Geridos (instâncias das Classes de Objectos Geridos). A raíz desta árvore chama-se **Root.**

- **Árvore de Registo**, que contém as especificações de organismos de normalização, referentes a documentos e templates. O conteúdo desta árvore pode ser utilizado para originar outras Classes de Objectos Geridos (re-utilização). A raíz desta árvore não tem nome atribuído - **unnamed**.

5.1.4.1 Árvore de Herança

A Árvore de Herança tem início num nó que se chama **Top.** As relações de herança entre as várias Classes de Objectos Geridos são definidas a partir desta árvore. A sua estrutura é definida numa Template, a Template MANAGED OBJECT CLASS, através do camnpo REGISTERED AS.

5.1.4.2 Árvore de Inclusão

A Árvore de Inclusão tem início num nó que se chama **root**. Esta árvore é utilizada para identificar os próprios Objectos Geridos. A estrutura de Inclusão não afecta as operações sobre os Objectos Geridos. Para além da atribuição dos nomes, a Árvore de Inclusão está associada à existência dos próprios objectos: se o Objecto Geridos Superior deixar de existir, o Objecto Gerido Subordinado também deixa de existir.

A construção da Árvore de Inclusão efectua-se com base numa outra Template - a template NAME-BINDING. A estrutura de inclusão é definida através do campo que referencia o Objecto Superior. Existe também um campo que define o nome do Objecto dentro da sua Árvore de Inclusão e que é utilizado na construção do nome global do objecto.

5.1.4.3 Árvore de Registo

Conforme já foi referido a Árvore de Registo contém as especificações de organismos de normalização, referentes a documentos e templates, podendo o seu conteúdo ser utilizado para originar outras Classes. A raíz deste árvore não tem nome (**unnamed**)

Como exemplo de Classes pré-definidas que fazem parte desta árvore temos: LOG-RECORD, DISCRIMINATOR, SCHEDULER e TRANSPORT CONNECTIONS. Exemplo de Tipos abstractos pré-definidos são COUNTER e THRESHOLD. Para terminar, exemplos de Notificações pré-definidas são: ALTERAÇÃO DE ATRIBUTO, ALTERAÇÃO DE Estado, CRIAÇÃO E REMOÇÃO DE OBJECTOS GERIDOS.

5.1.5 Atribuição de nomes

Na criação de uma Classe de Objectos Geridos existe um Atributo Obrigatório, que conduz à criação do nome do OG. O valor desse Atributo só é atribuído quando o Objecto Gerido é criado.

Para identificar a Classe utiliza-se a Árvore de Herança e para identificar o Objecto Gerido a Árvore de Inclusão.

Cada Objecto Gerido está incorporado numa Árvore de Inclusão e cada Sistema tem, pelo menos, uma raíz local, subordinada a uma estrutura geral, a Árvore de Inclusão, que começa na root. Esta hierarquia de inclusão, dá origem a uma hierarquia de nomes, que gera, globalmente, nomes únicos.

A identificação do objecto pode ser realizada, de forma diferente, em dois contextos distintos:

- No contexto local do sistema gerido em que o objecto se insere utiliza-se uma **Forma Local**, em que o Objecto Gerido é identificado em relação ao objecto, Superior mais elevado na hierarquia (normal/ é *System*).
- No contexto do sistema de gestão utiliza-se uma **Forma Global**, em que se adiciona à identificação anterior a do próprio sistema local.

5.1.5.1 Nomes Locais

Método de Identificação Local

Para identificar o método de atribuição de Nomes Locais, consideremos então o exemplo da identificação dum dado porto de uma carta de interface de um Hub, representado na **Error! Reference source not found.**

Na parte mais à esquerda da figura representa-se a parte da Árvore de Inclusão que é relevante para este efeito; na parte central a identificação de cada um dos Objectos dentro da Árvore de Inclusão e na parte mais à direita a identificação local do objecto desde o início da sua Árvore de Inclusão.

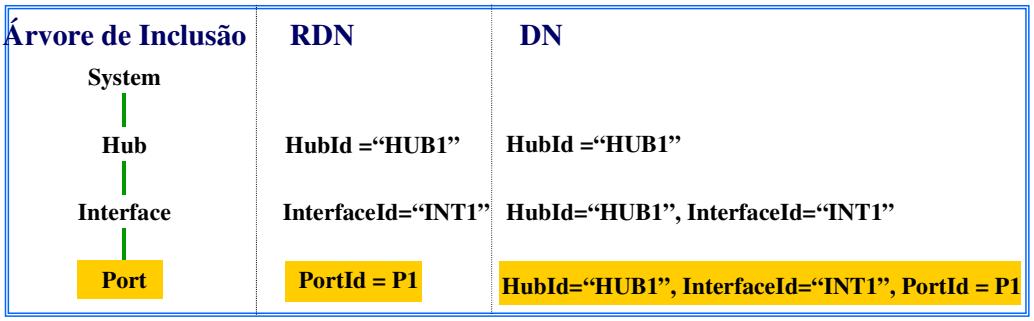


Figura 30: atribuição de nomes

Conforme já foi referido, para cada Classe de Objectos Geridos define-se **um Atributo Identificador** que serve para identificar as instâncias dessa Classe. Por exemplo, a classe Port é identificada pelo atributo PortId. Quando a Classe se instância, esse Atributo Identificador assume um determinado valor, que é o **Valor Específico do Atributo Identificador**. No caso do exemmplo anterior, o valor assumido é “P1”. Assim sendo, a identificação do Objecto em relação à sua Árvore de Inclusão - **Relative Distinguished Name** (RDN) – efectua-se com base no par de valores constituído por (Atributo Identificador, Valor Específico do Atributo Identificador). Retomando o exemplo anterior, verifica-se que a identificação do Objecto Gerido P1 é dada por PortId=”P”. Este tipo de identificação denomina-se de **Attribute Value Assertation (AVA).**

A identificação Local do Objecto Gerido é dada pelo **Distinguished Name** (DN), que é uma sequência de RDNs desde a raíz da Árvore de Inclusão até ao Objecto Gerido. No caso do exemplo anterior, a identificação local do porto P1 era dada por: HubId=”HUB1”, InterfaceId=”INT1”, PortId=”P1”.

Atribuição de nomes durante a fase de criação de OGS

Durante a realização da operação CREATE, o Gestor pede que o Sistema gerido forneça parte do nome do Objecto Gerido. Pela sua parte, o Gestro identifica o Objecto Gerido que o contém, por análise da Árvore de Inclusão, e identifica o Sistema Gerido em que o objecto se insere.

5.1.5.2 Nomes Globais

Método de Identificação Global

A atribuição de Nomes Globais também se efectua com base em AVAs (razões de consistência). Para a Classe de Objectos Geridos **System** (definida em DMI) consideram-se os seguintes Atributos:

- **SystemTitle**, que define o tipo de nome que foi registado para o Sistema Gerido.
- **SystemId**, que relaciona o nome do Objecto Gerido com outra hierarquia de nomes, ex: Directórios OSI

5.1.6 Utilização de Templates

A complexidade do Modelo de Informação deu origem à criação de modelos pré-definidos para a especificação das Classes de Objectos Geridos – **Templates** - cujo objectivo é facilitar a tarefa do projectista dum sistema de gestão deste tipo.

A especificação duma Template deve ter em consideração a sintaxe global de cada pedaço de especificação, a ordem pela qual os componentes aparecem, os componentes que podem ser omitidos ou repetidos e o que constitui cada componente. Assim sendo, uma Template é constituída por um conjunto de elementos pré-definidos, um conjunto de elementos a definir e uma label que identifica a Template e que serve para a referenciar noutras Templates A possibilidade de referenciar outras templates permite re-utilizar partes de componentes, possibilitando assim um desenvolvimento modular.

5.1.6.1 Estruturação de Classes de Objectos Geridos através de Templates

Cada Classe de Objectos Geridos está relacionada do ponto de vista hierárquico com uma ou mais Superclasses, sendo composta por um conjunto de Pacotes Obrigatórios e um conjunto de Pacotes Condicionais. As instâncias dessas Classes são identificadas através de uma outra Template – NAME_BINDING.

Template MANAGED OBJECT CLASS

Uma Classe de Objectos Geridos é definida na Template MANAGED OBJECT CLASS que constitui o núcleo central da definição da Classe. Com excepção da Template NAME BINDING, que serve para identificar os Objectos Geridos (instâncias das Classes), todas as outras são referenciadas directa ou indirecta/ a partir desta Template.

Vamos efectuar a sua descrição, com recurso ao exemplo descrito na **Error! Reference source not found.**.

A Template contém um conjunto de 5 elementos pré-definidos:

- **MANAGED-OBJECT-CLASS**, que identifica a template de definição de Classe de Objecto Gerido.
- **DERIVED FROM**, que especifica a Superclasse a partir da qual a Classe derivou e da qual herrdou as suas propriedades. Só não existe para a Classe de Topo.
- **CHARACTERIZED BY**, que lista os Pacotes Obrigatórios que são incluídos em todas as instâncias da Classe.
- **CONDITIONAL PACKAGES**, que lista qualquer Pacote cuja inclusão numa instância da Classe depende do resultado da avaliação de uma condição , que é efectuada no instante de instanciação. Um Pacote Condicional pode ser incluído como Obrigatório numa Subclasse.
- **REGISTERED AS**, que aloca um identificador global, como nome da Classe de Objecto Gerido.

```
< class-label>    MANAGED OBJECT CLASS
[DERIVED FROM     <class-label>          Referências à(s)
                  [,<class-label>]*;     Superclasse(s)
]
[CHARACTERIZED BY <package-label>        Caract. obrigatórias
                  [,<package-label>]*;   (i.e.,Atributos,oper.
                                         notif., comportamento)
]
[CONDITIONAL PACKAGE                     Caract. condicionais
                  <package-label> PRESENT IF
                  <conditional-definition> incluindo a condição
                  [,<package-label> PRESENT IF
                  <conditional-definition>]*;
]
REGISTERED AS     object identifier;     Identificador do MOC na
                                         árvore de registo
```

Figura 31: Template que descreve uma Classe de Objectos Geridos

Templates para a definição de Pacotes

A especificação da pacotes inclui a definição de cada um dos elementos que compõem uma Classe. Assim sendo:

- **BEHAVIOUR DEFINITIONS**, contém a descrição do comportamento.
- **ATTRIBUTES**, define os Atributos que estão incluídos no Pacote.
- **ATTRIBUTE GROUPS,** permite a inclusão de Grupos de Atributos no Pacote.
- **ACTION**, permite a inclusão de Acções no Pacote
- **NOTIFICATION**, permite a inclusão de Notificações no Pacote
- **PARAMETER**, que define um mecanismo genérico que se utiliza pelo facto do CMIP possuir campos que têm de ser preenchidos quando se define uma Classe.
- **REGISTERED AS**, que aloca um identificador global, como nome do Pacote.

Template NAME-BINDING

As instanciações das Classes de Objectos Geridos são definidas na Template NAME BINDING, que é constituída por um conjunto de elementos pré-definidos.

- **NAME-BINDING**, que identifica a template de definição de nomes de Objectos Geridos.
- **SUBORDINATE OBJECT-CLASS**, que especifica a Classe de Objectos Geridos em que o Objecto se insere.
- **NAMED BY SUPERIOR OBJECT-CLASS**, que define a Classe de Objectos Superior em que o Objecto Gerido se insere.

- **WITH ATRIBUTE**, que identifica o Atributo identificar do Objecto.
- **BEHAVIOUR**, que contém a descrição do comportamento.
- **CREATE** e **REMOVE**, que define se é possível criar ou remover dinamicamente Objectos Geridos (?).
- **REGISTERED AS**, que aloca um identificador global, como nome da Template.

EXEMPLO

Antes de continuarem utilizem o exemplo seguinte, referente a um Hub, para compreender os conceitos que têm vindo a ser estudados.

```
Hub  MANAGED OBJECT CLASS
       DERIVED FROM ISO/IEC 10165-2: Top;
       CHARACTERIZED BY:
            BEHAVIOUR
            ATTRIBUTES HubID GET,
                       NumberOfRelays GET,
                       TimeSinceHubSystemReset GET,
                       HubResetTimeStamp GET,
                       HubHealth GET
                       GroupMap GET;
            ACTIONS    ResetHubSystemAction,
                       RelayChangeOverAction;
            NOTIFICATIONS    HubHealth,
                       GroupRelayConfigChange,
                       ProprietaryExtensionAlarme;
       REGISTERED AS iso(1)std(0)iso8802(8802)csma(3)hubmgt(18)
6                objectclass(0)hubobjectclass(X)

HubName NAME-BINDING
       SUBORDINATE OBJECT CLASS    Hub;
       NAMED BY SUPERIOR OBJECT CLASS    ISO/IEC 10165-2: System;
       WITH ATTRIBUTES       HubID;
       BEHAVIOUR             HubBehaviour;
REGISTERED AS iso(1)std(0)iso8802(8802)csma(3)hubmgt(18)
              namebinding(0)hubname(X)

HubID ATTRIBUTE
       WITH ATTRIBUTE SYNTAX  IEEE802CommonDefinitions.uniqueIdentfier;
       MATCHES FOR Equality;
       BEHAVIO            UR HubIDBehaviour;
REGISTERED AS iso(1)std(0)iso8802(8802)csma(3)hubmgt(18)
            attribute(4)hubid(X)
```

Figura 32: Template que descreve uma Classe de Objectos Geridos

6.1 Modelo de Organização

A arquitectura de Gestão OSI baseia-se no modelo de organização Gestor-Agente, com a particularidade de que um dado sistema pode assumir ambas as funções. Por exemplo, a função pode variar dinamicamente com o tipo de acção de gestão que está a ser executada.

Em Gestão OSI existe o conceito de **Domínio** funcional ou administrativo. Num Domínio Funcional os Objectos Geridos são agrupados de acordo funções de gestão, por exemplo, por Áreas Funcionais. Num Domínio Administrativo os Objectos Geridos são agrupados de acordo com a organização a que pertencem. Em cada Domínio existem regras que são os critérios de policiamento.

Os Domínios são Classes de Objectos Geridos que, quando instanciados, também são alvo de gestão.

Para que haja uma cooperação eficiente entrs os vários sistemas, estes t~em de estar bem familiarizados com as capacidades de gestão dos outros sistemas.

6.2 Modelo de Comunicação

6.2.1 Categorias de Gestão

O modelo de Comunicação OSI incorpora três categorias de Gestão: **Gestão de Sistema**, **Gestão de Camada** e **Operação de Camada**, que se representam na **Error! Reference source not found.**.

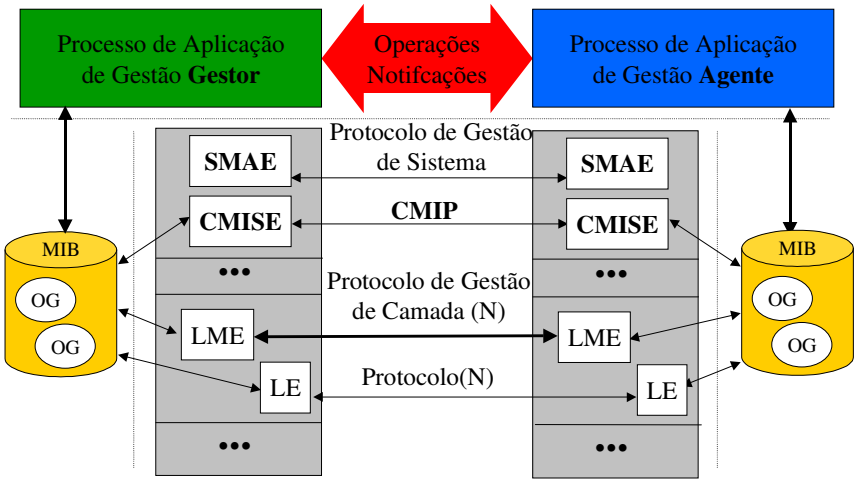


Figura 33: Representação das Categorias de Gestão

6.2.1.1 Gestão de Sistema

A Gestão de Sistema define o comportamento da gestão na perspectiva global do sistema a gerir, isto é, na perspectiva duma aplicação distribuída que involve a interacção entre o Gestor, o Agente e a MIB, que representa os recursos desse sistema. A parte da aplicação responsável pelos aspectos de comunicação é realizada pela **System Management Application Entity**. A comunicação entre as entidades distribuídas realiza-se através dum protocolo de gestão de sistema, que é suportado pelo protocolo de gestão existente na camada de aplicação do Modelo OSI, denominado de **Common Management Information Protocol** (CMIP). Este protocolo utiliza os serviços de gestão da camada de aplicação, denominados de **Common Management Information Service** (CMISE), que suportam o acesso aos Objectos Geridos remotos e a execução de operações sobre esses objectos.

6.2.1.2 Gestão de Camada

A Gestão de Camada refere-se às funções, serviços e protocolos de gestão que são específicos dum nível e que não necessitam dos serviços das camadas superiores do Modelo OSI. A Gestão de Camada realiza-se através de entidades denominadas de **(N)-Layer Management Entity**, que comunicam através do **(N) Layer Management**

Protocol. Exemplo desta categoria de gestão são os protocolos de encaminhamento, tais como o Intermediate-System to Intermediate-System routing protocol.

6.2.1.3 Operação de Camada

Para finalizar, a Operação de Camada refere-se as funções de gestão que têm de existir associadas a cada protocolo específico. Por exemplo, a definição do tamanho das janelas ou dos temporizadores do protocolo HDLC é uma tarefa de Operação de Camada.

6.3 Funcionamento geral

Para a comunicação entre o Gestor e o Agente a Gestão de Sistema baseia-se nos princípios definidos na Arquitectura de Protocolos OSI, isto a transferência de informação de gestão pressupõe a existência de uma ligação estabelecida, em cada camada de protocolo, das entidades pares que contêm o Gestor e o Agente. Assim sendo, a transferência de informação de gestão só pode ocorrer enquanto a ligação ao nível da aplicação estiver activa, isto é, após o seu estabelecimento e antes do seu cancelamento.

Durante a fase do estabelecimento da ligação são negociadas as capacidades do serviço de gestão, através da definição das **Unidades Funcionais** suportadas. Neste contexto é possível especificar, por exemplo, o método de acesso aos objectos e o tipo de respostas expectáveis.

6.3.1 CMISE e CMIP

O CMISE é um serviço orientado à ligação que utiliza os serviços do Association Control Service Element (ACSE) para gestão da ligação e os serviços Remote Operation Service Element (ROSE) para transferir as operações e notificações de Gestão.

Através do ACSE efectua-se o estabelecimento, interrupção, ou terminação da ligação de Gestão. O ROSE permite o suporte de Aplicações Interactivas, em modo não confirmado ou confirmado. É através deste serviço que se enviam comandos, (Gestor → Agente), e se enviam respostas a esses comandos ou notificações (Agente →. Gestor).

Conforme já foi referido, a transferência de informação de gestão realiza-se através do protocolo de gestão CMIP. Na sua versão inicial, este protocolo do nível de aplicação é suportado sobre a pilha de protocolos OSI. Porém, existe uma outra versão, denominada de CMIP over TCPI/IP , isto é, CMOT, em que o protocolo é suportado sobre a pilha de protocolos Internet. Nesta outra versão, o protocolo é suportado em TCP/IP, sendo normalmente utilizada a Classe de Transporte 0, num serviço orientado-à-ligação, ou a Classe de Transporte 4, num serviço não orientado-à-ligação.

Na transferência de informação de gestão, através do CMIP, é possível utilizar o seguinte conjunto de PDUs:

- **m-Get** e **m-Set**, através das quais se efectua a leitura ou escrita de Objectos Geridos.
- **m-Cancel-Get**, através da qual se cancela uma operação de GET que esteja em curso.
- **m-Action**, através se realiza uma dada acção em Objectos Geridos.
- **m-Create** e **m-Delete**, através das quais se procede à criação ou remoção dinâmica dum Objecto Gerido.
- **m-Event-Report**, através da qual se enviam Notificações assíncronas para o Gestor.

Conforme se representa na **Error! Reference source not found.**, existe uma correspondência directa entre os serviços CMISE associados à transferência de informação de gestão e as CMIP-PDUS. Ainda de acordo com essa tabela, algumas destas transferências implicam a existência dum serviço confirmado, outras não.

Operação de Gestão[1]	Serviço CMISE	Confirmação	CMIP-PDU
Notificação	M-EVENT-REPORT	O	m-Event-Report
Leitura	M-GET	C	m-Get
Escrita	M-SET	O	m-Set
Escrita valor por omissão	M-SET	O	m-Set
Execução duma acção	M-ACTION	O	m-Action
Criação dum OG	M-CREATE	C	m-Create
Remoção dum OG	M-DELETE	C	m-Delete

Tabela 11: campos ASN.1 da macro de definição de objectos em SMIv1 e SMIv2

6.3.2 Unidades Funcionais

Quando se estabelece uma ligação de gestão existe uma Unidade Funcionak Básica que lhe está associada – a Unidade Funcional de Núcleo (Kernel Functional Unit) – que permite o acesso a todos os serviços CMISE, à excepção do M-CANCEL-GET.

As Unidades Funcionais Adicionais permitem definir outras características da transferência de informação de gestão. Por exemplo, a execução de determinadas acções pode envolver vários objectos em simultâneo. Neste contexto, para uma única acção deverá ser devolvida uma resposta por cada um dos objectos afectados. Todas estas respostas têm de ser identificadas como fazendo parte da mesma acção, pelo que tem de existir um parâmetro identificador que as ligue. Este parâmetro é um **Linked-Reply** e para que esta capacidade esteja acessível é necessário que **a Unidade**

[1] Existem mais operações de Gestão, mas estas são as mais relevantes para o nosso estudo.

Funcional de Respostas Múltip las (Multiple Reply Functional Unit) tenha sido previamente negociada.

Quando se negocia e aceita esta unidade funcional é possível negociar a selecção de múltiplos objectos através da **Unidade Funcional de Selecção de Múltiplos Objectos**. Com excepção do serviço CMISE M-CREATE, é possível aplicar este critério de selecção a todos os outros e definir quais as Classes de Objectos e respectivas instâncias que se pretendem seleccionar, através da especificação de **regras de scoping**, que são aplicadas na Árvore de Inclusão. Existem quatro tipos de regras que se podem aplicar na selecção de objectos com scoping e que se encontram representadas na **Error! Reference source not found.**.

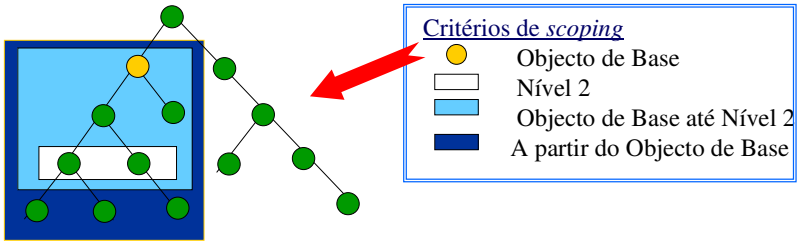


Figura 34: Aplicação de regras de scoping

Quando se efectua a selecção de múltiplos objectos, é possível aplicar dois critérios de sincronização distintos no seu processamento: **Sincronização Best-effort**, que é a opção por omissão, ou **Sincronização Atómica**, quando se pretende unicidade na realização da operação.

É possível ainda efectuar uma escolha mais detalhada, ou baseada em determinadas propriedades se, após esta Unidade Funcional estar negociada, se negociar também a **Unidade Funcional de Filtragem**[2].

EXEMPLO

Antes de continuarmos, utilizem esta tabela de exemplos para identificar outras situações em que eles sejam válidos.

Unidade Funcional	Exemplo de aplicação
Respostas Múltiplas	Suspender a actividade dos contadores de taxação
Múltiplos objectos (scoping)	Referenciar um sistema e colocar todas as suas entidades no estado administrativo locked.
Filtragem	Selecção das notificações a serem enviadas para o Gestor

[2] Existem mais Unidades Funcionais mas estas são as mais relevantes para o nosso estudo.

6.4 Modelo Funcional

Em Gestão OSI definem-se as 5 **Áreas Funcionais**, que já conhecem (certo ?), que assentam num conjunto de **Funções de Gestão**, que por sua vez são suportadas pelos serviços de gestão CMISE.

Esta situação está representada na **Error! Reference source not found.**.

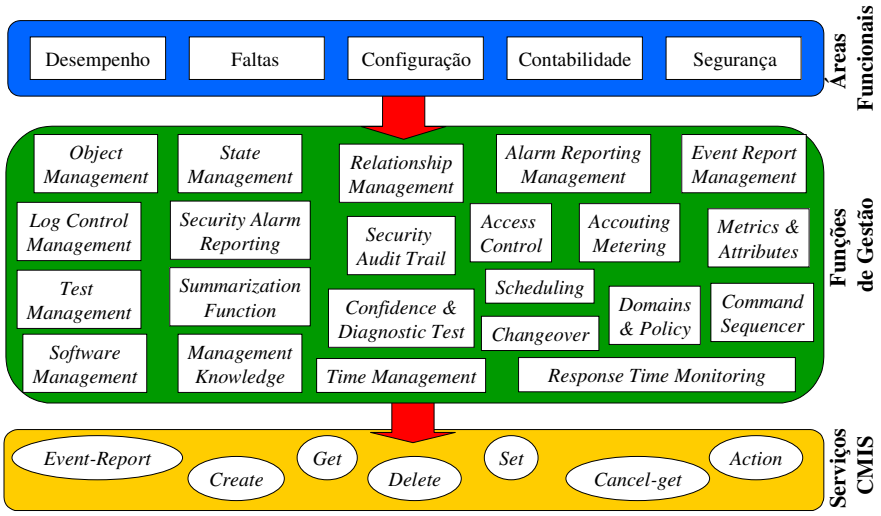


Figura 35: Arquitectura Funcional de Gestão

Como podem observar, a lista de Funções de Gestão é enorme ! Todas elas estão normalizadas, consistindo essencialmente num conjunto de Classes de Objectos pré-definidas, que se podem re-utilizar. O conjunto de Funções de Gestão está continuamente a aumentar. Vamos apenas referir algumas delas[3] , nomeadamente aquelas com que já tiveram algum contacto no TeMIP.

- **Object Management**: estrutura de notificações pré-definidas para reportar a criação ou remoção de Objectos Geridos e a modificação dos seus Atributos.
- **Log Control Management**: operações de coleccionamento e armazenamento de notificações em ficheiros de Logs.
- **State Management**: operações para gestão do estado do Objectos Geridos.
- **Alarm Reporting Management**: estrutura genérica de classificação de alarmes, de acordo com o tipo, causa provável…

[3] Se quiserem saber mais detalhes, ou conhecer as que não forem mencionadas, para além das normas, podem encontrar descrições resumidass destas funções em [1].

- **Access Control**: operações para introduzir e processar regras de controlo de acesso.
- **Scheduling**: controlo temporal das operações de gestão, com base na definição de periodos de tempo seleccionados.
- **Domain and Policy**: estabelecimento e adminstração de domínios e a definição de critérios de policiamento
- **Event Report Management**: identificação das condições que originam que um dado evento seja enviado para os destinos seleccionados.

7 Resumo Final

Nesta aula foi efectuada uma breve descrição da arquitectura de Gestão OSI. Trata-se de uma arquitectura de gestão em que Modelo de Informação é orientado-a-objectos, possuindo propriedades típicas desta aproximação, tais como, a herança e o alomorfismo, que o tornam vantagoso e flexível. Existem três estruturas de dados em que este modelo se baseia - Árvore de Herança que engloba a definição das Classes de Objectos Geridos, Árvore de Inclusão, onde de definem os Objectos Geridos, e Árvore de Registo, onde se incluem todos as Templates associadas à especificação de Classes de Objectos. Estas três estruturas oferecem um controlo flexível e facilitam o desenvolvimento de novas aplicações.

O elevado nível de estruturação do Modelo de Organização, conseguido através de uma estrutrura hierárquica de domínios, construídos à medida das necessidades das organizações, torna esta arquitectura particularmente adequada para redes complexas e de grandes dimensões.

A transferência de informação de gestão, realizada através do serviço/protocolo CMISE/CMIP. Esta transferência é orientada à ligação, e na fase inicial de estabelecimento da ligação de gestão são negociadas, entre o Gestor e o Agente, as suas características ao nível de segurança e de selecção de objectos. No âmbito da selecção de objectos existem dois conceitos particularmente relevantes: scoping, que permite identificar a parte da Árvore de Inclusão a que se pretende aceder – e filtragem, que a partir do sub-conjunto de objectos anteriormente identificado, permite selecionar objectos com base na aplicação de um conjunto de regras mais ou menos complexo. Pelo facto de ser possível seleccionar vários objectos para uma mesma mensagem enviada pelo Gestor, o Agente pode responder de forma atómica, isto é realizando a operação sobre todos os objectos, ou da forma Best-Effort, realizando a operação apenas nos objectos em que tal é possível. Adicionalmente, existe a possibilidade de enviar a resposta em várias respostas diferentes, que terão de estar sincronizadas. Na sua forma mais simples, não há possibilidade de efectuar selecção múltipla, pelo que não há necessidade de sincronozação nem de respostas ligadas. A grande vantagem deste Modelo de Comunicação reside, mais uma vez, na flexibilidade que permite suportar sistemas de diferentes complexidade, através de uma negociação adequada das características da comunicação.

Para terminar,o facto das Funções de Gestão serem normalizadas e serem constituídas por Classes de Objectos, permite re-utilizar especificações e tornar os sistemas de gestão coerentes.

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

Capítulo VI - A Rede de Gestão de Telecomunicações

Palavras chave:

1 Introdução

Efectuou-se uma breve apresentação da Arquitectura de Gestão OSI. Pela sua complexidade e elevado custo, esta arquitectura é particularmente adequada para as empresas operadoras de telecomunicações, que possuem redes de grande dimensão e complexidade. Uma outra razão, esta de natureza não técnica, que conduziu à sua aceitação junta destas empresas está relacionada com o facto desta arquitectura ter sido internacionalmente normalizada, por organismos governamentais onde estas empresas têm assento.

Vamos efectuar uma descrição da Rede de Gestão de Telecomunicações, vulgarmente conhecida pela sigla de TMN, abreviatura de Telecommunication Management Network, que teve a sua génese nesta arquitectura e que foi adoptada como solução de gestão das redes dos operadores de telecomunicações.

Vamos então começar por caracterizar o cenário, que cada vez mais, existe neste tipo de redes.

2 Enquadramento

2.1 Cenário de operação de uma empresa de telecomunicações

Hoje vamos fazer uma aula um pouco diferente. São vocês que vão começar a trabalhar. Imaginem então que vos pedem, na vossa qualidade de (quase) engenheiros, um relatório, cujo objectivo é caracterizar as actuais redes das operadoras de telecomunicações. Depois de reflectirem e de construirem a vossa visão da realidade, façam uma lista dos tópicos a incluir no relatório Não entrando em demasiados detalhes, facilmente se constrói uma lista como a que se apresenta em seguida:

Diversidade de equipamentos
Equipamento de transmissão, de comutação e equipamentos terminais
Diversidade de tecnologias básicas de rede:
Rede telefónica tradicional, SHD, ATM, SONET, X.25, RDIS, GSM, IP
Diversidade de serviços oferecidos sobre a tecnologia de base
Voz, transferência de dados, Video-conferência, Redes Privadas Virtuais
Dimensão da rede
Número de equipamentos muito elevado, dispersão geográfica

Dimensão da gestão
Número de Objectos Geridos muito elevado, Número de eventos muito elevado

Estando a rede caracterizada em termos tecnológicos, o passo seguinte é efectuar uma caracterização dos requisitos de gestão.

2.2 Requisitos de Gestão

Todas as Áreas Funcionais e todos os Níveis de Gestão têm de ser considerados. Na perspectiva das Áreas Funcionais, a descrição das funcionalidades associadas a cada área já foi efectuada com detalhe suficiente durante as primeiras aulas. Resta então, identificar as funções a desempenhar em cada um dos níveis da pirâmide de gestão: negócio, serviço, rede e elemento de rede. Está então na hora de prosseguirem o vosso trabalho de especificação.

Comecem pelo nível de negócio e imaginem-se gestores de uma empresa operadora de telecomunicações: que tipo de tarefas de gestão estariam na vossa esfera de competência ?. Já sabem, trabalhem antes de continuarem esta leitura.

No âmbito da gestão ao nível de negócio, a lista de tarefas de gestão a executar inclui:

- Planeamento e engenharia da rede,
- Alocação de domínios
- Definição de políticas administrativas, dentro do domínio administrativo ou entre domínios diferentes.
- Definição de políticas de fornecimento e de disponibilidade de serviço.
- Definição de políticas de segurança.
- Definição de estratégias de preço e tarifação.

No âmbito da gestão ao nível de serviço, esta lista é constituída por:

- Gestão do ciclo-de-vida do serviço em todas as suas fases, nomeadamente, criação, fornecimento, teste e operação
- Taxação do serviço
- Administração dos aspectos de segurança do serviço
- Fornecimento do serviço entre domínios administrativos diferentes.

A Gestão ao nível da rede é constituída por tarefas que englobam:

- Monitorização e controlo da rede
- Manutenação de circuitos
- Acesso à infra-estrutura
- Encaminhamento

Por último a Gestão ao nível do elemento de rede é constituída por um conjunto de tarefas simples, que incluem:

- Monitorização e controlo de componentes
- Inventário de componentes
- Detecção, diagnóstico e correcção de faltas.
- Supervisão de alarmes etc…

Com base neste cenário complexo, é possível resumir o problema de gestão de uma rede de telecomunicações, tendo por base o Modelo do Cubo que se representa na Figura 36.

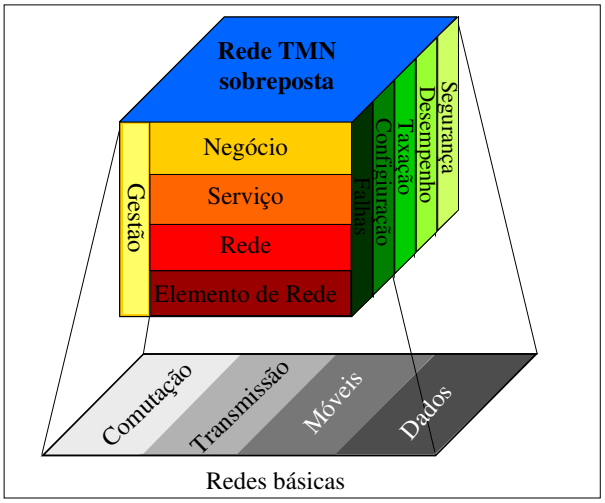


Figura 36: Dimensão da Gestão TMN

2.3 Selecção da Arquitectura de Gestão

Em poucas palavras, quais são então os requisitos fundamentais que a Arquitectura de Gestão que vai suportar a Rede TMN deverá possuir ?

- Em primeiro lugar, a **escalabilidade** que lhe permite operar em redes de grandes dimensões e complexidade.
- Em segundo lugar, a existência de interface abertas, que permitam introdusir facilmente novas funcionalidades, adaptadas aos novos requisitos, que irão sempre surgir.
- Em terceiro lugar, a existência de mecanismos eficazes de **segurança e controlo de acessos**, que estejam embebidos na própria arquitectura.
- Para terminar, o suporte da **gestão** ao nível **do serviço** e ao nível do **negócio**.

Já sabemos, pelo introdução desta aula que a escolha vai recair na Gestão OSI. Outra questão poderá, naturalmente, surgir: porque é que esta a Arquitectura de Gestão que se enquadra melhor nestas necessidades ?
Podem fazer uma nova pausa e continuar o vosso trabalho de investigação, antes de continuarem a leitura deste texto.

Eis então, a justificação !

De acordo com Hering, a Arquitectura de Gestão OSI possui um conjunto de propriedades que a tornam particularmente adequada a este cenário:

- Pelo facto de se basear numa aproximação orientada-a-objectos facilita uma análise top-down, possibilita a re-utilização de especificações e de implementações.

- OSI fornece diversas opções para a modelização de recursos complexos e para estruturar as MIBs de forma a reflectirem as relações entre os recursos. A utilização de Templates facilita fortemente o trabalho da equipa de desenvolvimento.

- As funções de Gestão OSI normalizadas são suficientemente genéricas e variadas para permitir a resolução de uma grande diversidade de problemas de gestão sem grande esforço, através da utilização de módulos de SW re-utilizáveis.

- Os Discriminadores, os mecanismos de filtragem e de scoping são métodos eficiente de transferência de funcionalidades para os Agentes, o que se torna particularmente relevante em redes de grandes dimensões.

- Para finalizar, a organização desta arquitectura de gestão suporta o conceito de domínio de gestão, o que se pode utilizar para modelizar os domínios administrativos, das operadoras de telecomunicações.

3 Arquitectura de Gestão TMN

3.1 Objectivos

TMN é uma arquitectura de camadas para a gestão das redes de telecomunicações, que foi concebida para satisfazer os seguintes objectivos:

- Existência de uma única rede, capaz de efectuar a gestão, de forma distribuída, de redes de diferentes tecnologias e fabricantes e de possibilitar a sua integração.
- Todos os aspectos de Operação e Manutenção (OAM) têm de ser considerados, incluindo o controlo e supervisão das redes de transporte, dos serviços e dos utilizadores.
- Todos os aspectos de OAM proprietários deverão ser integrados na arquitectura TMN.
- A gestão deverá englobar as funcionalidades associadas às cinco áreas funcionais e à pirâmide de gestão.
- Para efeitos de definição do fluxo de informação de gestão, deverão ser consideradas as interfaces entre domínios administrativos, isto é, entre fornecedores de serviços de telecomunicações (ou operadores de rede), ou entre estes mesmos fornecedores e o cliente do serviço.

3.2 Modelo funcional TMN

De acordo com a recomendação M.3010, onde se estabelecem os princípios da arquitectura TMN, para satisfazer estes objectivos as seguintes funcionalidades têm de ser consideradas:

- Possibilitar a transferência de informação de gestão entre ambientes TMN e ambientes não TMN.
- Possibilitar a conversão do formato da informação, de forma a que o fluxo de informação no ambiente TMN tenha um formato consistente.
- Possibilitar a transferência de informação entre diferentes locais do ambiente TMN.
- Possibilitar a manipulação da informação de gestão, de forma a que esta possua um formato inteligível para o utilizador.
- Garantir o acesso seguro à informação de gestão.

O **Modelo de Referência da Arquitectura TMN** baseia-se na existência de uma rede sobreposta a Rede de Telecomunicações que se pretende gerir, através da qual se efectuam as trocas de informação de gestão (ver Figura 37).

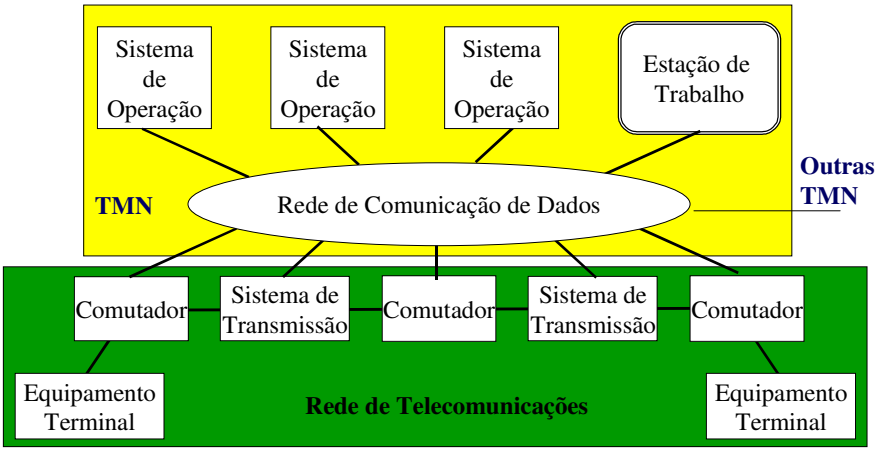


Figura 37: Relação entre a rede a gerir e a rede TMN

Conforme se representa na Figura 38, dentro deste Modelo de Referência as actividades são agrupadas em **Blocos Funcionais**, de acordo com as suas similaridades. Por sua vez, os Blocos Funcionais são constituídos por um conjunto de **Componentes Funcionais**, podendo-se verificar que uma dada Componente Funcional está incluída em diferentes Blocos Funcionais. Nos limites dos serviços fornecidos por cada Bloco Funcional situa-se um **Ponto de Referência**.

Este é um conceito novo, que vale a pena explicar um pouco melhor e que está presente em diversas arquitecturas oriundas do ITU-T (também se pode encontrar, por exemplo, na RDIS ou no ATM). Um Ponto de Referência é um conceito que serve para especificar um ponto de transferência de informação entre dois Blocos Funcionais não sobrepostos. Do ponto de vista prático, os Pontos de Referência são um dos aspectos fulcrais da Arquitectura TMN, pelos requisitos práticos que impõem na tecnologia das interfaces.

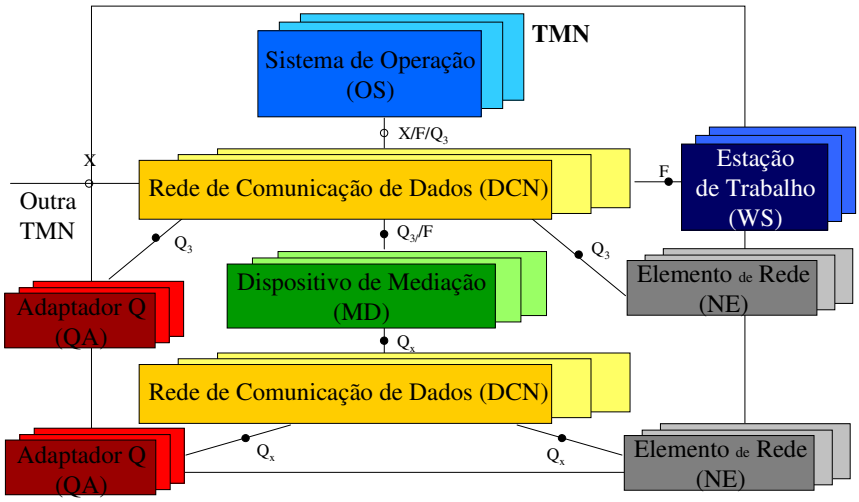


Figura 38: Modelo de Referência da Arquitectura de Gestão TMN

3.2.1 Blocos Funcionais

Existe um total de 6 Blocos Funcionais distintos, que se descrevem em seguida:

- **Estações de Traballho** (Worstations -WS), através das quais se efectua o acesso do utilizador ao ambiente TMN.
- **Funções de Sistema de Operação** (Operation System Function - OSF), que processa a informação de gestão, de forma a monitorizar e controlar as Redes de Telecomunicações.
- **Função de Comunicação de Dados** (Data Communication Function - DCF), que constitui a rede que permite efectuar o transporte de informação de gestão entre Blocos Funcionais distintos.
- **Função de Elemento de Rede** (Network Element Function - NEF), que representa o NE na perspectiva da gestão TMN, permitindo a sua monitorização ou controlo através de uma ou mais interfaces normalizadas.
- **Função de Mediação** (Mediation Function - MF), que transfere a informação entre os NE e os OS, convertendo a representação de objectos, dando suporte ao interfuncionamente entre camadas superiores de protocolo e efectuando determinados tipos de pré-processamento de informação típicos dum Agente.
- **Adaptadores Q** (Q Adaptors - QA), que transfere a informação entre os Blocos funcionais TMN e Oss ou NEs não TMN, executando acções semelhantes ao MF. Os QAs funcionam como gateways entre o ambiente TMN e o exterior.

3.2.2 Componentes Funcionais

Cada Bloco Funcional é constituído por um conjunto de Componentes Funcionais, tendo sido identificados, no ITU-T, os seguintes:

- **Função de Aplicação de Gestão** (Management Application Function), que implementa os serviços de gestão, definidos em M.3200 e que englobam as 5 áreas funcionais e os 4 níveis de responsabilidade da pirâmide de gestão. Exemplos de serviços de gestão, a este nível, são a medição de tráfego, supervisão de alarmes, gestão de pedidos de serviço etc.. Os serviços de gestão são suportados num conjunto de funções, que está definido em M.3400. Genericamente, cada função permite realizar uma dada sequência de acções sobre um conjunto de objectos geridos.
- **MIB**, que implementa o repositório de informação de gestão. Conforme já devem, saber, o método de implementação não está normalizado, mas o Modelo de Informação está normalizado, sendo baseado no modelo definido pela arquitectura de gestão OSI.
- **Função de Conversão de Informação** (Information Conversion Function), que é utilizado nos sistemas intermédios para efectuar a conversão entre Modelos de Informação.
- **Função de Apresentação** (Presentation Function), que efectua a tradução da informação entre o Modelo de Informação TMN e o formato da informação utilizador.
- **Adaptação Homem-Máquina** (Human Machine Adaptation), que efectua a conversão entre a informação da aplicação de gestão e a informação fornecida à Função de Apresentação, pela rede TMN.
- **Função de Comunicação de Mensagem** (Message Communication Function), que existe em todos os Blocos Funcionais que possuam uma interface física e que permite efectuar a transferência de informação de gestão com uma entidade par, isto é, outra Função de Comunicação de Mensagem.

3.3 Relação entre Blocos Funcionais e Componentes Funcionais

Existe uma relação simples entre os Blocos Funcionais e os Componentes Funcionais apresentados. Tentem por exemplo identificar, onde se localizam as MIBs, os Gestores, os Agentes, a conversão de informação etc... Desta vez está na hora do café….

Retomemos então com o assunto TMN. Para simplificar, a relação entre Blocos e Componentes Funcionais está definida na Tabela 12.

Bloco Funcional	Componentes Funcionais
Sistema de Operação	MIB
	Aplicação de gestão OSF (Gestor ou Agente)
	Adaptação Homem-Máquina
Estação de Trabalho	Apresentação de informação
Elemento de Rede	MIB
	Aplicação de gestão NEF (Agente)
Mediador	MIB
	Aplicação de gestão MF (Gestor ou Agente)
	Conversão de informação
	Adaptação Homem-Máquina
Adaptador Q	MIB
	Aplicação de gestão MF (Gestor ou Agente)
	Conversão de informação

Tabela 12: Relação entre os Blocos e as Componentes Funcionais

3.3.1 Pontos de Referência

Os Pontos de Referência definem a informação que circula entre Blocos Funcionais e estão divididos em 3 grupos distintos[4]:

- **Pontos de Referência q**, que existem entre Blocos Funcionais do ambiente TMN que contenham uma Função de Aplicação de Gestão. Em particular, o Ponto de Referência q_3, refere-se à ligação entre o Sistema de Operação e os Elementos de Rede e o Ponto de Referência q_x à ligação entre o Mediador e os Elementos de Rede.
- **Pontos de Referência f**, que existem na ligação da Estação de Trabalho ao ambiente TMN.
- **Pontos de Referência x**, que existem na ligação entre Sistema de Operação, em que pelo menos um deles faz parte do ambiente TMN.

EXEMPLO

Passemos à análise dum exemplo concreto de uma Arquitectura Funcional: consideremos então, ums rede em que se pretende individualizar, para fins operacionais, os níveis de responsabilidades definidos na pirâmide de gestão.

Neste contexto, vão existir os seguintes Blocos Funcionais (ver Figura 39):

- Um Elemento de Rede, para representar os recursos de rede a gerir

[4] Existem mais Pontos de Referência, mas estes são os mais siginificativos.

- Um Bloco de Mediação, para transferir a informação entre o Elemento de Rede e os Sistemas Operacionais.
- Um conjunto de Blocos Operacionais, ligados em cascata, em que cada um efectua o controlo e supervisão dum dado nível da pirâmide de gestão.

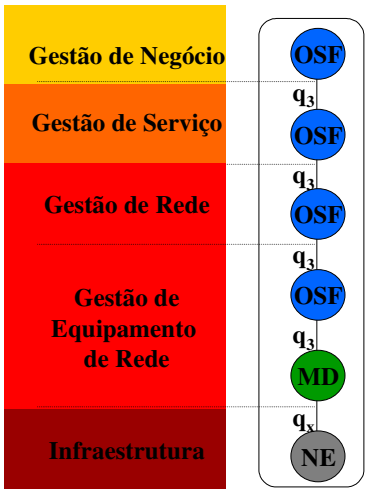


Figura 39: Exemplo de Arquitectura Funcional

3.4 Arquitectura Física TMN

A Arquitectura Física especifica a realização das funções TMN em sistemas físicos. Existem 6 tipos de entidade físicas que se relacionam directamente com os Blocos Funcionais: **Estação de Trabalho**, **Sistema de Operação**, **Rede de Dados, Elemento de Rede**, **Mediador** e **Adaptador Q**. Estas entidades podem ser configuradas de diferentes maneiras, dando origem a ambientes TMN com diferentes estruturas. Cada uma destas entidades em si pode ser gerida, o que significa que cada entidade física TMN pode ser considerada um Elemento de Rede.

Quando os Blocos Funcionais são implementados em sistemas físicos diferentes, os Pontos de Referência transformam-se em **Interfaces**, existindo, por analogia, 3 tipos de Interfaces distintas: **Q**, **F** e **X**. De todas as Interfaces, a Interface Q3 é aquela que já foi objecto de uma análise mais profunda. Neste contexto, definiu-se que esta Interface se baseia no Modelo de Comunicação OSI, em particular em CMISE/CMIP.

EXEMPLO

Para terminar, vamos analisar um exemplo de uma arquitectuta física e do seu mapeamento em termos de arquitectura funcional.

Considere-se então a existência de duas redes distintas, por exemplo a rede telefónica e a rede de dados X.25, em que se pretende implementar uma única arquitectura de gestão TMN, que satisfaça os seguintes requisitos:

- Em cada uma das redes tem de existir uma aplicação de gestão do tipo Agente, que se encarrega de efectuar algum tipo de pré-processamento sobre os dados recebidos da sua rede.

- Existe um único Gestor que coopera com os Agentes para efectuar a supervisão e controlo das duas redes
- Os Agentes e o Gestor estão fisicamente distantes.

Uma breve análise destes requisitos permite concluir que:

- A Arquitectura Funcional deverá contemplar a existência dum Bloco Funcional de Elemento de Rede, que possui funções semelhantes em ambas as redes, mas que em termos de Arquitectura Física se reparte em duas entidades distintas.
- Analogamente, existe um Bloco Funcional de Mediação, que simboliza as funcionalidades dos Agentes e que também se divide em duas entidades, quando se transita para a Arquitectura Física.
- A comunicação entre os Agentes e as redes a gerir é simbolizada pelocomunicação Bloco Funcional de Comunicação de Dados, que mais uma vez se instancia em duas entidades distintas na Arquitectura Física.
- Para representar o Gestor utiliza-se o Bloco Funcional de Sistema de Operação, que em termos de Arquitectura Física é só um.
- A comunicação entre o Gestor e os Agentes é simbolizada pelo Bloco Funcional de Comunicação de Dados que se instancia numa única entidade, na Arquitectura Física.

Assim sendo, obtêm-se as duas arquitecturas que se representam na Figura 40.

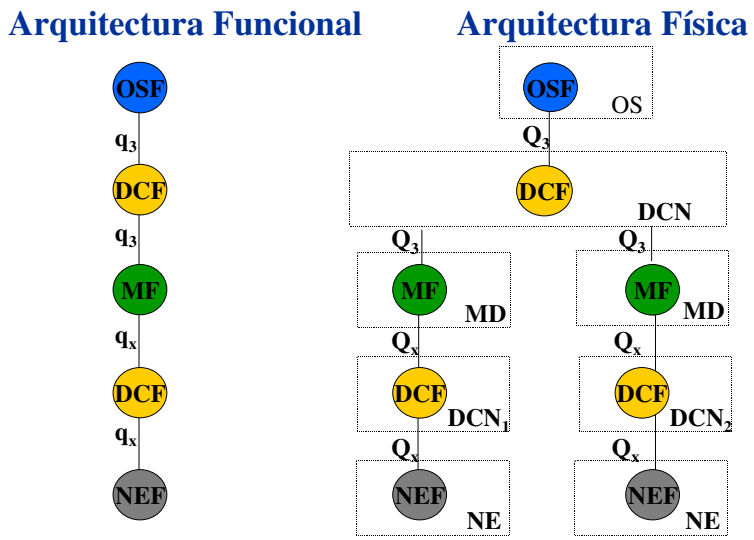


Figura 40: Exemplo da relação entre a Arquitectura Funcional e Física

4 Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

Capítulo VIII - Novas arquitecturas de Gestão

Palavras chave:.

1 Introdução

Até agora, foram estudadas duas importantes arquitecturas de Gestão: a Gestão SNMP, amplamente divulgada no âmbito das redes de dados, e a Gestão OSI/TMN, dedicada essencialmente às rede de telecomunicações. Qualquer uma destas arquitecturas se baseia na existência dum protocolo de comunicação dedicado às funções de gestão.

A perspectiva dos sistemas distribuídos é radicalmente diferente: a gestão é uma aplicação distribuída, como tantas outras, podendo assim utilizar os mecanismos de suporte à comunicação entre sistemas distribuídos já existentes. Nesta nova visão, desempenham um papel fundamental as tecnologias de objectos distribuídos, que permitem criar arquitecturas clientes-servidor flexíveis em que a lógica do negócio e os dados são encapuslados em objectos, que se podem localizar em qualquer parte do sistema distribuído.

Surge então, da parte de Object Management Group (OMG), o CORBA como outra alternativa standard a estas arquitecturas de Gestão. Esta alternativa, assume um peso significativo, tanto mais que foi criada uma aliança entre os dois organismos de normalização: ISO e OMG.

A grande vantagem da utilização do CORBA como alternativas às arquitecturas de gestão clássicas é a possibilidade de utilizar uma só arquitectura para o desenvolvimento, utilização e gestão dos sistemas, deixando assim de existirem diferenças entre o que se gere e o que é gerido.

Como concorrente do CORBA, a Microsoft lançou, pela mesma data, uma outra tecnologia de objectos distribuídos, denominada de Commom Object Model (COM), que se tornou no standard *de facto.*

Com a generalização da utilização da WEB como meio de suporte ao processamento cooperativo, novas arquitecturas de gestão surgirão:as arquitecturas baseadas na WEB. Para estas novas arquitecturas, as vantagens apontadas para a sua utilização são a possibilidade de obter soluções independentes da plataforma de suporte, que podem ser executadas em qualquer equipamento, desde que este disponha dum browser.

Três soluções surgiram neste enquadramento: a solução do IETF, a solução da SUN MicroSystems, baseada em JAVA, e a solução do Web-Based Entreprise Management (WBEM). Qualquer uma delas, revela as mais modernas tendências de gest

2 Arquitectura de Gestão baseada em CORBA

2.1 Conceitos fundamentais

Em 1992, o OMG normalizou um modelo de gestão de objectos distribuídos (OMA-Object Management Architecture), cujo objectivo é permitir que a cooperação entre objectos que seja independente da localização dos mesmos. Diferentes sub-modelos foram definidos no seu âmbito, constituindo o CORBA o core da especificação.

O Modelo CORBA é uma arquitectura Cliente-Servidor, que se pode caracterizar por um conjunto de princípios muito simples e que estão ilustrados na Figura 41.

- O Cliente não sabe a localização do objecto, o sistema operativo em que objecto é executado, nem tão pouco como é que o servidor o implementa.
- O Cliente apenas sabe a interface que o objecto do Servidor define.
- É uma arquitectura genérica em que se definem propriedades fundamentais de objectos genéricos, mas não objectos específicos, relativos a gestão de redes e de sistemas.

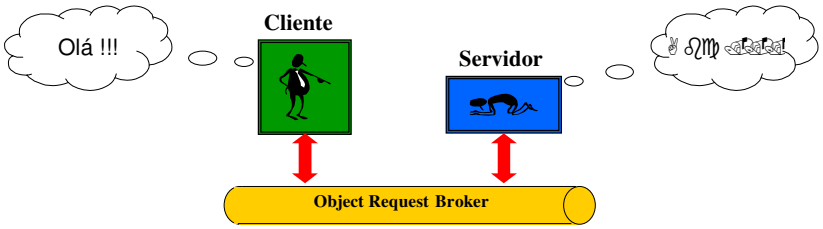


Figura 41: O Modelo CORBA

2.2 Sub-modelos

A arquitectura CORBA contempla os quatro sub-modelos (ver Figura 42) que foram definidos e que existem na arquitectura de gestão OSI: **modelo de comunicação**, **modelo de organização, modelo de informação** e **modelo funcional**.

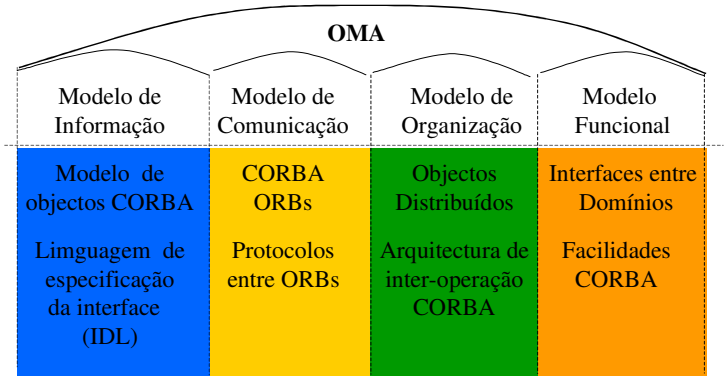


Figura 42: Sub-modelos da Arquitectura CORBA

2.2.1 Modelo de Informação

O Modelo de Informação CORBA é orientado a objectos. Para se obter uma arquitectura portável, um conceito comum de objecto CORBA é necessário para garantir a operação em ambientes heterógeneos.

Modelo de Objectos CORBA

O **Modelo de Objectos CORBA** especifica modelos muito gerais para representar coisas e conceitos, fazendo uma distinção importante entre:

- A implementação do objecto, representado por um pedaço de código que implementa as operações que podem ser realizadas com o objecto.
- A referência do objecto, quer representa a sua identidade e que é usada pelos clientes para endereçarem um objecto quando uma operação é invocada.

Uma operação é uma interacção fornecida por um objecto, cuja assinatura (**signature**) consiste no nome da operação, no tipo de resultado e numa lista de parâmetros (pode ser vazia). Adicionalmente, pode ser incluída informação relativa à lista de excepções, semântica da operação (best-effort versus at-most-once) e informação de contexto a ser transmitida durante a invocação da operação.

Contrariamente a outros modelos de objectos, em CORBA os atributos são representados são representados como pares de operações, valor em que as operações podem ser do tipo **read** e **set**, como forma de indicar respectivamente atributos de leitura e de escrita.

Interface Definition Language IDL

Uma interface define o conjunto de operações que o cliente pode invocar num objecto. Em CORBA as interfaces são representadas independentemente da linguagem de programação utilizada, utilizando uma notação que se denomina de **Interface Definition Language** (IDL).

O IDL está estruturado de forma modular, sendo constituído por 5 elemento essenciais:

- **Declarações de constantes e de tipos de dados**, incluindo a definição dos tipos de dados usualmente existentes nas linguagens de programação, isto é, tipos simples como sejam números com e sem sinal, em virgula fixa ou flutuante, caracteres, booleanos, e tipos compostos como sejam arrays, sequências e uniões.
- **Operações**, declaradas de uma forma similar à que se usa em C++, contém o nome da operação, o tipo de resultado e uma lista de parâmetros. A cláusula opcional **raise** serve para descrever excepções.
- **Atributos**, que simplificam a definição da interface, sendo indicados por pares de operação, valor.
- **Declarações de interfaces**, onde se agrupam conjuntos de constantes, dados, atributos e operações.
- **Declarações de módulos**, onde se agrupam conjuntos de interfaces.

Eis um exemplo de um módulo IDL, representado na Figura 43, que ilustra bem estes conceitos. Para peritos como vocês em programação é fácil de descobrir que existe um mapeamento simples entre as interfaces IDL e as linguagens orientadas a objectos, nomeadamente, C++ e JAVA.

```
module CosEventComm {
      exception Disconnected();
      interface PushConsumer{
            void push(in any data) raises(Disconnected);
            void disconnect_push_consumer();
      };
      interface PushSupplier{
            void disconnect_push_supplier();
      };
      interface PullSupplier{
            any pull(); raises(Disconnected);
            any try_pull(out boolean has_event);
                  raises(Disconnected);
            void disconnect_pull_supplier();
      };
      interface PullConsumer {
            void disconnect_pull_consumer();
      };
};
```

Figura 43: Exemplo dum Módulo IDL

2.2.2 Modelo de Comunicação

O Object Request Broker

O modelo de comunicação define um conceito fundamental de toda a arquitectura: **Object Request Broker**, usualmente denominado de **ORB** e representado na Figura 44.

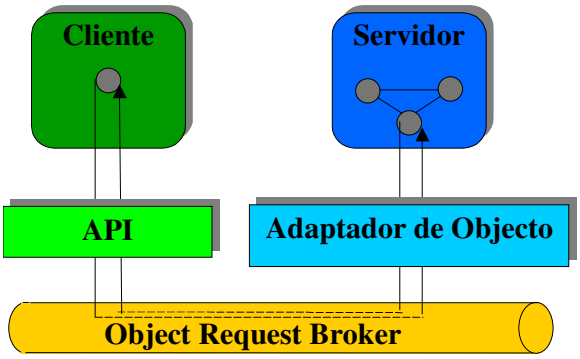


Figura 44: Comunicação Cliente-Servidor através dum ORB

Um ORB é uma infraestrutura que define a interacção entre as aplicações clientes, que requerem os serviços, e as aplicações servidoras, que os executam. A utilização de ORBs permite aos Clientes, através da utilização de **APIs normalizadas** , invocarem métodos nos servidores de uma forma transparente. O CORBA 2.0 define um conjunto de procedimentos de inicialização que permitem a um componente localizar o seu ORB. Assim sendo, quando o ORB recebe o pedido é responsável por localizar o objecto, transferir os parâmetros, invocar o método e entregar o resultado de volta ao Cliente. Para que tal seja possível, entre o ORB e o Servidor tem de existir um **Adaptador de ORB** que permita suportar o sistema operativo e a linguagem de programação que o Servidor utiliza para implementar o método.

Protocolos entre ORBs

Para permitir a interacção entre ORBs diferentes, o CORBA define um outro tipo de componentes que são os protocolos entre ORBs: **inter-ORB protocol** (ver Figura 45).

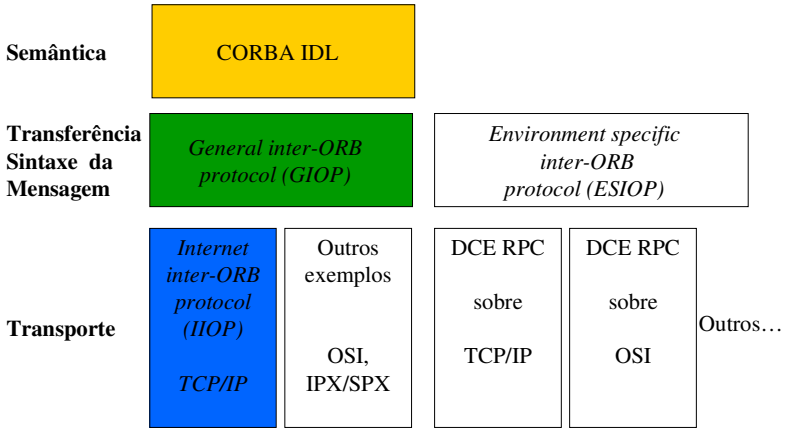


Figura 45: Arquitectura de comunicação Inter-ORB

Para inplementar esta comunicação de uma forma normalizada, a especificação CORBA define um conjunto de formatos de mensagens e de representação de dados, denominado de **General Inter-ORB Protocol** (GIOP), que se destina exactamente à comunicação entre ORBs. O GIOP foi desenhado para correr sobre qualquer protocolo de transporte orientado à ligação. Para o caso de redes TCP/IP, existe um mapeamento do GIOP para o TCP, denominado **de Internet Inter-ORB Protocol**.

Normalmente um Cliente invoca um método depois de estabelecer uma ligação, utilizando para tal a representação semântica do IDL. O GIOP encarrega-se de converter a informação IDL em tipos de dados simples, que serão transportados pelas mensagens do protocolo de comunicação, tendo em atenção todos os aspectos de implementação relacionado com o ordenamento de sequência de bytes e com os alinhamentos de memória.

O transporte da informação realiza-se utilizando um protocolo de comunicação que é especifico da arquitectura de comunicação utilizada.

2.2.3 Modelo de Organização

O modelo de organização CORBA tem por base fundamental a cooperação entre objectos distribuídos simétricos, não se verificando uma hierarquia Gestor – Agente, como na Gestão OSI ou SNMP. Dependendo dos requisitos de implementação, os objectos podem assumir o papel de agente, de gestor, ou ambos, estabelecendo-se assim um modelo de comunicação entre entidades pares.

O modelo organizacional do CORBA, na sua versão 2.0, permite o agrupamento dos ORBs de forma a representar aspectos organizacionais ou tecnológicos. O conceito de **domínio** é fundamental e serve para agrupar conjuntos de objectos, os **membros** dum domínio, que partilham entre si determinadas propriedades. Nada disto é novidade, uma vez que este conceito já vem desde a gestão OSI, certo?

Do ponto de vista de arquitectura, a existência destes domínios tem de ser transparente para a invocação do método, o que significa que o facto dos objectos do servidor existirem para lá das fronteiras do domínio do cliente nem sequer deve ser visível. Para suportar estas funcionalidades, o CORBA tem mecanismos de inter-operabilidade, que permitem realizar o **bridging** na comunicação entre clientes e servidores residentes em ORBs diferentes.

2.2.4 Modelo funcional

Conforme se representa na O Modelo Funcional CORBA define três conceitos fundamentais: **serviços**, **facilidades** e **domínio**.

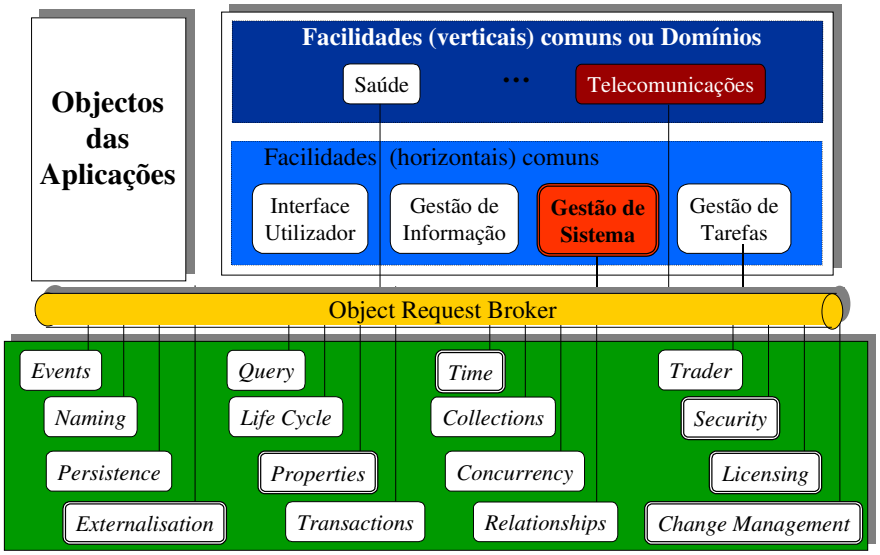


Figura 46: Modelo Funcional CORBA

Na arquitectura CORBA existe um conjunto de objectos normalizados que fornece os serviços de suporte, necessários a todos os componentes: gestão de nomes, persistência, gestão de ciclo de vida, eventos, transacções, concorrência, relações entre objectos e segurança.

O segundo nível do modelo funcional é constituído por um conjunto de objectos que implementam as **CORBA facilities**., isto é os componentes que colaboram entre si, ao nível da aplicação, definindo facilidades comuns, tais como, Interfaces Utilizador, Gestão de Informação. Gestão de Sistema e Gestão de Tarefas. Apesar deste objectos também serem normalizados pelo OMG, contrariamente os serviços, são de implementação opcional.

No nível hierárquico superior situam-se os **Domain Interfaces** ou **Objectos de Negócio**, que se situam imediatamente abaixo da aplicação, desempenhando funções de negócio específicas. Estes objectos podem representar clientes, aplicações financeiras, de gestão etc...

2.3 A arquitectura Cliente-Servidor

2.3.1 O Cliente CORBA

Um Cliente CORBA é responsável por invocar métodos que estão associados a objectos que existem no Servidor e por receber os resultados associados à sua invocação. Estes métodos podem ser previamente conhecidos ou não, dando origem a uma arquitectura flexível, que abarque as duas soluções.

Assim sendo, dum ponto de vista de arquitectura, conforme se representa na Figura 47, existe um conjunto de 4 componentes de infraestrutura que devem ser utilizados num Cliente CORBA, nomeadamente:

- **Repositório de Interface**, que contem todas as assinaturas dos objectos que estão registados, descritas em notação IDL (**metadata**). Uma interface de programação permite o acesso e o armazenamento dos metadados.
- **Stubs IDL**, que fornecem interfaces de invocação estática a objectos e aos serviços por eles fornecidos. Os métodos e os parâmetros são especificados em tempo de compilação, através da utilização de compiladores IDL.
- **Interface ORB**, que fornece funções que geralmente são úteis às aplicações. Por exemplo: conversão entre referências de objectos e strings, acesso a metadata etc..
- **Interface de Invocação Dinâmica**, que é utilizada para invocar métodos que só são determinados em tempo de execução. Existem interfaces de programação CORBA que servem para especificar os objectos do Servidor que devem ser extraídos do Repositório de Interface, os métodos que devem ser invocados, a lista de parâmetros e os resultados a obter.

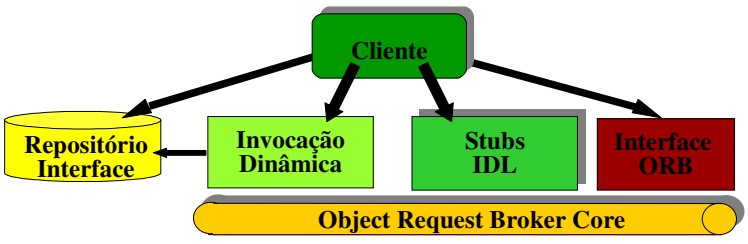


Figura 47: Arquitectura dum Cliente CORBA

2.3.2 O Servidor CORBA

Um Servidor CORBA recebe pedidos de operação, selecciona o objecto, invoca e executa a operação, utilizando os parâmetros disponibilizados pelos Cliente. Do ponto de vista de arquitectura, conforme se ilustra na Figura 48, o Servidor é constituído por 5 componentes de infraestrutura, nomeadamente:

- **Repositório de Implementação**, que armazena a informação relativa às classes de objectos que o Servidor contém, às instâncias e às referências dos objectos.
- **Stubs IDL do Servidor**, que fornecem interfaces de invocação estática aos serviços fornecidos por um objecto. São gerados de forma semelhante aos stubs IDL do Cliente, através da utilização de compiladores.
- **Interface ORB**, que fornece funções que geralmente são úteis às aplicações. Por exemplo: conversão entre referências de objectos e strings, acesso a metadata etc..
- **Interface de Invocação Dinâmica**, que é utilizada para possibilitar a invocação de métodos para os quais não existe uma descrição estática. Este componente interpreta as mensagens que recebe e, ao mesmo tempo, identifica o objecto alvo e o método invocado.
- **Adaptador de Objectos**, envia a invocação dos métodos do Cliente para o Servidor.

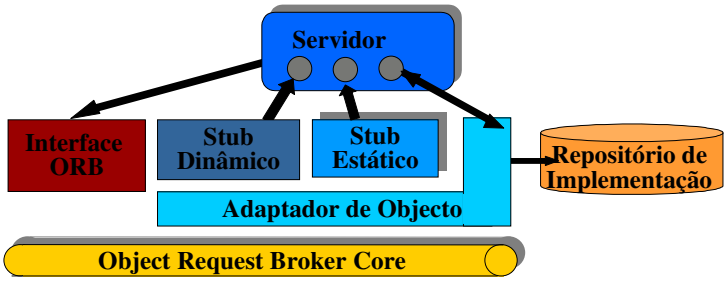


Figura 48: Arquitectura dum Servidor CORBA

2.4 A gestão na arquitectura CORBA

O Modelo CORBA define uma facilidade de gestão, denominada de System Management CORBA facilitie, que define as interfaces e serviços necessários à gestão dos seus componentes, permitindo assim efectuar a sua configuração, instalação, operação e reparação.

No seu âmbito são definidas as seguintes interfaces:

- **Instrumentação**, que permite coleccionar informação sobre a carga dum componente, tempo de resposta, recursos consumidos, débito etc..
- **Colecção de dados**, que permite coleccionar informação sobre um histórico de eventos.
- **Qualidade de Serviço**, que permite seleccionar níveis de serviço num componente, em áreas relativas à disponibilidade, desempenho, fiabilidade e recuperação.

- **Segurança**, que permite gerir a segurança do sistema (não implementar os mecanismos de segurança)
- **Gestão de Eventos**, que permite gerar, registar, filtrar e enviar notifiações para as aplicações de gestão.
- **Escalonamento**, que permite escalonar tarefas repetitivas e processadores de eventos
- **Tracking de instâncias**, que permite associar objectos de gestão a outros objectos de gestão, que estajem sujeitos a critérios de policiamento comuns.

3 Arquitecturas de Gestão baseadas na WEB

3.1 Solução do IETF

Na visão do IETF a gestão baesada na WEB irá complementar a gestão SNMP, através da utilização de Clientes e Servidores que suportem o protocolo HTTP, para dar a acesso a informação de gestão estática, dinâmica e interactiva. A utilização de browsers no Cliente irá reduzir os custo de instalação, aumentando simultaneamente a segurança. Nesta perspectiva, o SNMP continuará a ser utilizado como ferramenta de instrumentação dos sistemas e das redes.

Para que um browser dedicado à gestão aceda exclusivamente a informação de gesrão, o servidor tem de ficar à escuta num porto dedicado à gestão (280), diferente to porto 80, usualmente atribuído ao HTTP.

Um Agente SNMP que funcione conjuntamente com um servidor http permite que a informação de gestão seja acedida através dos 2 protocolos: SNMP ou http. Para determinar se um dadoa Agente suporta http utiliza-se uma MIB que anuncia as capacidades HTTP do Agente.

3.2 Arquitectura de Gestão baseada em JAVA

3.3 Arquitectura de Gestão WBEM

4 Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

Módulo III

Plataformas de Gestão

Capítulo I: Arquitecturas de Gestão e as plataformas.

Palavras chave: Gestão Isolada, Gestão Coordenada, Plataforma de Gestão.

1 Introdução

Agora que já sabem como se estrutura uma arquitectura de gestão e quais os aspectos que devem ser normalizados estão aptos a iniciar o estudo das ferramentas de gestão, vulgarmente conhecidas pelo nome de Plataformas de Gestão.

Comecemos então por identificar como é que uma plataforma de gestão se insere nas ferramentas de gestão e como se relaciona com uma arquitectura de gestão.

2 Ferramenta de Gestão, Plataformas e Arquitecturas de Gestão

Conforme foi descrito anteriormente, num sistema de gestão podem existir uma ou mais aplicações gestor. Por exemplo, podem-se considerar uma aplicação diferente para cada AF, ou uma aplicação para a monitorização e outra para o controlo, ou ainda outra solução qualquer.

Se as diversas ferramentas funcionarem de uma forma isolada não vão ocorrer transferências de informação entre elas, possívelmente vão existir diferentes interfaces utilizador e talvez até diferentes operadores de rede, cada um especializado num dada aplicação. Esta situação era muito comum à uns anos atrás, quando os fornecedores de soluções de gestão, desenvolviam soluções de gestão para os seus produtos, sem normalização e sem documentação que permitisse desenvolver interfaces. Ou seja, tratava-se de um cenário de **Gestão Isolada.**

Uma solução um pouco melhor consiste em permitir a transferência de informação entre as diferentes ferramentas e possibilitar o acesso à informação através de uma interface-utilizador única. Aparentemente é uma boa solução, mas ainda apresenta problemas: existe informação replicada pelas várias aplicações, com representações diferentes, o que pode dar origem a inconsistências. A migração para uma solução deste tipo realizou-se com um custo relativamente baixo: os produtos mantiveram-se, desenvolveram-se interfaces entre produtos e interfaces-utilizador comuns. Não sendo ainda possível atingir um cenário de Gestão Integrada, conseguiu-se um cenário de **Gestão Coordenada**.

A evolução para a Gestão Integrada implica a existência de uma aplicação de gestão que possa ser vista como uma só. Isto significa que tem de ter uma única interface utilizador, que tem de possibilitar a transferência de informação entre os elementos que a constituem e, mais importante de tudo, tem de possuir uma estrutura de informação única e ser capaz de gerir qualquer tipo de equipamento, independentemente do seu fabricante. Daí a importância das Arquitecturas de Gestão, como meio de garantir essa capacidade. Uma **Plataforma de Gestão** é uma ferramenta de Gestão que serve exactamente estes objectivos, sendo utilizada, essencialmente, no âmbito das empresas que operadoras de telecomunicações.

3 Conceito de Plataforma de Gestão

O que é vocês acham então que é uma plataforma de Gestão ? É uma aplicação de gestão que fornece diversas funcionalidades de gestão, de uma forma independente dos fabricantes.

E uma vez a plataforma feita será que não se podem acrescentar ou modificar as funcionalidades existentes ? Se pensarmos que existe uma estrutrura de informação única, uma interface-utilizador única e capacidade de interface entre os vários componentes, talvez não seja difícil de conceber essa possibilidade. Isto significa que as Plataformas para além de possibilitarem a execução da gestão da rede também fornecem ambientes de desenvolvimento, para que se possam integrar facilmente novas funcionalidades.

Como é então a arquitectura de uma Plataforma de Gestão ?. Se calhar não é muito diferente da arquitectura genérica de uma qualquer aplicação gestor de sistema de gestão. Pensem um pouco sobre isso e construam a vossa solução e comparem-na com a que vos apresento em seguida.

3.1 Arquitectura de uma Plataforma de Gestão

3.1.1 Estrutura geral

De uma forma simplista pode-se considerar que o Plataforma de Gestão é uma super-aplicação que engloba aplicações diversas, um conjunto de mecanismos de comunicação e um repositório de informação próprio. Esta visão simplista encontra-se representada na descrição mais complexa da figura 3.

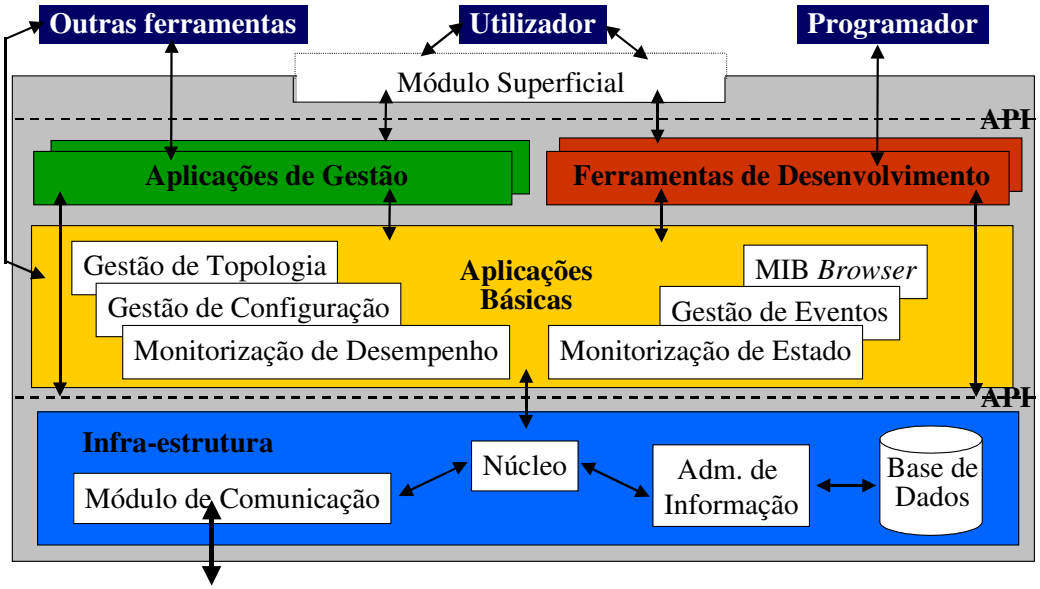


FIGURA 3: ESTRUTURA GERAL DE UMA PLATAFORMA DE GESTÃO

De acordo com esta figura, uma Plataforma de Gestão é constituída por três módulos diferentes – Módulo Superficial, Módulo de Aplicações e Módulode Infra-estrutura- que trocam informações através de APIs normalizadas.

- **Módulo Superficial**: No topo existe o Módulo Superficial que é responsável pela interface com o exterior, isto é, com os Operadores de Rede, com os programadores ou com outras aplicações, que não estejam integradas na Plataforma.

- **Módulo de Aplicações**: A nível intermédio situa-se o Módulo das Aplicações que engloba todas as aplicações existentes na Plataforma, quer sejam aplicações de Gestão, que efectuam a interface com o Operador de Rede, quer sejam aplicações que permitam o desenvolvimento de novas aplicações, quer sejam ainda aplicações básicas de gestão, isto é, Entidades Gestoras.

- **Módulo de Infra-estrutura**: Por último, na base situa-se o Módulo de Infra-estrutura, que engloba o núcleo da Plataforma, as componentes de comunicação e de administração e o repositório de informação.

Analisemos então cada um dos elementos constituintes desta estrutura, tendo em consideração os vários módulos que a compõem.

3.1.1.1 Módulo Superficial

Este módulo é responsável pela interface com os utilizadores ou com outras aplicações. Normalmente, existem três tipos de interfaces disponíveis – interfaces gráficas, interfaces de comando-linha e APIs para programação. A operação da rede pode-se efectuar através das interfaces gráficas ou das interfaces comando-linha. Para a programação podem-se utilizar APIs ou interfaces gráficas. A comunicação com outras aplicações efectua-se através de módulos desenvolvidos com base nas referidas APIs.

Esperem pelo início das aulas de laboratório para experimentar e aprender mais sobre as interfaces de uma Plataforma de Gestão.

3.1.1.2 Módulo de Aplicações

O módulo de Aplicações engloba o conjunto de aplicações de gestão. Este conjunto pode-se dividir em três grandes grupos: Aplicações Básicas, que executam as funções essencias de gestão, Aplicações de Gestão que se utilizam, em conjunto com as primeiras para a gestão de sistemas complexos e de grande dimensão e, por último, as Ferramentas de Desenvolvimento que suportam o desenvolvimento de novas aplicações.

1 Aplicações Básicas

Na grande maioria das Plataformas, existem então 6 Aplicações Básicas de Gestão: **Monitorização de Estado**, **Monitorização de Limiares**, **Gestão de Eventos, Gestão de Configuração**, **Gestão de Topologia**, **Gestão de Desempenho** e **Pesquisadores de MIBs** (MIB Browsers). Embora não estejam normalizadas, apresentam bastantes

similaridades, nos aspectos básicos, nas várias plataformas existentes. Apesar disto, existe uma tendência crescente de normalização nesta área.

Monitorização de Estado

O objectivo da Monitorização de Estado é assegurar que a informação sobre o estado dos recursos se mantêm, tanto quanto possível, actualizada no Gestor. A Monitorização de Estado realiza-se por polling (iniciativa do gestor), acedendo O Gestor a determinados atributos que caracterizam os recursos nos OGs. Podem ser utilizados os seguintes mecanismos:

- Mensagens de eco: o Gestor envia uma mensagem, utilizando um protocolo qualquer (isto é, que não seja específico para a gestão) e aguarda, durante um determinado tempo, a resposta do Agente. Um exemplo comum de utilização deste mecanismo é a utilização do protocolo ICMP pela aplicação Ping.

- Polling de estado através de Protocolos de Gestão, sem ligação: o Gestor envia uma mensagem de gestão e aguarda, durante um determinado tempo, a resposta do Agente. Como não existe ligação estabelecida, se a resposta não for recebida, o Gestor não consegue determinar o estado dos recursos e diversas situações poderiam estar a acontecer: a resposta podia-se ter perdido, o Agente podia estar demasiado ocupado para responder atempadamente ou a resposta podia estar demasiadamente atrasada, por congestão na rede. Um exemplo de aplicação deste mecanismo é o protocolo SNMP sobre UDP (a estudar posteriormente).

- Estabelecimento de uma ligação de gestão, para troca de informação de gestão, através de um Protocolo de Gestão: o Gestor estabelece uma ligação de gestão com o Agente e acede, regularmente, aos atributos dos OGs que representam o sistema gerido por esse Agente. A situação em que a ligação é terminada, abruptamente, é sinalizada como uma falha no Agente. Esta solução é mais fiável do que as anteriores, mas carece de uma ligação por cada Agente. Exemplo deste tipo de mecanismo pode-se encontrar na Arquitectura de Gestão OSI, no protocolo CMIP (que também será estudado posteriormente).

Monitorização de Limiares

O objectivo da Monitorização de Limiares é assegurar que a informação sobre Qualidade de Serviço. Na Monitorização de Limiares os valores actuais dos atributos dos OGs são comparados com valores pré-estabelecidos, valores de limiar e quando são ultrapassados dão origem à emissão de um evento. Para realizar esta tarefa é fundamental definir o seguinte conjunto de parâmetros: pontos de medida, intervalo de polling, valores de limiar e condições que originam a geração de um evento.

O correcto funcionamento da Monitorização de Limiares depende de um dimensionamento adequado destes parâmetros. Normalmente, um Operador com experiência da rede define um primeiro conjunto de parâmetros, activa o procedimento de Monitorização de Limiares e, se os resultados da operação não forem satisfatórios, procede a alterações na parametrização efectuada.

EXEMPLO

Clarifiquemos um pouco estes conceitos através de um exemplo simples: considerem a rede da RNL gerida através de uma Plataforma de Gestão. Imaginem que os gestores da rede querem avaliar a Qualidade de Serviço que prestam aos alunos. Uma das formas de o fazer é medir o tempo de atraso entre dois pontos quaisquer da rede. Para algumas aplicações, um atraso muito elevado significa que a comunicação não se está a realizar e a transferência de informação é abortada (ex: FTP), pelo que é realmente um parâmetro relevante. Nestas condições, os gestores da rede podem definir o tempo entre medidas – normalmente 15 min – e um valor máximo de atraso admissível que, quando for ultrapassado gera um evento, que será processado pela Gestão de Eventos.

Gestão de Eventos

O objectivo da Gestão de Eventos é receber e processar os eventos, quer os que são gerados internamente noutras aplicações da plataforma, quer os que são gerados pelo sistema gerido. Os eventos externos são recebidos por polling ou através da recepção de acontecimentos assíncronos (notificações ou traps). Exemplos de acontecimentos que conduzam à geração de eventos são: ultrapassagem de valores de limiar, alteração de estado, alteração de configuração, condições de erro etc..

No âmbito da Gestão de Eventos são realizadas 7 tarefas principais: recepção e processamento, armazenamento em ficheiros de logs, filtragem dos eventos a enviar ao Operador de Rede, filtragem dos acções a realizar e correlação de eventos, de forma a simplificar o diagnóstico da causa real do evento.

EXEMPLO

Retomemos o exemplo que estava a ser analisado: no Gestor de Eventos o Operador da Rede deverá definir a acção a realizar, quando receber um evento referente à situação de atraso máximo excedido, por exemplo, envio de um email para o responsável pela rede, naquele instante. Assim sendo, sempre que este evento for gerado, o responsável da operação será informado por email e, a partir desse instante poderá desencadear as acções necessárias ao diagnóstico e resolução do problema.

Imaginem que, a partir de determinado instante, a rede fica muito congestionada. Nestas condições, o atraso medido entre quaisquer dois pontos da rede torna-se muito elevado. De 15 em 15 minutos são recebidos pelo menos, em evento por cada nó da rede. Se pensarmos que em cada nó podem existir vários pontos de medida este número cresce rapidamente. Nestas circunstâncias os Operadores de Rede são inundados com excesso de informação, quer sejam eventos quer sejam os emails. Para fazer face a este problema, embora todos os eventos sejam recebidos e armazenados, os Operadores da Rede podem seleccionar um sub-conjunto de eventos a visualizar, definindo critérios de filtragem e podem definir critérios mais restritivos para a geração de emails. Por exemplo, podem seleccionar apenas uma dada região da rede e definir um número de eventos recebidos antes do email ser enviado.

Gestão de Configuração

O objectivo da Gestão de Configuração é permitir a parametrização dos recursos ou aceder a parametrização existente, utilizando as facilidades de gestão existentes na plataforma. Estas facilidades de gestão podem ser realizadas de diferentes formas, nomeadamente:

- através da transferência de mensagens de gestão entre o Gestor e o Agente.
- através das facilidades de configuração existentes no sistema a configurar, acessíveis por login remoto.
- através da utilização na Plataforma de Gestão de aplicações proprietárias dos fabricantes de equipamento.

As duas últimas opções são extremamente importantes, dada a falta de mecanismos de segurança que permitam a utilização segura da configuração através dos protocolos de gestão.

Gestão de Topologia

O objectivo da Gestão de Topologia é permitir a obtenção de informação de configuração necessária à descoberta de topologia da rede, através de um processo de auto-descoberta. Este processo pode ser executado periodicamente em background, ou pode ser uma aplicação autónoma, executada por um utilizador com privilégios. As alterações topológicas que se verificarem podem ser automaticamente actualizadas na repositório de informação da Plataforma. Uma outra alternativa consiste em armazenar essas alterações num repositório de informação próprio, sendo o repositório de informação da Plataforma actualizado apenas a pedido do Operador, por meio dos comandos adequados.

Monitorização de Desempenho

O objectivo da Gestão de Desempenho é permitir a obtenção de informação sobre o comportamento do sistema a gerir. Para realizar esta tarefa é fundamental definir um conjunto de parâmetros semelhante ao que foi definido para a Gestão de Limiares, isto é: os pontos de medida, o intervalo de polling, a duração do periodo de medição e os procedimentos a exceutar quando uma dada medição não se pode realizar, tais como repetir posteriormente ou ignorar. Normalmente, os valores das medidas são armazenados para posterior tratamento estatístico.

Pesquisadores de MIBs

O objectivo dos Pesquisadores de MIBs é possibilitar o accesso directo à informação de gestão existente nos sistemas geridos. Este tipo de aplicação possibilita apenas a visualização da informação. O seu funcionamento depende do Modelo de Informação especifíco da Arquitectura de Gestrão utilizada no sistema gerido.

Aplicações de Gestão

As Aplicações de Gestão oferecidas pelas Plataformas variam, em número, funcionalidades e complexidade. Algumas vezes constituem aplicações autónomas

que, através de uma interface adequada, podem ser integradas na Plataforma. Desta forma, é possível construir um sistema à medida das necessidades do cliente, com recurso a soluções tanto quanto possível normalizadas.

Para dar uma ideia da diversidade de Aplicações de Gestão existente, vamos analisar exemplos das mais usuais: Gestão de Problemas, Help-Desk, Correlação de Eventos, Gestão de Inventário, Gestão de Facilidades, Contabilidade e Taxação, Distribuição de SW, Gestão de Segurança, Gestão de Sistema e Gestão de Rede.

- **Gestão de Problemas**: Os sistemas de Gestão de Problemas são utilizados para documentar, identificar a acompanhar a evolução do estado de todos os problemas detectados no sistema gerido, quer pelos Operadores, quer pelos Clientes, quer ainda pela própria Plataforma de Gestão.

- **Help-Desk**: Os sistemas de Help-Desk são o ponto de contacto privilegiado entre os clientes e o fornecedor do serviço. Através destes sistemas o cliente pode efectuar reclamações, pedir informações, requisitar novos serviços etc.. O Sistema de Help-Desk é responsável por encaminhar cada pedido do cliente para a aplicação que executa os procedimentos necessários à sua satisfação, efectuando assim a ponte com o resto das Aplicações de Gestão.

- **Correlação de Eventos**: Os sistemas de Correlação de Eventos são utilizados para reduzir a enorme quantidade de eventos recebidos pelo Operador da Rede, em consequência de uma dada situação. Os eventos recebidos pela Aplicação de Gestão de Eventos são enviados para esta aplicação, que com base num conjunto de regras pré-definidas, identifica a causa do problema que esteve na génese do evento antes que este seja enviado para o Operador.

- **Gestão de Inventário**: Os sistemas de Gesrão de Inventário são responsáveis por coleccionar, armazenar e apresentar ao utilizador informação de natureza técnica ou operacional respeitante aos sistemas geridos, isto é, aos recursos físicos, SW etc…

- **Gestão de Facilidades**: Os sistemas de Gestão de Facilidades são responsáveis pelo planeamento e pela documentação das infra-estruturas, nomeadamente sistemas de cablagem, edifícios etc…

- **Contabilidade e Taxação**: A parte dos sistemas de Contabilidade e Taxação que normalmente estão integrados numa Plataforma de Gestão são responsáveis por medir a ocupação dos recursos e armazenar a informação obtida em ficheiros de logs. Os restantes aspectos são realizados no exterior da Plataforma. Alguns produtos deste tipo, dedicados a redes de telecomunicações, integram facilidades relacionadas com a gestão automática de contas, incluindo a autenticação e autorização, registo de actividade, definição de tarifas etc…

- **Distribuição de SW**: Os sistemas de Distribuição de SW podem variar desde sistemas simples, que se limitam a fazer um ftp, até sistemas complexos que incluem: a verificação das configuração das máquinas alvo, o controlo da distribuição de SW, a verificação pós-instalação e a actualizaao automática da informação de documentação, existente nos sistemas de Gestão de Inventário.

- **Gestão de Segurança**: Os sistemas de Gestão de Segurança são responsáveis pelos procedimentos de criptografia, gestão de chaves e utilização de certificados. Alguns produtos possuem APIs que escondem a complexidade destes procedimentos dos programadores comuns, possibilitando o desenvolvimento de aplicações seguras.

- **Gestão de Sistema:** Os sistemas de Gestão de Sistemas são responsáveis pela gestão de PCs, impressoras, ficheiros, serviços de directórios, servidores de email etc…

- **Gestão de Rede**: Os sistemas de Gestão de Rede são responsáveis pela gestão da rede e dos seus componentes. Normalmente estão associados a tecnologias específicas, tais como LANs ou WANs.

Aplicações de Desenvolvimento

As Aplicações de Desenvolvimento oferecidas pelas Plataformas são utilizadas para desenvolver novas aplicações, incluindo Compiladores de MIBs, ferramentas para o desenvolvimento de Agentes, ferramentas para desenvolvimento de Aplicaçãoes de Gestão. Normalmente incluem ferramentas gráficas e linguagens de programação próprias, que não são mais do que extensões a linguagens de programaçãop existentes.

3.1.1.3 Módulo de Infra-estrutura

O Módulo de Infra-estrutura engloba todos os restantes componentes que são necessários para que a Plataforma funcione, isto é, o núcleo que coordena a, o serviço de transporte de informação de gestão, que permite efectuar a transferência de informação de gestão com o sistema gerido e Repositório de Informação de Gestão, com a respectiva adminstração.

- Núcleo: o núcleo coordena a interacção entre os vários componentes da Plataforma.

- Componente de Comunicação: a componente de comunicação, permite a transferência de informação de gestão entre a plataforma e os sistemas geridos. Os protocolos de gestão a incluir dependem das Arquitecturas de Gestão que a Plataforma suporta.

- Componente de Administração de Informação: a componente de Administração de Informação é responsável pela adminstração e acesso à base de dados que constitui o Repositório de Informação de Gestão.

Depois de tanta informação, sugiro a habitual pausa para café, mas desta vez sem trabalho atribuído…

4 Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

Capítulo IX - Análise de Plataforma de Gestão comerciais

Palavras chave:.

1 Introdução

Bom, agora que já estão mais refeitos, vamos analisar, de uma forma resumida, duas das Plataforma de Gestão existentes no mercado: a **NetView**, da IBM e a **OpenView** da HP. Estas plataformas adoptam princípios diferentes e sofreram evoluções diferenciados, porque surgiram de mundos diferentes, isto é, das redes SNA/MVS e das redes IP/UNIX. A análise que se efectua está longe de ser um estudo exaustivo, ou seja, muito coisa interessante sobre estes produtos ficou, certamente, por abordar. De qualquer modo, ficam a saber que existem !! Também ficam desde já informados que existem outras plataformas, que apesar de também serem importantes, não serão abordada. O tempo é pouco….

Quanto ao TeMIP, aguradem até à próxima aula, mas entretanto vão pensando sobre o que é que acham que se pode fazer com ele ?.

2 A Plataforma de Gestão Netview

2.1 Um pouco de história…

A Plataforma de Gestão NetView surgiu a partir de um conjunto de aplicações, desenvolvidas pela IBM para a gestão da sua solução de rede, o SNA. Em 1978, surge uma interface de comandos, o Network Communications Control Facility (NCCF). Durante 1979 é lançada uma aplicação destinada a determinar problemas da rede, o Network Problem Determination Application (NPDA). Novas aplicações foram adicionadas às existentes, até que, em 1986 é formalmente anunciado o NetView. Desde então, a plataforma tem sofrido novos desenvolvimentos, para responder aos requisitos de mercado.

2.2 Um pouco sobre o NetView…

O NetView original foi desenvolvido para a gestão da arquitectura de rede da IBM, o SNA, funcionando sobre o sistema operativo MVS. A versão NetView/6000 possibilita a gestão de redes TCP/IP, requerendo uma máquinad RISC 6000, com sistema operativo UNIX. Existe uma outra versão do NetView, o NetView para WINDOWS, que se destina a gerir redes locais que suportem o protocolo de gestão SNMP.

A descrição da arquitectura do NetView encontra-se representada na figura 4. Conforme se pode verificar, o NetView insere-se dentro do MVS. A interface com as redes a gerir efectua-se através da aplicação VTAM, que interage com a rede através de um FEP (*Front End Processor*). A interface utilizador-rede é partilhada com VTAM.

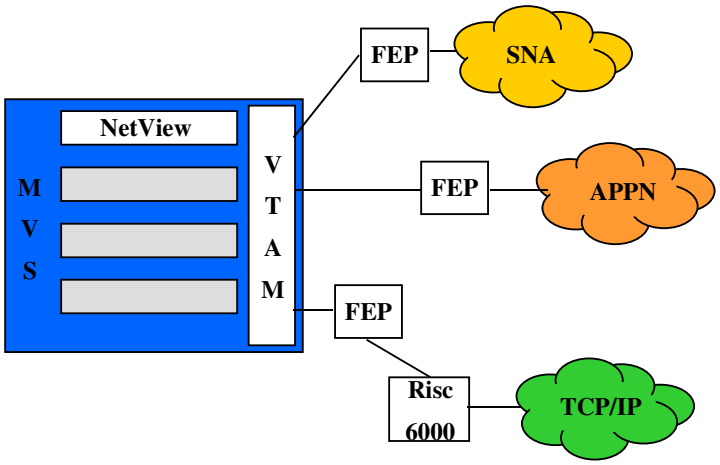


FIGURA 4: VISÃO CONCEPTUAL DA PLATAFORMA DE GESTÃO NETVIEW

A nível funcional o NetView é composto por 5 componentes principais:

- Network Communications Control Facility (NCCF), que é uma interface de comando-linha, através da qual o operador pode enviar/receber comandos MVS, NetView ou VTAM.

- o Network Logical Data Manager (NLDM), que é um monitor de sessões, através do qual é possível avaliar o funcionamento das sessões activas. Por exemplo, registando os dados para avaliação posterior, medindo o tempo de resposta, a disponibilidade etc…

- Network Problem Determination Administration (NPDA), é um monitor de hardware que mantém informação sobre os recursos físicos da rede, tais como, modens, terminais, controladores de disco, impressoras etc..

- o Status Monitor (STATMON), recolhe informação sobre os recursos, para ser apresentada sob a forma de tabelas ou sob a forma gráfica.

- o Advanced Peer-to-Peer Networking Topology and Accounting Manager (APPNTAM), que permite comunicar com redes SNA, APPN etc…

Existem outros componentes que podem fazer parte do sistema, mas estes 5 constituem o sistema básico.

3 A Plataforma de Gestão OpenView

3.1 Um pouco de história…

Originalmente o OpenView foi desenvolvido para a gestão da arquitectura de rede da TCP/IP, porque a HP estava vocacionada para esta tecnologia e para o sistema UNIX. Posteriormente evoluiu para um produto capaz de gerir outras tecnologias, muito divulgadas no mercado, isto é o SNA e o NetWare. A integração destas três tecnologias de rede, torna o OpenView um produto muito divulgado.

Actualmente o OpenView pode ser entendido com uma família de produtos, que vai integrando os novos requisitos de gestão. Neste contexto, vão surgindo diversos produtos destinados à gestão de diferentes tecnologias de rede: por exemplo, o Distributed Management Platform ,que também possibilita a gestão de redes OSI. Também têm surgido produtos mais vocacionados para a gestão de sistemas

3.2 Um pouco sobre o OpenView

A nível conceptual (ver figura 5), o OpenView é constituído por um núcelo central, o Network Node Management (NNM), que é responsável pela gestão de redes TCP/IP, quer sejam redes de grande área (WANs- Wide Area Newtork), quer sejam redes locais (LANs – Local Area Network). Para a gestão de outras tecnologias, existem os módulos apropriados, nomeadamente, o NetWare Management ou o SNA Node Manager.

O NNM funciona sobre o sistema operativo UNIX ou Solaris, existindo um produto com funcionalidades semelhantes que funciona sobre o sistema operativo MS-Dos ou Windows. A gestão é baseada em SNMP, sendo utilizado também o protocolo X, para a gestão das janelas. O NNM interage com uma base de dados INGRES ou Oracle, na versão NNM 4.0.

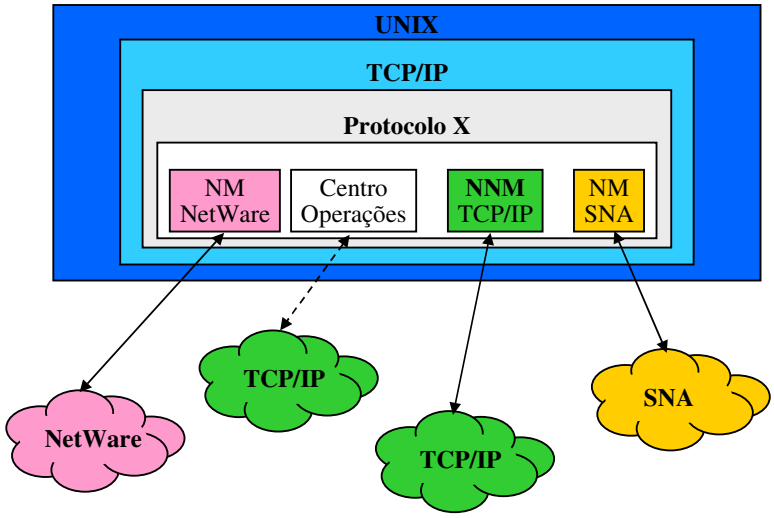


FIGURA 5: VISÃO CONCEPTUAL DA PLATAFORMA DE GESTÃO OPENVIEW

A nível funcional, o módulo NNM enquadra um conjunto variável de aplicações de gestão, que incluem aspectos de monitorização, gestão de eventos, recolha de dados e customização de aplicações, ao nível de interface utilizador.

4 Resumo final

Uma Plataforma de Gestão é uma ferramenta que oferece um simultaneamente ambiente de operação de gestão, com diversos tipos de funcionalidades e um ambiente de desenvolvimento, para a construção de novas aplicações ou modificações das existentes. A Plataforma também funciona como ferramenta de integração, ao

possibilitar a gestão de equipamentos proprietários de diferentes fabricantes, ou que se possam gerir através de diferentes arquitecturas de gestão.

Em termos estruturais uma Plataforma divide-se em três módulos: o de interface, através do qual o operador acede ao sistema de gestão; o das aplicações, onde se localizam as aplicações básica, as aplicações de gestão mais complexas e as ferramentas de desenvolvimento; e o de infra-estrutura que contém a informação de gestão e os protocolos de comunicação com os agentes.

É usual as Plataforma de Gestão incluírem as seguintes aplicações básicas: Monitorização de Estado, Monitorização de Limiares, Gestão de Eventos, Gestão de Configuração, Gestão de Topologia, Gestão de Desempenho e Pesquisadores de MIBs Embora não estejam normalizadas, apresentam bastantes similaridades nas várias plataformas. O leque de oferta de aplicações de gestão é mais vasto de deve ser dimensionado à medida das necessidades do cliente, incluindo normalmente: a Gestão de Problemas, Help-Desk, Correlação de Eventos, Gestão de Inventário, Gestão de Facilidades, Contabilidade e Taxação, Distribuição de SW, Gestão de Segurança, Gestão de Sistema e Gestão de Rede.

As Plataformas de Gestão NetView e OpenView sofreram evoluções, com o objectivo de se adaptarem aos novos requisitos de mercado. O NetView surgiu inicialmente como ferramenta de gestão de redes SNA/MVS e o OpenView como ferramenta de gestão de redes TCP/IP/Unix. Actualmente, ambas as plataformas incluem estas, e outras, arquitecturas.

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[1]: Capítulos 11 e 13

[8]: Capítulos 2, 3, 6 e 7

Capítulo III – A Plataforma de Gestão TeMIP

Palavras chave:Director, Entity, Classes, Presentation Module, Function Module, Access Module, Domínios de Gestão, Alarmes, Eventos, Logs, Filtragem, Correlação, Escalonamento.

1 Introdução

Nesta aula vamos iniciar o estudo da Plataforma de Gestão TeMIP. Depois de efectuarmos uma breve apresentação da sua arquitectura, vamos introduzir os módulos principais que a constituem. Tal como vos sugeri, espero que tenham reflectido sobre as funcionalidades que o TeMIP deve conter, pois, conforme terão oportunidade de verificar, vou ficar à espera das vossas sugestões para ir completando o cenário…

Comecemos então por um pouco de história…

1.1 Um pouco de história…

A Plataforma de Gestão TeMIP…

O TeMIP é uma Plataforma de Gestão que pode ser utilizada como ferramentas de execução, desenvolvimento e integração. Como ferramenta de execução, o TeMIP permite realizar a gestão de um sistema em rede, através de acções de monitorização, controlo e recepção de eventos. Sendo, um sistema que teve a sua genése na Arquitectura de Gestão OSI, o funcionamento de gestão está fortemente baseado em eventos. Como ferramenta de desenvolvimento o TeMIP integra a possibilidade de desenvolver novas funcionalidades, quer através do desenvolvimento de novas aplicações, quer através da customização das aplicações existentes. O seu papel como ferramenta de integração é fundamental, não só porque possibilita a gestão integrada de diversos tipos de tecnologias de sistemas, como também porque permite a integração na plataforma de outras ferramentas de gestão.

O TeMIP é uma Plataforma de Gestão Integrada que adopta os princípios da arquitectura cliente-servidor. As primeiras versões do TeMIP eram baseadas, integralmente, no sistema operativo UNIX. Actualmente, está em desenvolvimento uma nova versão, em que os Clientes podem correr em Windows-NT.

Em sistema de pequena e média complexidade, existe um único servidor – **Director TeMIP** -, com um ou mais Clientes, a executarem diferentes módulos de acesso ou de apresentação. Porém, em sistemas complexos, pode existir uma arquitectura distribuída, com vários Directores TeMIP.

2 Arquitectura Geral da Plataforma de Gestão TeMIP

2.1 Conceitos gerais

De acordo com a terminologia TeMIP (ver figura 6) um Operador acede à plataforma através de uma Interface de Gestão, para executar acções de monitorização e controlo. O acesso ao TeMIP efectua-se através de um Director TeMIP (gestor) que interage com uma Entidade TeMIP (Agente e MIB associada), para aceder à informação de

gestão. Esta interação realiza-se através de um protocolo de gestão, que seja adequado à tecnologia associada à Entidade TeMIP.

Estruturalmente, um Director TeMIP divide-se em três módulos distintos – Form, Function e Access -, que comunicam com os módulos adjacentes através de Interfaces de Gestão.

- **Form**: O módulo Form é responsável pelas funções de apresentação da informação de gestão ao utilizador, pelo que, se designaráa por módulo de Apresentação (Presentation Module – PM).

- **Function**: O módulo Functiom realiza as funções de gestão, quer sejam as aplicações básicas, quer sejam aplicações mais complexas.

- **Access**: O módulo Access é responsável por fornecer o acesso a Entidades TeMIP, suportadas em diferentes tecnologias de rede.

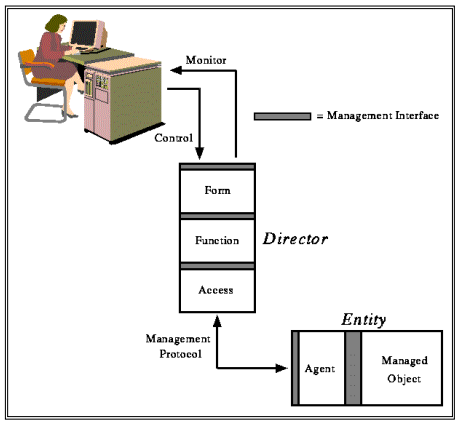


FIGURA 6 VISÃO CONCEPTUAL DO MODELO DE GESTÃO TEMIP.

Estruturalmente, uma Entidade TeMIP é constituída pelo Agente e pelos OGs , pelos quais o Agente é responsável. O Agente funciona como um intermediário, que responde a directivas de gestão, de forma a realizar acções de monitorização ou controlo, e envia reporte de eventos.

2.2 Modelo de Informação do TeMIP

As entidades no TeMIP estão estruturadas hierarquicamente, em Classes. No nível mais elevado da hierarquia, existe uma entidade global, **Global Entity**, da qual podem derivar várias classes, **Child Entities**.

De forma a garantir que as várias entidades são identificadas correctamente, o TeMIP mantém um repositório de informação global – **TeMIP Namespace** - que contém os nomes de todas as Entidades Globais e sub-entidades. Antes de ser utilizada pelo TeMIP, qualquer entidade tem de ser registada neste repositório, com o nome distinto de todas as outras entidades que lá existam.

Cada entidade é caracterizada por vários conjuntos de atributos, **Attribute Partitions**, que incluem:

- **Characteristics**: que definem o comportamento da entidades, relativamente a parâmetros que estas não podem modificar, por exemplo: ter um dado processo de recolha de alarmes activo.
- **Counters:** que definem o comportamento da entidades, relativamente a parâmetros que não podem ser alterados pelo TeMIP, por exemplo: número de pacotes recebidos.
- **Status:** que define o estado da entidade.
- **Statistics:** que fornecem informação sobre o desempenho, faltas, fiabilidade da entidade num dado período de tempo.
- **Identifiers:** que identificam a entidade dentro da sua classe e a localizam para efeitos de gestão.
- **Reference:** que fornecem, exclusivamente para o TeMIP, informação referente à entidade.

2.3 Arquitectura funcional de um Director TeMIP

Analisemos mais detalhadamente (figura 7) a Arquitectura Funcional de um Director TeMIP, a funcionar sobre o sistema operativo UNIX.

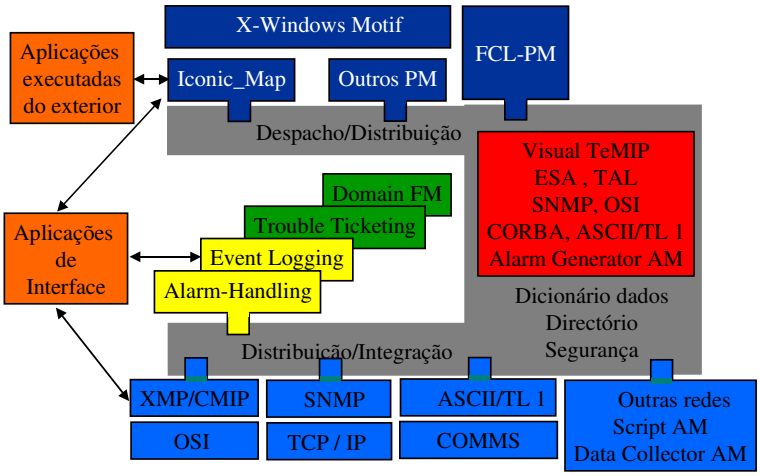


FIGURA 7 ARQUITECTURA DE UM DIRECTOR TEMIP BASEADO EM UNIX.

Comparando a arquitectura representada com a arquitectura de uma Plataforma de Gestão genérica é fácil de concluir que são bastantes semelhantes. Todas as aplicações se localizam em redor de um bloco central que é o **núcleo** da plataforma. Este núcleo é responsável por coordenar o funcionamento do TeMIP, efectuando o despacho, a distribuição, a integração da informação e a administração do repositório de informação de gestão.

Na parte inferior da figura localiza-se um conjunto de aplicações que constituem os diferentes módulos de acesso que o TeMIP suporta. Juntamente com o núcleo, estes módulos constituem a infra-estrutura da plataforma. Os módulos de acesso são

responsáveis pela interacção com equipamentos de rede de diversas arquitecturas, nomeadamente redes OSI, com capacidades de gestão CMIP, e redes TCP/IP com capacidades de gestão SNMP. Também existe a possibilidade de aceder a equipamentos utilizando outros tipos de interface, por exemplo, uma interface ASCII ou CORBA.

Na parte intermédia situam-se todas as aplicações de gestão que integram a plataforma e todas as aplicações, que embora não sejam parte integrante desta, podem cooperar com o TeMIP na gestão da rede. No âmbito do TeMIP existem aplicações básicas de gestão, tais como a aplicação de gestão de alarmes (**Alarm-Handling_FM**) ou a aplicação de gestão de eventos (**Event-Handling_FM**), representadas na figura. Existem ainda aplicação não básicas que são necessárias à operação da rede, nomeadamente a aplicação de gestão de problemas (**Trouble Ticketing**) e aplicações de suporte ao desenvolvimento de novas aplicações de gestão, por exemplo, o compilador da linguagem **Visual TeMIP** e interface gráfica de desenho de aplicações, **Graphical Ascii Toolkit** (**GAT**).

Por último, ao nível superior existem todas as aplicações que interagem com o operador, quer através de uma interface gráfica, baseada em X windows/ Motif, como por exemplo a aplicação **Iconic_map_PM**, quer através de uma interface comando-linha, como a aplicação **Fcl_PM**. Em termos da estrutura de uma plataforma genérica, estas aplicações fazem parte do chamado módulo superficial.

No manual de introdução ao TeMIP existem mais detalhes sobre cada destes módulos. Para além disso, no decorrer das próximas aulas, alguns deles serão descritos com mais pormenor. Por agora é importante os conceitos fundamentais que estão em causa.

3 Funcionalidades do TeMIP

3.1 Conceitos fundamentais

O TeMIP tem a sua génese na Arquitectura de Gestão OSI, pelo que a sua funcionalidade está orientada para a visão que esta arquitectura tem da gestão. Neste contexto podem-se identificar duas características fundamentais:

- O TeMIP suporta **Domínios de Gestão**, permitindo a organização das entidades que compõem os sistemas em rede, em grupos definidos com base em critérios estabelecidos pelo Operador da Rede.

- O TeMIP recolhe a informação de gestão, essencialmente, através do método de reporte de acontecimentos assíncronos, recebendo informação sobre os **Alarmes** e os **Eventos** que surjam nas entidades que constituem o sistema gerido. A informação recolhida pode ser armazenada em **Logs**. A recolha de informação por polling, embora se possa realizar, não constitui a base de funcionamento da Plataforma de Gestão, isto é, não é através deste método que se procede à avaliação do estado do sistema em rede.

Talvez esteja na altura da nossa habitual pausa para o café. Aproveitem-na para reflectir um pouco sobre estes conceitos. Para ajudar a vossa reflexão, pensem num sistema em rede de uma grande organização, espalhado por diversos locais

geograficamente distantes, com uma grande quantidade e diversidade de equipamento. Pensem também que os recursos humanos, disponíveis para a gestão são limitados, assim como o seu conhecimento sobre os sistema a gerir e que, para além de tudo podem estar dispersos geograficamente. Nestas condições, qual será, do vosso ponto de vista, a resposta a cada uma das seguintes questões:

- Como é que se deveriam organizar os Domínios de Gestão ?
- Que tipo de informação se deveria recolher nos alarmes e nos eventos ?

3.2 Domínios de Gestão

3.2.1 Conceito de Domínio e de Sub-Domínio

Do ponto de vista conceptual, um Domínio de Gestão é um conjunto de entidades, que se agrupam, sendo geridas de uma forma autónoma. O objectivo desta divisão é aumentar a eficiência da gestão. A forma como o agrupamento de entidades se realiza está dependente da estrutura do sistema em rede e da estrutura da própria organização, em que este se insere. Assim sendo, não há critérios pré-estabelecidos para construir Domínios de Gestão. O Operador da Rede pode optar por agrupar as entidades de acordo com a sua localização geográfica, com o tipo de equipamento, ou mesmo com a estrutura da organização.

É possível construir uma estrutura hierárquica de Domínios. Neste tipo de estrutura, um dado domínio divide-se em um ou mais **Sub-Domínios**: o Domínio tem acesso à informação de gestão de todas as entidades que lhe pertencem e, cada Sub-Domínio, só tem acesso à informação de gestão das suas entidades.

3.2.2 Suporte de Domínios no TeMIP

No TeMIP as Entidades Globais podem ser agrupadas em Domínios, podendo uma dada entidade fazer parte de um ou mais domínios, simultaneamente. Por sua vez, os Domínios podem ser sub-divididos em Sub-Domínios. Em termos de sistema, cada um deste domínios é representado como uma Entidade TeMIP, que é registada no TeMIP Namespace e gerida pelo módulos funcionais adequados, nomeadamentes:

- **Domain_FM**: que é responsável pelo agrupamento das entidades.
- **Graphics_FM**: que é responsável pela representação gráfica dos Domínios, na Aplicação Iconic_map.

Quando os Domínios de Gestão são representados graficamente na aplicação Iconic_map, cada entidade é representada por um símbolo e o conjunto de entidades que constituem um Domínio é representado através de um **mapa** da rede. Neste mapa, as entidades podem estar arranjadas numa perspectiva topológica ou lógica, podendo

ou não existir simbolos gráficos ou texto, para auxiliar a representação. Podem-se utilizar shortcuts para facilitar a navegação entre Domínios e/ou Sub-Domínios.

Na aplicação Iconic_map também é possível definir diferentes níveis de visualização num dado mapa, através da criação de **Layers**. O objectivo dos Layers é reduzir a quantidade de informação apresentada ao Operador da Rede, isto é, a sua utilização não altera em nada a informação de gestão que é recolhida no Domínio associado.

A informação sobre a constituição e estruturação dos Domínios de Gestão também pode ser consultada, através da aplicação Fcl_PM.

3.3 Alarmes, Eventos e Logs

3.3.1 Conceito de Alarme, Evento e Log

Em termos conceptuais, os alarmes e os eventos representam informação que é enviada para o Gestor, assíncronamente, quando o Agente é notificado da ocorrência de determinado acontecimento no sistema gerido. Um alarme serve apenas para descrever situações de mau-funcionamento enquanto que, um evento, também descreve outro tipo de situações, nomeadamente alterações de estado, criação ou remoção de entidades.

Um log é um ficheiro onde se podem armazenar os eventos recebidos (incluindo alarmes), de forma a que posteriormente sejam processados e analisados estatisticamente. Para se conseguir analisar a informação obtida é necessário utilizar um formato normalizado, para reportar os alarmes e os eventos, que é definido por cada Arquitectura de Gestão.

A quantidade e o tipo de situações que causam a geração de eventos ou alarmes depende fortemente da Arquitectura de Gestão associada ao sistema gerido. Se esta arquitectura for orientada ao reporte de eventos – caso da Arquitectura de Gestão OSI - a descrição do estado da rede efectua-se, maioritariamente, com base nestes alarmes e eventos. Neste contexto, o número de situações que conduzem a sua geração é muitissimo elevado. O elevado número de eventos que podem ser recebidos neste tipo de Arquitecturas, faz com que estas tenham de suportar mecanismos que permitam lidar, de forma eficiente, com grandes volumes de informação (eventos). Estes mecanismos incluem processos de **filtragem**, **escalonamento**, **armazenamento** e **correlação**, mais ou menos complexos.

Por outro lado, se a arquitectura de gestão for orientada ao polling – caso da Arquitectura de Gestão Internet - a descrição do estado da rede efectua-se, essencialmente, com base nas respostas recebidas dos Agentes. Sendo o papel do reporte de eventos diminuto, o número de situações que os geram são muito mais reduzidas. Neste tipo de Arquitecturas, os processos anteriormente mencionados não são tão importantes.

Um exemplo simples, ajuda a ilustrar estes conceitos. Considerem que, num dado instante se verifica a falha numa ligação física entre dois nós, entre os quais estava a decorrer uma transferência de ficheiros. Esta falha física vai implicar perdas de informação, em cada uma das camadas da arquitectura de protocolos. No primeiro

caso, estas perdas, e tudo o que correr mal nos sistemas, são reportadas através de alarmes. Para diagnosticar rapidamente a situação, será conveniente filtrar toda a informação em excesso e poder correlacionar a informação recebida, de forma a identificar que alarmes diferentes são motivados pelo mesmo problema. No segundo caso, só se reporta a quebra de ligação física porque as perdas serão contabilizadas quando se efectuar o polling aos sistemas.

3.3.2 Suporte de Alarmes e de Eventos no TeMIP

Da mesma forma que um sistema em rede é sub-dividido em vários Domínios de Gestão, também é possível definir, dentro de um domínio, diferentes visões sobre a actividade dos Alarmes e dos Eventos, que permitem estabelecer o âmbito de intervenção de cada equipa de operadores. Esta funcionalidade é suportada por duas entidades TeMIP: os **Operation Context**, associados aos Alarmes, e os **Event Logs**, associados aos Eventos.

3.3.2.1 Suporte de Alarmes

Os alarmes recebidos num Director TeMIP podem ser recolhidos e armazenados por diferentes Operation Contexts. Do ponto de vista de sistema, um Operation Context é uma Entidade TeMIP, que possui determinadas propriedades que identificam as regras a aplicar e o estado do processo de recolha de alarmes. A gestão de um Operation Context pode ser efectuada através da janela **Operation-Context-Control-Window**. Através de execução de determinadas directivas é possível suspender ou activar a colecção de alarmes de Operation Context especificado.

Existem quatro possibilidades diferentes de visualizar, graficamente, os alarmes que estão a ser recolhidos por um dado Operation Context: Iconic_map, **Alarm-Handling-Window**, **Full-Alarm-Window** e **View-Alarm-Window**.

A forma mais simples consiste em utilizar o serviço de Notificações e visualizar o alarme na aplicação Iconic_map. Esta visualização consiste na alteração de cor do símbolo que representa a entidade que gerou o alarme ou, numa visão mais global, o domínio onde o alarme ocorreu.

A visualização dos Alarmes em tempo-real é efectuada através da Aplicação Alarm-Handling Window. Na sua janela principal, esta aplicação fornece uma descriçãogeral de todo o processamento de alarmes, estado dos Operation Contexts, estatísticas sobre os alarmes recebidos e uma descrição muito reduzida dos alarmes recebidos.

Para obter a descrição completa de alarme deverá ser utilizada a janela Full-Alarm-Window. A descrição obtida corresponde à estrutura de reporte de alarmes, definida pelas normas OSI.

A visualização de todos os alarmes armazenados é efectuada através da aplicação View-Alarm-Window. A informação fornecida é de natureza estática, pelo que se deve refrescar a janela sempre que se quiser obter informação actualizada.

3.3.2.2 Suporte de Eventos

O modo como os eventos são suportados no TeMIP é muito semelhante ao modo como os alarmes são suportados. De uma forma abreviada, verifica-se que:

- os eventos recebidos podem ser recolhidos e armazenados por diferentes Event Logs.
- Um Event Log é uma entidade TeMIP, que pode ser gerida através janela **Event-Log-Panel-Window**.
- Os eventos podem ser visualizados na aplicação Iconic_map, de uma forma semelhante aos alarmes.
- A visualização estática é possível através da janela **Event-Records-View-Window**, não sendo possível efectuar visualização dinâmica.

4 Resumo final

O TeMIP é a Plataforma de Gestão de Digital/Compaq e tem a sua génese na arquitectura de gestão OSI, pelo que recebe informação, primordialmente, através do método de reporte de eventos.

O TeMIP estrutura a informação utilizando uma metodologia orientada a objectos, em que cada entidades (objecto) possui um conjunto de características, operações que pode executar e notificações que pode emitir. Cada entidade é identificada univocamente no espaço de nomes TeMIP.

A Plataforma é constituída por um Director TeMIP, que funciona como Gestor, ou por vários, em casos de arquitecturas distribuídas, complexas e redundantes a falhas. Tal como uma plataforma genérica, a plataforma é constituída por três módulos - o de Acesso (AM), o que contém as funções de gestão (FM) e o que se encarrega da apresentação dos resultados (PM) – que comunicam utilizando o núcleo.

A nível funcional a plataforma permite agrupar as entidades em domínios, gerir alarmes e eventos e monitorizar ou controlar entidades. Este tipo de acções está disponível através de interfaces gráficas ou de linha de comandos.

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[9]: Leitura geral

Capítulo IV - Gestão de Alarmes e de Eventos.

Palavras chave: Ciclo-de-vida, Gestão de Problemas, Outstanding, Acknowledge, Terminate, Operation Context, Discriminator Construct, Tap Filter, Heap Filter, Threshold Filter, Transient Filter.

1 Introdução

Na aula passada foi-vos apresentada a Arquitectura do TeMIP e as suas principais funcionalidades. Conceitos como estruturação em domínios, gestão de alarmes e eventos, monitorização e configuração de recursos já não são novidades para vós. Porém, as imensas potencialidades que a Plataforma apresenta neste contexto ainda não foram estudadas. É exactamente esse o assunto da aula de hoje.

Comecemos então o estudo da gestão de alarmes, analisando de uma forma mais aprofundada a descrição de um alarme.

2 Gestão de alarmes

2.1 Problemas associados à descrição de um alarme

Conforme foi referido na aula anterior, um alarme contém informação referente a uma situação de mau-funcionamento ocorrida num qualquer sistema gerido. É com base na descrição do Alarme que o Operador da Rede efectua o diagnóstico e conduz a resolução do problema que esteve na sua origem. Uma acção eficiente do Operador de Rede depende de duas características fundamentais da descrição do alarme:

- Possuir um formato normalizado, que possa ser utilizado independentemente da tecnologia do sistema em rede que gerou o alarme.
- Conter toda a informação necessária e suficiente para que o Operador possa efectuar a identificação e resolução do problema que originou o alarme.

É exactamente este último aspecto que eu gostaria que analisassem por vós, antes de continuarem esta leitura. Pensem então, que tipo de informação é que deverá estar contida na descrição de um alarme. Recordem que esta informação faz parte da janela Alarm-Full-Window, que foi apresentada na aula anterior.

2.2 Informação necessária à descrição de um alarme

De uma forma muito genérica, existem quatro tipos de informação que devem estar incluídos nesta descrição e que permitem identificar: a origem, o tipo falha e a gravidade da falha, assim como o estado de resolução da mesma.

- A origem da falha é constituída pela identificação do sistema em que o alarme ocorreu, mais especificamente do objecto que o gerou, e pela data de ocorrência.
- O tipo de falha é identificado através da indicação do tipo de alarme e do diagnóstico, efectuado pelo Agente, sobre a causa provável. Este diagnóstico é

muito limitado, porque se efectua com base num conjunto de informação muito restrito.

- A gravidade da falha está associada à própria natureza da situação que a provocou e ao tempo de espera pela sua resolução.

- O estado de resolução está relacionado com os procedimentos que o Operador deverá executar com o objectivo de resolver a falha.

Analisemos mais em detalhe este aspecto….

2.3 Procedimentos de gestão operacional de alarmes

2.3.1 Conceitos gerais

Quando um alarme chega ao TeMIP, o Operador da Rede pode alterar o seu estado, ou pode associar o alarme a um problema. Na primeira situação, o Operador responsabiliza-se pelo processamento, individual de cada alarme. Na segunda situação, o Operador está, de certa forma, a requisitar os serviços de pessoal de manutenção, que está encarregado de resolver os problemas do sistema em rede. Se pela descrição que for efectuada, diferentes alarmes conseguirem ser associados ao mesmo problema, o seu diagnóstico e resolução serão conduzidos de forma mais eficiente do que, se cada alarme for processado individualmente. Porém, esta possibilidade só existe quando o TeMIP integra o sistema de gestão de problemas.

Embore nos centremos na primeira situação (por ser aquela que vão poder reproduzir no laboratório), do nosso ponto de vista é importante saber que a segunda existe.

2.3.2 Ciclo de vida dos alarmes

Cada alarme que chega ao TeMIP, proveniente de um dado objecto de um sistema em rede, passa pelos mecanismos de filtragem e escalonamento de (pelo menos) um Operation Context e se for armazenado no seu Repositório de Informação, transforma-se num objecto. Como qualquer outro objecto, o alarme pode ser gerido até que o problema que o causou seja resolvido e o alarme se transforme num item histórico. Isto significa que o alarme possui um ciclo-de-vida e que, dentro deste ciclo, o seu estado varia de acordo com as acções realizadas pelo Operador da Rede.

Assim sendo, quando o alarme é recebido está no estado não processado (**Outstanding**). O Operador de Rede pode alterar o estado do alarme, indicando que tomou conhecimento da sua ocorrência (**Acknowledged**), ou indicando que o seu processamento está terminado (**Terminated**).

Só depois desta útlima situação se ter atingido é que o alarme pode ser arquivado (**Archived**)., isto é, pode passar ao histórico. O histórico não é mais do que uma base de dados relacional. Alternativamente, a partir do momento em que o processamento do alarme está completo, o alarme pode ser deitado fora (**purged**).

Esta situação está representada na figura 8.

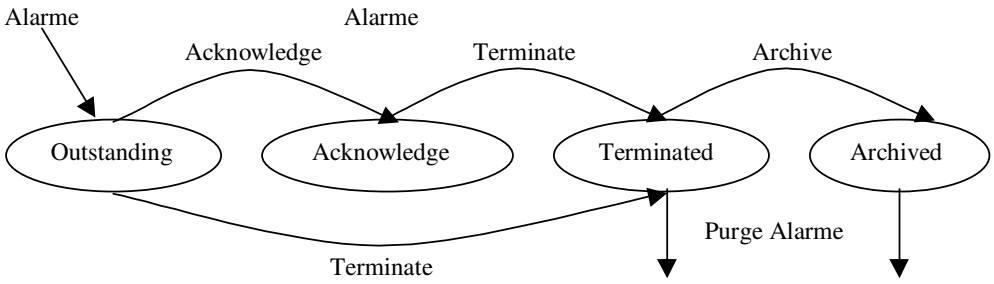


FIGURA 8 CICLO DE VIDA DOS ALARMES EM REPRESENTAÇÃO DE DIAGRAMA DE ESTADOS

2.4 Arquitectura de gestão de alarmes

Vamos agora analisar a gestão de alarmes na perspectiva da arquitectura de processos do TeMIP. Esta análise será realizada com recurso à figura 9.

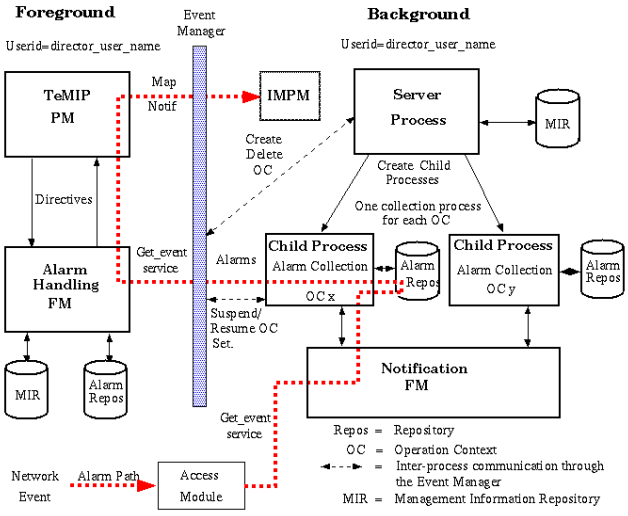


FIGURA 9 ARQUITECTURA DE PROCESSOS DA GESTÃO DE ALARMES DO TEMIP

A arquitectura de processos TeMIP baseia-se na existência de dois tipos de processos:

- Processos de foreground, que efectuam a interface com o utilizador
- Processos de background que são responsáveis pela gestão de alarmes (ou eventos).

Os processos de background são constituídos por um servidor, que é responsável por gerir os Operation Context, criando e removendo instâncias, e por um conjunto de processos filhos, cada um dos quais associado à recolha e armazenamento de alarmes de um dado Operation Context.

Cada um destes processo possui as suas estruturas de dados internas, que constituem os seus Repositórios de Informação de Gestão (MIRs) privados. Adicionalmente, cada processo filho possui um outro repositório onde armazena os alarmes recebidos que ainda não foram arquivados nem apagados.

O processamento dos alarmes é executado, em conformidade com as normas OSI, através da cooperação de diferentes módulos funcionais:

- **Alarm_Handling_FM**: que é responsável por apresentar a informação de qualquer alarme, de acordo com a estrutura definida para um relatório de alarm-OSI.

- **Alarm_FM**: que é responsável por gerar alarmes, internamente na Plataforma, quando determinadas condições, definidas através de regras, são satisfeitas.

- **Notification_Services_FM**: que é responsável por reportar os Alarmes ao módulo Alarm_Handling_FM, sob a forma de um alarme OSI. Este módulo consulta o Domain_FM para verificar se a entidade que gerou o Alarm realmente pertence ao domínio.

Antes de continuarem, sugiro uma pausa. Pensem entretanto, na quantidade de alarmes que uma só falha pode motivar. Como é que se pode reduzir tanta informação, de alguma forma redundante ?

2.5 Funcionalidades associadas à Gestão de Alarmes

2.5.1 Conceitos gerais

Quando surge uma falha num sistema em rede, pode ser gerada uma grande quantidade de alarmes, que se repetem periodicamente, até que seja enviada uma indicação em contrário. Se por um lado este comportamento tem a vantagem de que a descrição da falha está bem documentada, por outro lado, uma diversidade tão grande de informação dificulta o diagnóstico da situação, podendo inclusivamente conduzir a que outras falhas sejam ocultadas. Para além destes problemas de natureza operacional, um débito muito elevado de alarmes vai conduzir a uma sobrecarga desnecessária de informação nos Repositórios de Alarmes dos vários Operation Context. A aplicação de mecanismos de filtragem de alarmes, que podem actuar a diversos níveis - visualização, armazenamento ou recolha – vai permitir resolver estes problemas de excesso de informação.

Embora a filtragem de alarmes seja muito importante, sobretudo em Arquitecturas de Gestão baseadas em eventos, a sua utilização não resolve a diversidade de problemas que podem ocorrer. Assim, em termos de armazenamento de informação nos Repositórios de Alarmes, também é preciso assegurar que este armazenamento não se realiza nos períodos em que as equipas de gestão não se encontram em

funcionamento. Este procedimento é assegurado através de um mecanismo de Escalonamento que define os períodos de actividade de um dado Operation Context.

Em termos de resolução de problemas é fundamental assegurar que os alarmes são detectados o mais rapidamente possível. Numa Plataforma de Gestão não existe nenhum mecanismo que permita resolver esta situação, uma vez que a detecção implica uma acção explicita de um Operador da Rede. Neste contexto, a Plataforma pode apenas detectar estas situações e avisar, de alguma forma, o Operador de Rede da sua ocorrência. No TeMIP este procedimento é efectuado através do Alarm Escalation, que permite aumentar o nível de gravidade dos alarmes.

Quando um problema é resolvido é enviada um alarme para o TeMIP que se identifica pelo facto da sua severidade ser Clear. Este alarme tem um processamento especial, uma vez que deverá ser correlacionado com os alarmes anteriormente recebidos pelo sistema, referentes ao mesmo problema.

2.5.2 Filtragem de Alarmes

Do ponto de vista conceptual, a filtragem de alarmes permite seleccionar do conjunto total de alarmes que são enviados para o sistema de gestão, um sub-conjunto de alarmes mais restrito, que satisfaz as regras definidas nos filtros.

A primeira questão que se coloca é identificar em que pontos do sistema de gestão se devem aplicar os filtros, para que realmente se consiga reduzir a quantidade de informação a processar. Considerando que cada um dos módulos que compõe a plataforma executa um tipo de processamento diferente é possível concluir que:

- Ao nível superficial deve ser possível limitar a quantidade de alarmes que o Operador de Rede pode visualizar, aplicando **Filtros de Visualização**.

- Ao nível intermédio deve ser possível limitar a quantidade de alarmes que é recebida e armazenada num Operation Context, aplicando **Filtros de Alto Nível**.

- Ao nível inferior deve ser possível limitar a quantidade de alarmes que é recebida na plataforma de gestão, aplicando filtros de **Baixo Nível**.

2.5.2.1 Filtros de visualização

Os filtros de visualização são aplicados na Alarm-Handling-Window e não têm qualquer influência na informação armazenada. Neste contexto, se houver filtros definidos nesta aplicação, podem ser visualizados os alarmes filtrados na janela View-Alarm-Window, que acede directamente ao Repositório de Alarmes.

Os filtros podem especifica diversos tipos de parâmetros incluídos na descrição do alarme. Por exemplo pode-se filtrar de acordo com aspectos organizativos da rede, isto é, por Domínio, Sub-Domínio, ou Entidade, pode-se filtrar de acordo com as características dos alarmes, tais como, tipo, nível de severidade ou causa provável, ou pode-se filtrar de acordo com a própria organização do TeMIP,isto é, por Operation Context. Nalgumas situações os campos são definidos directamente, enquanto que noutras situações implicam a definição de uma expressão regular.

2.5.2.2 Filtros de alto nível

A filtragem de alto nível é aplicada nos Operation Context, seleccionando quais os alarmes que serão armazenados no respectivos Repositórios de Alarmes, através da aplicação de um conjunto de regras, definidas de acordo com princípios definidos na Arquitectura de Gestão OSI.

Por cada Operation Context em que se defina este tipo de filtragem, estas regras são definidas num filtro denominado de **Discriminator Construct**. Este filtro é constituído por dois tipos de filtros – Sub-Filtros Bloqueantes e Sub-Filtros Passantes – que por sua vez são constituídos por um conjunto de expresssões regulares.

Os Discriminator Constructs e os Sub-Filtros podem ser definidos e aplicados, através de janelas na aplicação Iconic_map, ou através de comandos Fcl. Os Discriminator Constructs formam uma biblioteca, de forma a que possam ser re-utilizados por diversos Operation Contexts.

O princípio de filtragem de alto nível está representado na figura 10: o evento ou alarme começa por ser processado pelos sub-filtros bloqueantes e se alguma das condições de filtragem se verificar é deitado fora; seguidamente, é processado pelos sub-filtros passantes e só se todas as condições de filtragem se verificarem é que é enviado para o Repositório de Alarmes do Operation Context.

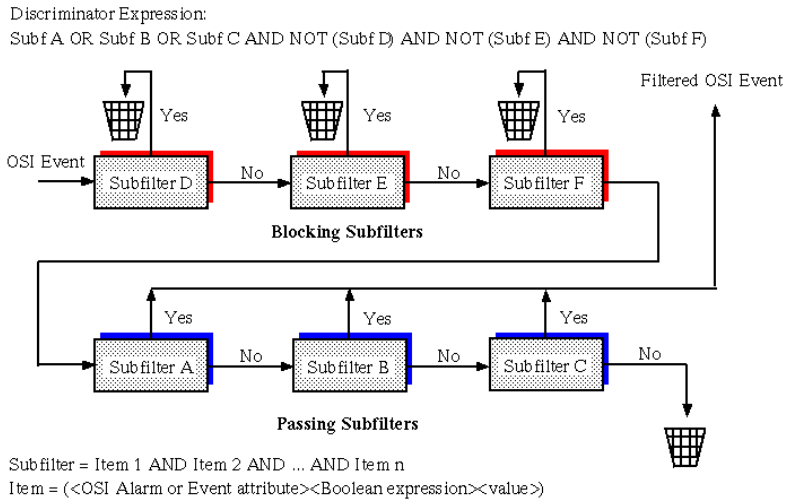


FIGURA 10: FILTRAGEM DE ALTO NÍVEL

Que tipos de sub-filtros se devem definir ? Não há regras pré-estabelecidas, isto é, os critérios dependem das características do sistema de gestão de faltas que se quer implementar. Apesar disso, consideremos apenas um caso exemplificativo das potencialidades deste tipo de filtragem.

EXEMPLO:

Os sub-filtros passantes podem definir o âmbito de interesse do Operador de Rede na actividade de alarme. Neste contexto, podem identificar Domínios, Classes de

equipamentos , etc.. Os Sub-filtros bloqueantes podem restringir a gama de alarmes que passa pelo Sub-Filtro Passante.

No caso anterior, os Sub-Filtros Passantes indicam que, num dado Domínio só se pretende receber alarmes provenientes de determinado tipo de equipamento. Um Sub-Filtro Bloqueante poderia indicar que não se pretende receber alarmes de pequena gravidade, pelo que, todos os alarmes com severidade inferior a um dado nível deveriam ser bloqueados.

2.5.2.3 Filtros de baixo nível

A motivação da Filtragem de Baixo Nível está associada ao facto de existirem alarmes/eventos que surgem em consequência de outros e que, normalmente, não contêm informação significativa para o Operador da Rede. Este tipo de filtragem é implementada nos AMs e está associada a alarmes e eventos em simultâneo. Os critérios de filtragem são totalmente diferentes dos critérios utilizados pelos filtros anteriores, estando geralmente associados ao comportamente temporal dos alarmes/eventos recebidos.

No TeMIP foram definidos quatro tipos de filtros de baixo nível - Tap Filter, Transient Filter, Threshold Filter e Heap Filter-, que funcionam em cascata, servindo para resolver diferentes situações:

- **Tap Filter** é o primeiro filtro da cascata e utiliza-se para evitar que eventos/alarmes não desejados cheguem ao TeMIP e provoquem possíveis situações de congestão.
  Este é um filtro do tipo On/Off, que bloqueia os eventos/alarmes que não satisfazem os critérios definidos no Discriminator Construct (ver figura 11).

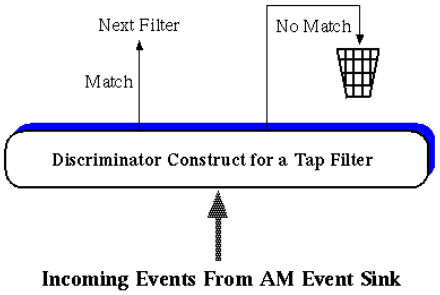


FIGURA 11: FILTRAGEM DE BAIXO NÍVEL –TAP FILTER

- **Transient Filter** é segundo filtro da cascata, sendo utilizado para evitar que alarmes/eventos gerados em situações de instabilidade sejam processados pelo TeMIP. Por exemplo, uma interface com uma má ligação pode gerar diversos alarmes, até que a ligação fique estável, com a utilização deste filtro estes alarmes não chegam sequer a ser processados.
  Este filtro colecciona eventos, que armazena numa fila durante um dado período de tempo. Se durante esse tempo for recebido um Clear, significa que o problema se resolveu e não é enviado nenhum alarme/evento para o Operador; caso

contrário, considera-se que se está em presença de um problema de longa duração, ou de maior gravidade, e é enviado o primeiro alarme/evento para o Operador da Rede, contendo informação extra, referente a outros alarmes similares que também tenham chegado durante o período de tempo (ver figura 12).

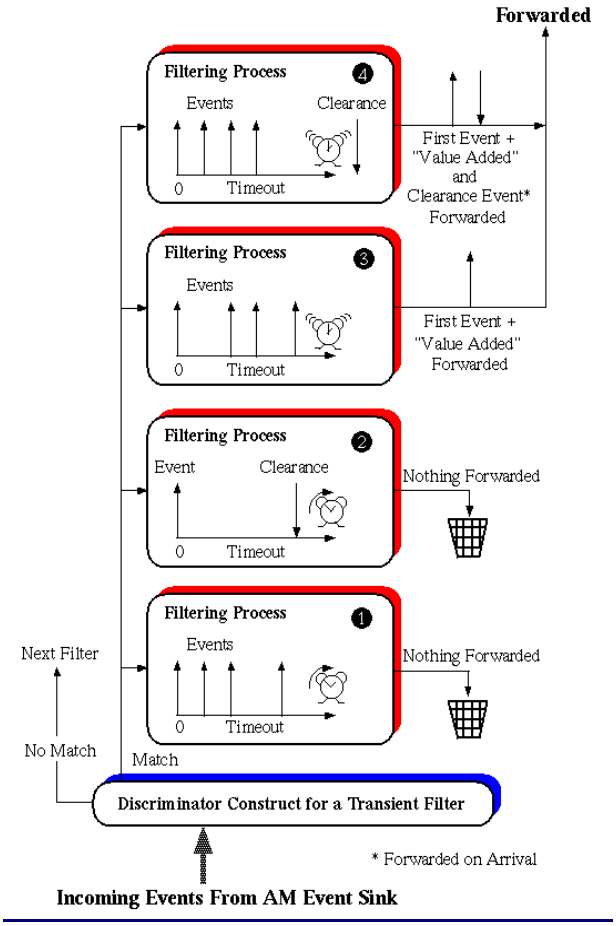


FIGURA 12: FILTRAGEM DE BAIXO NÍVEL –TRANSIENT FILTER

- **Threshold Filter** é o terceiro filtro da cascata, sendo utilizado para evitar que alarmes/eventos isolados, motivados por problemas que entretanto se auto-resolveram, sejam processados pelo TeMIP. Por exemplo, uma ligação X.25 que vai abaixo, mas que entretanto é restabelecida não deverá gerar alarmes que sejam processados.
  Este filtro colecciona eventos, que armazena numa fila durante um dado período de tempo. Se durante esse tempo for excedido um número pré-definido de alarmes/eventos, considera-se que o problema é significativo, porque atingiu um certo ritmo de geração de alarmes/eventos e é enviado um alarme/evento, com informação extra; caso contrário, considera-se que o problema não é significativo e não é enviado nada (ver figura 13).

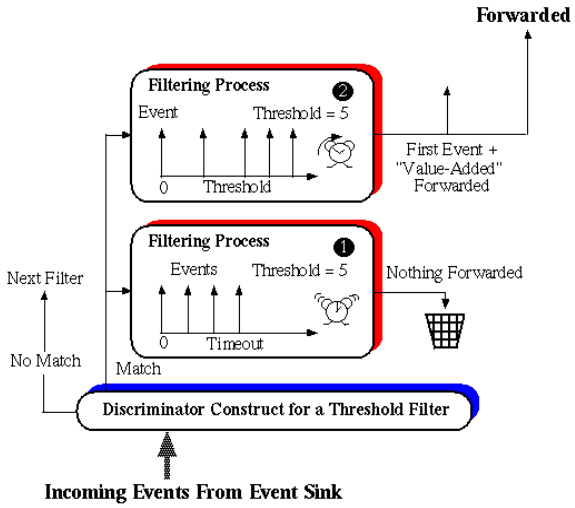


FIGURA 13: FILTRAGEM DE BAIXO NÍVEL –THRESHOLD FILTER

- **Heap Filter** é o último filtro da cascata, sendo utilizado para fornecer informação sobre os alarmes/eventos de uma forma estatística, deixando ao critério do Operador da Rede a decisão de avaliar sobre a gravidade do problema.
  Este filtro colecciona eventos, que armazena numa fila durante um dado período de tempo. Findo esse tempo é enviado um alarme/evento,contendo informação estatística referentes aos alarmes/eventos recebidos durante esse período (ver figura 14).

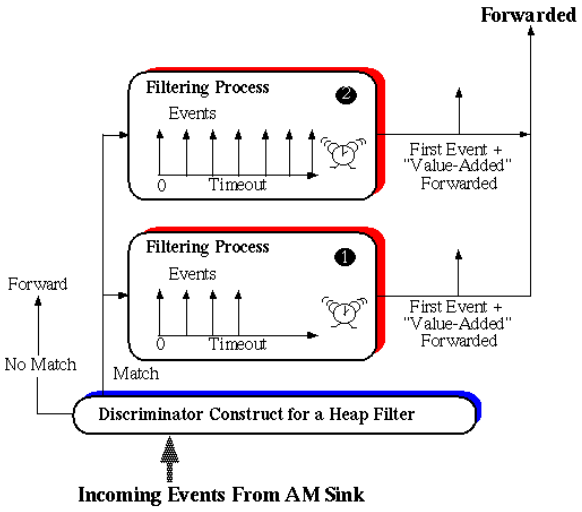


FIGURA 14: FILTRAGEM DE BAIXO NÍVEL – HEAP FILTER

2.5.3 Escalonamento de Alarmes

Do ponto de vista conceptual, a escalonamento de alarmes permite seleccionar os períodos de tempo em que recolha de alarmes está activa. Os Pacotes de Escalonamento são definidos de forma semelhante aos Discriminator Constructs.

Neste contexto, cada Operation Context pode ter associado um Pacote de Escalonamento, podendo ser definido, através de janelas na aplicação Iconic_map, ou através de comandos Fcl.

Um Pacote de Escalonamento define a data de ínicio e de fim da colecção de alarmes, os dias de actividade semanal e, dentro de cada dia, as horas de actividade. Em alternativa, pode-se definir apenas dias de actividade e dias de inactividade.

2.5.4 Escalation de Alarmes

Cada alarme que chega ao TeMIP tem um nível de gravidade, atribuído pelo sistema gerido, que varia entre **Indeterminate**, **Critical**, **Major**, **Minor**, **Warning** e **Clear**. Através deste procedimento o nível de gravidade do alarme é aumentado automaticamente, se este não passar ao estado acknowledge durante um período de tempo pré-estabelecido, sendo gerado um novo alarme. Nestas circunstâncias o alarme original chama-se de Escalated Alarm e o novo alarme gerado de Escalation Alarme.

2.5.5 Correlação de Alarmes

A correlação de alarmes ocorre após a recepção de um alarme com o nível de gravidade igual a Clear, que indica que um problema anteriormente detectado já foi resolvido.

Através da correlação os alarmes anteriormente recebidos, referentes ao mesmo problema, são identificados e o seu estado passa a Clear. Sempre que a correlação se verifique, é gerado um alarme fictício, **Pseudo-Alarm**, que representa todos os alarmes que estão correlacionados.

Alarmes correlacionados partilham as seguintes características do Clear Alarme: Objecto gerido, Tipo, Causa Provável e Problemas Específicos.

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[9]: Leitura geral

Capítulo V — Ferramentas de desenvolvimento

Palavras chave: GAT, Especificação Funcional, Interface ASCII.

1 Introdução

Na aula passada foi-vos apresentada a Gestão de Alarmes do TeMIP. As suas potencialidades como ferramentas de execução não se restrinjem ao que foi descrito, englobando também uma Gestão de Eventos, igualmente poderosa, e facilidades de monitorização e controlo de sistemas geridos. A nível funcional e conceptual a Gestão de Eventos é muito semelhante à Gestão de Alarmes, apresentando diferenças pontuais que serão ilustradas na sessões laboratoriais. Os aspectos de monitorização e controlo assumem particular relevância quando os sistemas geridos possuem uma Arquitectura de Gestão orientada ao polling. Assim sendo, será interessante descrevê-los, posteriormente, quando se abordar o TeMIP como ferramenta de gestão de redes TCP/IP.

Hoje ser-vo-á dada a perspectiva do TeMIP como ferramenta de desenvolvimento, dando especial enfâse a uma ferramenta gráfica, que permite desenvolver AMs para dispositivos que suportem Ascii ou TL1. A escolha desta ferramenta para ilustrar as potencialidades do TeMIP neste domínio,deriva do facto de servir para ilustrar todos os conceitos, de uma forma suficiente simples para se poder realizar ao longo de uma aula.

2 O TeMIP como ferramenta de desenvolvimento

O TeMIP oferece diferentes possibilidades de integração, com diferentes níveis de desenvolvimento. É possível integrar novas funcionalidades sem efectuar qualquer tipo de desenvolvimento, efectuando adaptações (customização) ou desenvolvendo integralmente novas aplicações.

A integração de facilidades sem desenvolvimento efectua-se de três formas diferentes:

- **Application Launch**: executar uma aplicação através de um Módulo de Apresentação, definindo um ficheiro de definição de interface de aplicação.
- **Collection AM**: coleccionar eventos de objectos não geridos pelo TeMIP, definindo mensagens através de uma shell de comandos.
- **Script AM**: activar scripts e integrar o resultado no TeMIP.

A costumização efectua-se para os Módulos de utilizando os toolkits associados a cada arquitectura de gestão, SNMP, OSI e CORBA. Relativamente a estas duas arquitecturas, a customização também se pode efectuar ao nível de Módulo de Apresentação. Nesta situação a plataforma funciona como agente. A customização de Módulos de Acesso ASCII efectua-se através de uma interface gráfica, o **Graphical ASCII ToolKit**.

O desenvolvimento de aplicações do Módulo de Apresentação também se pode efectuar através de **TeMIP Access Library**, no caso das aplicações correrem sobre o sistema operativo UNIX ou Windows.

Por último, é possível utilizar a linguagem de programação **Visual TeMIP**, que é baseada em C++, para desenvolver FMs ou AMs.

3 O Graphical ASCII Toolkit - GAT

3.1 Motivação do GAT

O TeMIP interage com os elementos de rede através de AMs. A capacidade de suportar diferentes protocolos realiza-se através da utilização de diferentes Módulos de Acesso. À medida que novos elementos de rede são adicionados novos AMs tem de ser desenvolvidos. O objectivo do GAT é tornar o processo de desenvolvimento mais rápido e, simultaneamente, permitir a realização de alterações a AMs existentes, de forma a possibilitar a introdução de novas funcionalidades, adaptadas aos novos requisitos do mercado.

3.2 Funcionalidades dos AMs obtidos através do GAT

Os AMs desenvolvidos através do GAT permitem gerir a ligação com o elemento de rede, através da utilização de protocolos normalizados, tais como o TCP/IP, X.25 e RS232. Estes AMs podem monitorizar o estado das ligações, tentar restabelecer de forma automática ligações que falham e enviar alarmes para a Plataforma quando a comunicação falha.

Os AMs desenvolvidos através do GAT funcionam com um Agente para o elemento de rede associado. Neste contexto, convertem as directivas TeMIP em sequências de comandos ASCII, coleccionam mensagens ASCII, emitidas assíncronamente pelos elementos de rede, gerando eventos para a Plataforma, fazem polling aos equipamentos que não são capazes de gerar estas mensagens assíncronas, armazenam as mensagens em ficheiros de log etc..

O nível de funcionalidade obtida depende do grau de costumização do AM.

3.3 Descrição do AM

O GAT é constituído por um conjunto de componentes, cuja relação se representa na figura 15. De uma forma genérica, o GAT fornece dois canais de comunicação distintos entre o TeMIP e o elemento de rede. Existem então quatro componentes fundamentais no GAT:

- **Asynchrounous Message Handel** (AMH): um canal unidireccional e assíncrono, que se utiliza para o TeMIP receber mensagens não solicitadas, o.

- **Integrated Command Generator** (ICG): um canal de comunicação bidireccional e síncrono, que se utiliza para o TeMIP enviar comandos para o elemento de rede e receber as respostas por este geradas.

- **Management Module** (MM): uma interface que contém a representação do modelo de informação do elemento de rede a gerir e que possibilita o acesso da informação de gestão associada a esse elemento às aplicações do TeMIP.

- **Network Element Interface** (NE I/F): uma interface física com o elemento de rede, que se realiza através de servidores de comunicação, que não se podem alterar. Existem um servidor para cada um dos protocolos de comunicação suportados: RS232, X.25 ou TCP/IP.

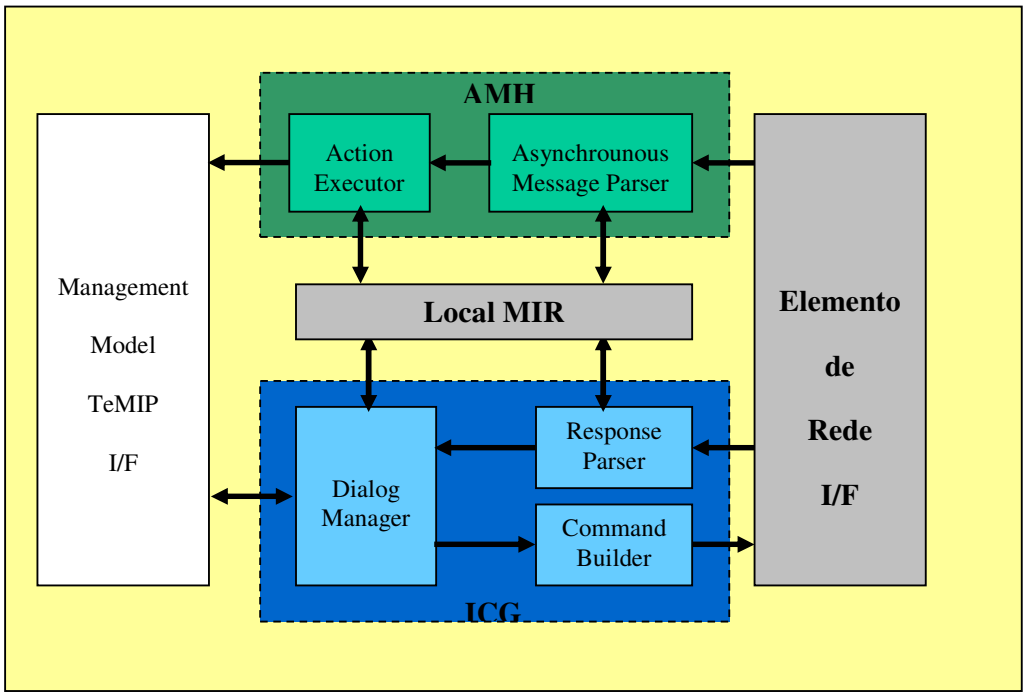


FIGURA 15: COMPONENTES DO MÓDULO DE ACESSO GAT

3.3.1 AMH - Asynchrounous Message Handler

O AMH é responsável por processar acontecimentos assíncronos, quer sejam alarmes quer sejam mensagens de estado não solicitadas. Esta parte do AM é constituída por um Parser e um Action Executor.

O Parser analisa a cadeia de caracteres que constitui a mensagem, extrai a parte da mensagem que foi previamente considerada como significativa, armazena-a como um **Item Set** e determina a acção a realizar. A extracção da mensagem efectua-se com base na aplicação de um conjunto de regras.

As acções a realizar pelo Action Executor fazem parte de uma lista de acções pré-definida. Se esta lista não for suficiente para processar um dado evento, é possível adicionar-lhe novas acções. A lista de acções pré-definidas contém acções que permitem: gerar um evento OSI, actualizar o valor de um atributo, actualizar o estado operacional e gerar um evento de alteração de estado, manter uma lista de alarmes activos no elemento de rede, fazer o reset a um temporizador de inactividade e gerar um evento de auto-descoberta, quando um novo elemento de rede é detectado.

3.3.2 Integrated Command Generator

O ICG é responsável por controlar e monitorizar o elemento de rede, através do envio de comandos e recepção de respostas. Este componente é constituído por um Dialog Manager, um Command Builder e um Response Parser.

O Dialog Manager é responsável por receber e processar as directivas TeMIP, enviando-as para o Command Builder e por receber do Response Parser as respostas correspondentes geradas pelos elementos de rede. Uma sequência de Directivas TeMIP e respostas constitui um Diálogo.

No Command Builder, cada directiva TeMIP é convertida numa cadeia de caracteres ASCII - um comando reconhecido pelo elemento de rede – que é enviada através da interface com esse elemento.

As respostas recebidas destes elementos são processadas no Response Parser, antes de serem devolvidas ao Dialog Manager. O processamento a realizar é semelhante ao que é efectuado pelo Asynchrounous Parser, podendo-se aglutinar os dois num só.

3.4 Funcionalidades disponíveis no GAT

O GAT é uma aplicação gráfica, para customizar AMs com a estrutura anteriormente definida.

O ponto de partida é a **Customization Main Window**, a partir da qual se podem efectuar diferentes funções, conforme a frame que se esteja a utilizar:

- A Component Frame permite seleccionar o bloco a costumizar, por meio de uma descrição gráfica semelhante à que anteriormente se definiu.

- A Detail Frame descreve as características da costumização actual.

- A Output Frame contém as mensagens geradas pelas ferramentas associadas ao GAT, quer sejam mensagens de erro, avisos, indicações de progresso, informação de caracter geral ou informação não especifícada.

A selecção de um bloco vai originar o aparecimento de uma nova janela, a **Component Editor Window**, cuja estrutura depende do bloco seleccionado. Esta janela é sempre composta por diferentes colunas, que permitem definir, gradualmente, as características do bloco.

3.4.1 Procedimentos de costumização

3.4.1.1 Pré-requisitos

No desenvolvimento de um módulo de acesso é fundamental compreender três aspectos essenciais, referentes ao elemento de rede que se pretende gerir: as suas funcionalidades, as características da sua interface de gestão e as funcionalidades que se pretendem suportar na Plataforma.

Assim sendo, o primeiro passo no desenvolvimento de um AM através do GAT consiste na construção de uma **Especificação Funcional**, que contemple estes aspectos. Com base no que já se sabe sobre o GAT, é possível ser um pouco mais minucioso na definição da Especificação Funcional.

- A especificação deve então iniciar-se pela descrição semântica das capacidades do elemento de rede. Esta tarefa pode-se realizar, por exemplo, através dum manual de utilização.

- Seguidamente é fundamental identificar as possibilidade que o elemento de rede oferece para a monitorização da ligação à plataforma. Por exemplo, se o elemento possui um mecanismo de controlo do tipo Heartbeat, o GAT pode detectar e monitorizar os sinais por ele gerados. Caso contrário, o GAT deverá gerar mensagens de Keep-alive para manter a ligação com o elemento de rede.

- Estando os aspectos de ligação assegurados, é necessário identificar os alarmes gerados pelo elemento de rede, os comandos recebidos e as respostas enviadas. É fundamental incluir na especificação exemplos reais, que possam ser utilizados durante a fase de desenvolvimento e teste. Este aspecto permite evitar problemas originados por inconsistência entre a situação real e a informação existente na documentação do elemento de rede.

- Por último, é necessário definir os requisitos de gestão impostos pela plataforma, isto é identificar quais as acções de gestão que se espera realizar através do TeMIP.

Com base nesta Especificação Funcional de alto nível deve ser construída uma outra Especificação, que contemple os aspectos relativos à arquitectura do GAT. Esta especificação realiza-se em 6 etapas diferentes:

1. **Modelo de entidades**: definição do conjunto deClasses de Objectos Geridos que representa o elemento de rede a gerir. Para cada Classe considerada especificam-se os Atributos, Operações e Notificações emitidas.

2. **Identificação do Elemento de Rede**: identificação do elemento de rede nas mensagens recebidas na plataforma e mapeamento na Entidade TeMIP correspondente.

3. **Mapeamento entre mensagens e eventos TeMIP**: definição do mapeamento entre as mensagens recebidas e os eventos TeMIP e dos procedimentos a realizar para com as mensagens que não são mapeadas.

4. **Comandos**: definição dos comandos suportados pelo AM

5. **Características diversas**: definição de características diversas do AM, tais como o seu estado operacional, a manutenção da lista de alarmes e dos valores dos atributos na MIR.

6. **Ligação ao elemento de rede**: definição da ligação entre o AM e o elemento de rede.

4 Resumo final

No âmbito da operação da rede, mais precisamente da gestão de faltas, o TeMIP oferece um conjunto de funcionalidades fortemente baseados nos princípios da Aqruitectura de Gestão OSI.

A nível de gestão de alarmes possui funcionalidades de filtragem, escalonamento e correlação. A filtragem pode ser executada a três níveis diferentes: na interface com o utilizador, antes do armazenamento nos Repositórios de Alarmes ou ainda na chegada da informação de alarme ao AM. Cada alarme armazenado é convertido num objecto alarme, que possui uma descriçao normalizada.

O TeMIP oferece diferentes possibilidades de integração de novos módulos, de forma a obter os resultados pretendidos no melhor tempo possível. Uma destas ferramentas, o GAT, serve para desenvolver módulos de acesso para equipamentos com interface ASCII, o que corresponde a um universo relativamente grande de situações reais.

A arquitectura do GAT foi apresentada assim como o tipo de especificação funcional que deve ser produzido antes de se efectuar o desenvolvimento. A informação fornecida corresponde a uma abordagem muito genérica, que não dispensa de forma alguma a consulta dos manuais, caso se pretenda desenvolver um AM.

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[11]: leitura completa.

Tema: Sistemas de Gestão de Problemas

Palavras chave: GAT, Especificação Funcional, Interface ASCII.

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

2ª Aula

Tema: Sistemas de Gestão de Problemas

Palavras chave:

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

3ª Aula

Tema: Gestão de Redes e Gestão de Sistemas

Palavras chave:

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

4ª Aula

Tema: Operaçãp estruturada de sistemas em rede

Palavras chave:

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

5ª Aula

Tema: Níveis de Serviço e Gesão de Níveis de Serviço

Palavras chave: GAT, Especificação Funcional, Interface ASCII.

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

6ª Aula

Tema: Processos de negócio e ferramentas de gestão associadas

Palavras chave:

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

7ª Aula

Tema: Gestão de novos serviços

Palavras chave: GAT, Especificação Funcional, Interface ASCII.

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

8ª Aula

Tema: Avaliação de tendências de gestão. Conclusões.

Palavras chave: GAT, Especificação Funcional, Interface ASCII.

Introdução

Resumo final

Sugestão de leitura

Para aprofundar os conhecimentos sobre os assuntos da aula de hoje podem ler:

[

Bibliografia

1. "Integrated Management of Networked Systems – Concepts, Architectures and Their Operational Application", Heinz-Gerd Hegering, Sebastian Abeck e Bernhard Neumair, Morgan Faufmann Publishers, 1998

2. " Network and Distributed System Management", Morris Sloman, Addison-Wesley, 1996.

3. "SNMP SNMPv2 and RMON – A Pratical Network Management", 2ª edição, William Stallings.

4. "TMN – Telecommunication Management Network", Divakara Udupa, McGraw-Hill Series on Computer Communications

5. Recomendações do ITU-T, série X.700

   5.1. X.701 – System Management Overview

   5.2. X.710 – Common Management Information Service

   5.3. X.711 – Common Management Information Protocol

   5.4. X.712 – Common Management Information Protocol – Conformance State Proforma

6. RFCs do IETF

7. "The Lean Communications Providers – Serviving the Skateout through Service Management Excellence", E. Adams e K. Willetts, NMF, Mc-Graw Hill

8. "Multiplatform Network Management", D. Edgar Taylor, McGraw-Hill Series on Computer Communications.

9. TeMIP Introduction, AA-Q9J1E-TE, Compaq

10. Alarm Handling ---?

11. TeMIP Graphical ASCII Toolkit – Customization Manual, AA-QZASC-TE, Compaq