Implementação de TMN através de CORBA

Autor: Francisco Adell Péricas pericas@uol.com.br
Colaboradores: Daniel Puka puka@dainf.cefetpr.br
              Rafaela Vergès Mayrhofer rafaela@dainf.cefetpr.br
Data: Setembro de 1999
1. Sumário

1. SUMÁRIO ................................................................................................................. 2

2. TABELA DE FIGURAS .............................................................................................. 4

3. GLOSSÁRIO ............................................................................................................... 5

4. RESUMO .................................................................................................................. 7

5. INTRODUÇÃO ......................................................................................................... 8
   5.1 AMBIENTES TMN E CORBA ................................................................................. 8
   5.2 INTEGRAÇÃO DA INFRA-ESTRUTURA CORBA COM O AMBIENTE TMN .......... 9

6. TMN ......................................................................................................................... 10
   6.1 ESTRUTURA DO TMN ......................................................................................... 10
       6.1.1 Padrão TMN ............................................................................................ 10
       6.1.2 TMN, OSI e Gerência ............................................................................. 11
   6.2 MODELO FUNCIONAL DO TMN ....................................................................... 11
       6.2.1 Blocos TMN ............................................................................................ 12
       6.2.2 Distribuição das Funções de Gerência TMN ............................................. 12
   6.3 INTERFACES PADRÃO DO TMN ....................................................................... 14
   6.4 MODELO LÓGICO DO TMN .............................................................................. 16
   6.5 ÁREAS FUNCIONAIS DE GERENCIAMENTO ............................................... 17
   6.6 SERVIÇOS DE GERÊNCIA TMN ....................................................................... 18
   6.7 SOLUÇÕES TMN ............................................................................................... 18
       6.7.1 Integração de Gerência de Equipamentos .................................................... 18
       6.7.2 Interface de Programação Padrão Orientada a Objetos ............................... 19
       6.7.3 Ferramentas de Automação de Implementação de Aplicações TMN ............. 20

7. CORBA ...................................................................................................................... 21
   7.1 OMG .................................................................................................................... 21
   7.2 ESPECIFICAÇÕES RELACIONADAS ................................................................. 21
   7.3 OMA .................................................................................................................... 21
       7.3.1 Modelo de Referência da OMA ................................................................. 22
   7.4 ESTRUTURA DE UM ORB .................................................................................. 23
       7.4.1 ORB ......................................................................................................... 26
       7.4.2 Clientes ..................................................................................................... 26
       7.4.3 Implementações de Objetos ...................................................................... 26
       7.4.4 Referências a Objetos .............................................................................. 27
       7.4.5 IDL da OMG ............................................................................................. 27
       7.4.6 Mapeamento de IDL para Linguagens de Programação ........................... 27
       7.4.7 Stubs de Clientes ....................................................................................... 27
       7.4.8 Interface de Invocação Dinâmica .............................................................. 28
       7.4.9 Implementação de Skeletons ..................................................................... 28
       7.4.10 Interface Dinâmica de Skeletons ............................................................... 28
       7.4.11 Adaptadores de Objetos .......................................................................... 29
       7.4.12 Interface de um ORB ............................................................................... 29
       7.4.13 Repositório de Interfaces .......................................................................... 29
       7.4.14 Repositório de Implementações ............................................................... 29
       7.4.15 Exemplos de ORBs .................................................................................. 30
   7.5 ESTRUTURA DE UM CLIENTE ............................................................................ 31
   7.6 ESTRUTURA DE UMA IMPLEMENTAÇÃO DE OBJETOS ................................. 32
   7.7 ESTRUTURA DE UM ADAPTADOR DE OBJETOS ............................................. 33

2 de 51
8. IMPLEMENTAÇÃO DE TMN ATRAVÉS DE CORBA ................................................................. 41

8.1 CONSIDERAÇÕES TECNOLÓGICAS RELATIVAS À INTEGRAÇÃO TMN E CORBA .................. 41
8.1.1 Comparação entre as Características do CORBA, do TMN e do IETF SNMP ...................... 41
8.1.2 Questões Relativas à Integração de CORBA e TMN ............................................................. 42

8.2 INTEGRAÇÃO DA OMG ........................................................................................................ 44
8.2.1 CORBA para os Sistemas de Gerenciamento ....................................................................... 44
8.2.2 CORBA para os Sistemas Gerenciados ............................................................................... 45

8.3 INTERAÇÃO E ESPECIFICAÇÃO DO JIDM ............................................................................ 45
8.3.1 Especificação da Tradução .................................................................................................. 46
8.3.2 Especificação da Interação ................................................................................................ 47

8.4 ODMA ................................................................................................................................... 47

9. CONCLUSÕES ........................................................................................................................ 50

10. REFERÊNCIAS BIBLIOGRÁFICAS ......................................................................................... 51
2. Tabela de Figuras

FIGURA 1: TMN E A REDE DE TELECOMUNICAÇÕES

FIGURA 2: BLOCOS FUNCIONAIS DO TMN

FIGURA 3: INTERAÇÃO ENTRE GERENTE, AGENTE E OBJETOS GERENCIÁVEIS

FIGURA 4: BLOCOS FUNCIONAIS E COMPONENTES FUNCIONAIS

FIGURA 5: INTERFACES PADRÃO ENTRE COMPONENTES FUNCIONAIS TMN

FIGURA 6: INTERFACES Q

FIGURA 7: DECOMPOSIÇÃO DA FUNCIONALIDADE DE GERÊNCIA

FIGURA 8: EXEMPLO DE COMPONENTES TMN DISTRIBUÍDOS NAS CAMADAS LÓGICAS

FIGURA 9: API TMN/C++ DA TMF

FIGURA 10: ESTRUTURA TÍPICA DO CORBA

FIGURA 11: ESTRUTURA DAS INTERFACES DE UM ORB

FIGURA 12: CLiente UTILIZANDO UM STUB OU UMA INTERFACE DE INVOCAÇÃO DINÂMICA

FIGURA 13: IMPLEMENTAÇÃO DE OBJETO RECEBENDO UM PEDIDO

FIGURA 14: REPOSITÓRIOS DE INTERFACE E DE IMPLEMENTAÇÃO

FIGURA 15: ESTRUTURA TÍPICA DE UM CLIENTE

FIGURA 16: ESTRUTURA TÍPICA DE UM IMPLEMENTAÇÃO DE OBJETOS

FIGURA 17: ESTRUTURA TÍPICA DE UM ADAPTADOR DE OBJETOS

FIGURA 18: FORMAS DE INTEGRAÇÃO COM OUTROS SISTEMAS DE OBJETOS

FIGURA 19: MODELO DE REFERÊNCIA DE UM SISTEMA DA OMG

FIGURA 20: ENTRADAS E SAÍDAS DE UM TRADUTOR GDMO/ASN.1 PARA IDL

FIGURA 21: DOCUMENTOS RELATIVOS À ODMA

FIGURA 22: SISTEMA ABERTO DE GERENCIAMENTO BASEADO EM CORBA
3. Glossário

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>ASN.1</td>
<td>Abstract Syntax Notation One</td>
</tr>
<tr>
<td>ATM</td>
<td>Asynchronous Transfer Mode</td>
</tr>
<tr>
<td>BLA</td>
<td>Business Level Agreements</td>
</tr>
<tr>
<td>BML</td>
<td>Business Management Layer</td>
</tr>
<tr>
<td>BOA</td>
<td>Basic Object Adapter</td>
</tr>
<tr>
<td>CMIP</td>
<td>Common Management Information Protocol</td>
</tr>
<tr>
<td>CMIS</td>
<td>Common Management Information Service</td>
</tr>
<tr>
<td>CMISE</td>
<td>Common Management Information Service Entry</td>
</tr>
<tr>
<td>COM</td>
<td>Component Object Model</td>
</tr>
<tr>
<td>CORBA</td>
<td>Common Object Request Broker Architecture</td>
</tr>
<tr>
<td>DCF</td>
<td>Data Communication Function</td>
</tr>
<tr>
<td>DCM</td>
<td>Data Communication Network</td>
</tr>
<tr>
<td>DPE</td>
<td>Distributed Processing Environment</td>
</tr>
<tr>
<td>EFD</td>
<td>Event Forwarding Discriminator</td>
</tr>
<tr>
<td>EML</td>
<td>Element Management Layer</td>
</tr>
<tr>
<td>ETSI</td>
<td>European Telecommunication Standards Institute</td>
</tr>
<tr>
<td>FDDI</td>
<td>Fiber Distributed Data Interface</td>
</tr>
<tr>
<td>GDTS</td>
<td>Guideline for Definition of Managed Objects</td>
</tr>
<tr>
<td>IDL</td>
<td>Interface Definition Language</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IIOP</td>
<td>Internet Inter-ORB Protocol</td>
</tr>
<tr>
<td>IPC</td>
<td>Inter-Process Communication</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>ITU</td>
<td>International Telecommunications Union</td>
</tr>
<tr>
<td>JIDM</td>
<td>Joint Inter-Domain Management</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LOA</td>
<td>Library Object Adapter</td>
</tr>
<tr>
<td>MAF</td>
<td>Management Application Function</td>
</tr>
<tr>
<td>MCF</td>
<td>Message Communication Function</td>
</tr>
<tr>
<td>MD</td>
<td>Mediation Device</td>
</tr>
<tr>
<td>MF</td>
<td>Mediation Function</td>
</tr>
<tr>
<td>MIB</td>
<td>Management Information Base</td>
</tr>
<tr>
<td>NE</td>
<td>Network Element</td>
</tr>
<tr>
<td>NEF</td>
<td>Network Element Function</td>
</tr>
<tr>
<td>NEL</td>
<td>Network Element Layer</td>
</tr>
<tr>
<td>TMF</td>
<td>Telecommunications Management Forum</td>
</tr>
<tr>
<td>NML</td>
<td>Network Management Layer</td>
</tr>
<tr>
<td>ODM</td>
<td>Open Distributed Management</td>
</tr>
<tr>
<td>ODMA</td>
<td>Open Distributed Management Architecture</td>
</tr>
<tr>
<td>ODP</td>
<td>Open Distributed Processing</td>
</tr>
<tr>
<td>OLE</td>
<td>Object Linking and Embedding</td>
</tr>
<tr>
<td>OMA</td>
<td>Object Management Architecture</td>
</tr>
<tr>
<td>OMG</td>
<td>Object management Group</td>
</tr>
<tr>
<td>OODA</td>
<td>Object Oriented Database Adapter</td>
</tr>
<tr>
<td>OODB</td>
<td>Object Oriented Database</td>
</tr>
<tr>
<td>ORB</td>
<td>Object Request Broker</td>
</tr>
<tr>
<td>OS</td>
<td>Operations System</td>
</tr>
<tr>
<td>OSF</td>
<td>Operations System Function</td>
</tr>
<tr>
<td>OSI</td>
<td>Open Systems Interconnection</td>
</tr>
<tr>
<td>OSS</td>
<td>Operations Support Systems</td>
</tr>
<tr>
<td>OTMA</td>
<td>Open Telecommunications Management Architecture</td>
</tr>
<tr>
<td>POA</td>
<td>Portable Object Adapter</td>
</tr>
<tr>
<td>QA</td>
<td>Q Adapter</td>
</tr>
<tr>
<td>QAF</td>
<td>Q Adapter Function</td>
</tr>
<tr>
<td>SDH</td>
<td>Synchronous Digital Hierarchy</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>SIF</td>
<td>SONET Interoperability Forum</td>
</tr>
<tr>
<td>SMA</td>
<td>Systems Management Architecture</td>
</tr>
<tr>
<td>SMI</td>
<td>Structure of Management Interface</td>
</tr>
<tr>
<td>SML</td>
<td>Service Management Layer</td>
</tr>
<tr>
<td>SMK</td>
<td>Shared Management Knowledge</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>SONET</td>
<td>Synchronous Optical Network</td>
</tr>
<tr>
<td>TINA-C</td>
<td>Telecommunications Information Networking Architecture Consortium</td>
</tr>
<tr>
<td>TMN</td>
<td>Telecommunications Management Network</td>
</tr>
<tr>
<td>XOM</td>
<td>X/Open OSI-Abstract-Data Manipulation</td>
</tr>
<tr>
<td>XMP</td>
<td>X/Open Management Protocol</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WS</td>
<td>Workstation</td>
</tr>
<tr>
<td>WSF</td>
<td>Workstation Function</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
</tbody>
</table>
4. Resumo

As empresas provedoras de serviços de telecomunicações têm requerido de seus fornecedores ambientes integrados de gerenciamento capazes de garantir a interoperabilidade entre seus sistemas, ditos ambientes heterogêneos, o que seria alcançado através da utilização das normas propostas pela ITU-T para o gerenciamento de redes de telecomunicações: o TMN.

Por outro lado, a gerência de redes é essencialmente uma aplicação distribuída e por isso a utilização da tecnologia de objetos distribuídos, como o CORBA, tem se tornado naturalmente uma opção das empresas desenvolvedores de aplicações de gerenciamento de redes. No caso específico de TMN, a dificuldade reside em identificar como modelar os objetos gerenciáveis definidos pelo TMN como objetos CORBA e também como utilizar os serviços e facilidades definidos para o CORBA nas aplicações de gerência TMN.

Consórcios industriais e organismos internacionais de padronização têm se esforçado para integrar as funcionalidades da infra-estrutura do CORBA no ambiente TMN. As vantagens do CORBA, um sistema orientado a objetos com APIs bem definidas e padronizadas, poderão, desta forma, ser combinadas com a rigidez do CMIP, um protocolo de transferência de informações de gerenciamento TMN, em um ambiente heterogêneo.
5. Introdução

As empresas de telecomunicações estão passando por mudanças muito rápidas. Com as novas tecnologias, a desregulamentação e a crescente demanda dos clientes, elas têm uma vasta gama de novas oportunidades e desafios, desde a demanda de vídeo e a multimídia interativa até o mercado eletrônico.

A medida que as empresas têm unificado suas redes e seus sistemas, elas necessitam novas tecnologias e novos sistemas para viabilizar estas mudanças. Isto não tem sido uma tarefa fácil já que estas redes podem conter sistemas analógicos e digitais, equipamentos de vários fornecedores, diferentes tipos de sub-redes e vários protocolos de gerenciamento.

O TMN provê um ambiente de desenvolvimento que permite a interconectividade e a comunicação entre sistemas operacionais heterogêneos e redes de telecomunicações. TMN foi desenvolvido pela ITU-T como sendo uma infra-estrutura para, dinamicamente, suportar o gerenciamento e o desenvolvimento de serviços de gerência de redes de telecomunicações.

Paralelamente, a indústria de software de orientação a objetos atualmente apresenta uma variedade de modelos de arquitetura de objetos que provêm importantes capacidades de distribuição, mesmo com objetivos às vezes um pouco contraditórios.

Estão-se convergindo estes diferentes objetivos para prover a usuários cada vez melhores ferramentas e infra-estruturas para desenvolver sistemas corporativos padronizados com o máximo de desempenho e de gerenciabilidade.

5.1 Ambientes TMN e CORBA

Para o gerenciamento de redes de telecomunicações, de serviços e de equipamentos, a ITU-T propôs a série M.3000 de recomendações definindo o TMN. Atualmente a tecnologia TMN está sendo aplicada amplamente no gerenciamento de redes SDH, assim como de redes ATM e de redes sem fio.

O TMN define basicamente três arquiteturas: arquitetura física, arquitetura de informação e arquitetura funcional. As arquiteturas física e funcional são inerentes do TMN enquanto que a arquitetura de informação é baseada no ambiente de gerenciamento OSI (série de recomendações X.700 da ITU-T), onde o CMIS e o CMIP são utilizados para viabilizar a troca de informações de gerência entre gerentes e agentes.

A arquitetura física especifica como os componentes físicos, tais como as informações de rede, de equipamentos de rede e de sistemas operacionais, são configurados.

A arquitetura funcional especifica como as várias funções do TMN, tais como OSFs, MFs e NEFs, são configuradas para possibilitar as funcionalidades de gerenciamento.

Por outro lado, o CORBA tem sido amplamente adotado no desenvolvimento de sistemas distribuídos em todas as áreas de tecnologia de informação, entre as quais encontram-se os serviços e as aplicações de gerenciamento de redes de telecomunicações.

O CORBA provê a infra-estrutura para a interoperabilidade de aplicações de gerência orientadas a objetos em um ambiente distribuído, e, recentemente, têm havido esforços crescentes para a integração do CORBA com o ambiente TMN.

O padrão do CORBA, especificado pela OMG, tem se tornado o ambiente natural para suportar não apenas a implementação dos OSs do TMN mas também a implementação da interface de comunicação entre os OSs e entre o OS e os NEs.
O CORBA também tem sido efetivamente utilizado na incorporação de novas tecnologias emergentes, tais como WWW e Java, dentro dos sistemas de gerenciamento de redes.

Os esforços para integrar as funcionalidades da infra-estrutura do CORBA no ambiente TMN têm sido dirigidos por consórcios industriais, tais como a OMG, o TMF e a X/Open, por organismos internacionais de padronização, tais como ISO e ITU-T, e por algumas empresas, tais como a HP, a DSET, a Bell Laboratories.

O grupo JIDM é um esforço conjunto de padronização conduzido pela X/Open e pelo TMF. O objetivo do JIDM é integrar as tecnologias do CMIP, do SNMP e do CORBA, focando-se especialmente na interoperabilidade CMIP/CORBA e SNMP/CORBA.

Já o TINA-C define uma arquitetura baseada em CORBA a ser empregada especificamente nas aplicações de gerência de serviços para o mercado de gerenciamento distribuído de TMN. A definição desta gerência de serviços é baseada em um modelo que identifica os papéis que compõem o serviço e o relacionamento entre eles.

Paralelamente, a OMG tem apresentado uma proposta de especificação de uma arquitetura para sistemas TMN baseados em CORBA. Esta arquitetura inclui uma implementação alternativa para as interfaces OSI e para os sistemas de gerência da ISO baseada no paradigma do CORBA.

Em resposta a estas propostas de padronização, o ITU-T tem definido como um de seus objetivos a concretização de um redirecionamento da especificação de TMN. Neste TMN, o ITU-T pretende agregar em um grupo de estudos a pesquisa de uma arquitetura integrada de gerenciamento distribuída, chamada OTMA. A OTMA será baseada na arquitetura ODMA especificada conjuntamente pela ISO e ITU-T.

5.2 Integração da Infra-estrutura CORBA com o Ambiente TMN

As vantagens do CORBA, um sistema orientado a objetos com APIs bem definidas e padronizadas, simplificando a implementação de aplicações distribuídas, podem ser combinadas com a rigidez do CMIP, um protocolo de transferência de informações de gerenciamento TMN, em um ambiente heterogêneo.

O CORBA também permite reutilizar padrões definidos por outros organismos, tais como ISO e ITU-T, tornando suas especificações úteis para sistemas TMN baseados em CORBA. Ele possibilita aos desenvolvedores de sistemas implementar soluções de gerenciamento utilizando tanto IDL quanto GDMO/ASN.1, de uma forma independente de protocolo, resultando em um solução de gerência integrada confiável e eficiente em ambientes heterogêneos.

A integração de CORBA em um ambiente TMN tem trazido alguns problemas relativos à interoperabilidade entre protocolos e aplicações de gerenciamento de redes de telecomunicações proprietárias, assim como em relação a requisitos de qualidade de serviço e de confiabilidade.

Este artigo apresenta uma breve descrição dos conceitos de TMN e de CORBA e vários tópicos relativos à integração destes ambientes, descrevendo algumas propostas de concepção de integração destas arquiteturas e ambientes.
6. TMN

6.1 Estrutura do TMN

TMN oferece uma infra-estrutura de desenvolvimento para redes que é flexível, escalonável, confiável e facilmente ampliável, pois padroniza formas de execução de procedimentos de gerência de redes e de comunicação através de redes. TMN permite a distribuição de processamento e assim permite obter níveis apropriados de desempenho e de eficiência de comunicação.

Os princípios de TMN devem ser incorporados numa rede de telecomunicações para enviar e receber informações e para gerenciar seus recursos ([1]). Uma rede de telecomunicações é composta por sistemas de comutação, circuitos de transmissão, terminais, etc. Na terminologia TMN, estes recursos são chamados elementos de rede (NEs).

TMN é por definição uma rede de computadores paralela à rede de equipamentos de telecomunicações, a qual permite a comunicação entre sistemas de suporte de operações (OSS) e elementos de rede (NE).

![Figura 1: TMN e a Rede de Telecomunicações](attachment:TMN диаграмма.png)

6.1.1 Padrão TMN

TMN está definido pela ITU-T, na série de recomendações M.3000, e especialmente na recomendação M.3010 [2]. Quando redes de telecomunicações implementam as definições TMN elas tornam-se interoperáveis, mesmo quando interagindo com redes e equipamentos de outros fornecedores e/ou provedores de serviços de telecomunicações.

TMN usa princípios de orientação a objetos e interfaces padrões para definir a comunicação entre entidades gerenciáveis em uma rede. A interface de gerência padrão para TMN é chamada interface Q.

A arquitetura e as interfaces TMN foram baseadas nos padrões existentes da ISO. Estes padrões incluem:
- CMIP [3]: define os serviços de gerência compartilhados entre entidades próximas;
- GDMO [4]: provê modelos para classificar e descrever recursos gerenciáveis;
- ASN.1 [5]: provê regras de sintaxe para os tipos de dados;
- **Modelo de referência de sistemas abertos**: define as sete camadas OSI do modelo de referência.

A tabela a seguir lista as recomendações genéricas TMN, editadas pela ITU-T, que são basicamente uma especialização das recomendações da série X.700 da ITU-T para gerência de sistemas:

<table>
<thead>
<tr>
<th>Recomendação ITU-T</th>
<th>Número</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview of TMN Recommendations</td>
<td>M.3000</td>
</tr>
<tr>
<td>Principles for a Telecommunications Management Network</td>
<td>M.3010</td>
</tr>
<tr>
<td>TMN Interface Specification Methodology</td>
<td>M.3020</td>
</tr>
<tr>
<td>Generic Network Information Model</td>
<td>M.3100</td>
</tr>
<tr>
<td>Managed Object Conformance Statements for the Generic Network Information Model</td>
<td>M.3101</td>
</tr>
<tr>
<td>Catalogue of TMN Management Information</td>
<td>M.3180</td>
</tr>
<tr>
<td>TMN Management Services: Overview</td>
<td>M.3200</td>
</tr>
<tr>
<td>TMN Management Capabilities Presented at the F Interface</td>
<td>M.3300</td>
</tr>
<tr>
<td>Management Requirements Framework for the TMN X Interface</td>
<td>M.3320</td>
</tr>
<tr>
<td>TMN Management Functions</td>
<td>M.3400</td>
</tr>
</tbody>
</table>

Desde a sua publicação, os padrões TMN têm sido promulgados por outros organismos de normatização, principalmente o TMF, a Bellcore e a ETSI ([6]). Em geral, os esforços do TMF e da Bellcore são direcionados à implementação e à disponibilização de um ambiente genérico de estabelecimento de requisitos detalhados de gerência de redes de telecomunicações, enquanto que os esforços da SIF e do Fórum ATM estão direcionados à especificação de interfaces de gerência TMN específicas.

### 6.1.2 TMN, OSI e Gerência

TMN está baseado na infra-estrutura de gerenciamento especificado pela ISO (OSI) e usa orientação a objetos, onde as informações de gerência dos recursos de rede são modeladas como atributos de objetos gerenciados. As funções de gerência são executadas por operações especificadas pelas primitivas CMIS [7].

A informação de gerência de uma rede de telecomunicações, assim como as regras pelas quais esta informação é apresentada e gerenciada, é referenciada como sendo a base de informações de gerência (MIB). Aplicações que gerenciam estas informações são chamados de entidades de gerenciamento. Uma entidade de gerenciamento pode assumir um dos seguintes dois papéis: gerente ou agente. Gerentes e agentes enviam e recebem requisições e notificações usando o protocolo CMIP.

Os benefícios de TMN são importantes pois permitem empresas gerenciar redes e serviços complexos e dinâmicos, assim como permitem que estas mesmas empresas continuem expandindo seus serviços e mantendo a sua qualidade.

TMN descreve redes de gerência de telecomunicações desde vários pontos de vista: através do modelo lógico ou do modelo de negócio, através do modelo funcional e através de um conjunto de interfaces padrão.

### 6.2 Modelo Funcional do TMN

TMN permite aos provedores de serviços de telecomunicações viabilizar a interconectividade e a comunicação entre OSSs e redes de telecomunicações. A interconectividade é conseguida através de interfaces padrão capazes de abstrair os recursos gerenciados na forma de objetos gerenciáveis.
6.2.1 Blocos TMN

TMN é representada por vários blocos funcionais que provêem um encapsulamento de funcionalidades de gerenciamento.

Figura 2: Blocos Funcionais do TMN

A tabela a seguir lista e descreve cada um dos componentes funcionais definidos pela TMN, com a sua respectiva funcionalidade:

<table>
<thead>
<tr>
<th>Bloco Funcional</th>
<th>Descrição</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>Executa funções do sistema operacional, inclusive monitoração de operações e controle de funções de gerência de telecomunicações. Um OS pode prover também algumas funcionalidades de MD, QA, WS.</td>
</tr>
<tr>
<td>MD</td>
<td>Provê a mediação entre interfaces TMN locais e o modelo de informações do OS. A função de mediação pode ser necessária para garantir que informações, escopo e funcionalidades estejam presentes precisamente na forma que o OS espera. Funções de mediação podem ser implementadas através de MDs em cascata.</td>
</tr>
<tr>
<td>QA</td>
<td>A QA permite a uma TMN gerenciar NEs que não implementem interfaces TMN: é um tradutor entre interfaces TMN e não-TMN.</td>
</tr>
<tr>
<td>NE</td>
<td>No escopo de TMN, um NE contém informações gerenciáveis que são monitoradas e controladas por um OS. Para permitir ser gerenciável dentro do escopo de TMN, um NE deve ter uma interface TMN padrão. Por outro lado, se um NE não possuir uma interface padrão, ele ainda pode ser gerenciado mas através de um QA. Um NE provê ao OS uma representação das suas informações e funcionalidades gerenciáveis (MIB). Um NE pode prover também algumas funcionalidades de OS, QA, MD.</td>
</tr>
<tr>
<td>WS</td>
<td>A WS executa funções de estação de trabalho. Ele traduz informações do formato TMN e as disponibiliza num formato apresentável ao usuário.</td>
</tr>
<tr>
<td>DCN</td>
<td>A DCN é a rede de comunicações dentro da TMN. Ela representa as camadas 1 à 3 da pilha OSI.</td>
</tr>
</tbody>
</table>

6.2.2 Distribuição das Funções de Gerência TMN

Um tipo de atividade de gerenciamento pode ser dividida em uma série de funções, cada qual pode existir em mesmos níveis lógicos de gerência ou em diferentes. Cada função de gerência é controlada por um único OS, podendo ocorrer procedimentos gerências distribuídos já que o gerenciamento TMN pode estar dividida entre vários OSs.
Os níveis lógicos de gerenciamento podem estar distribuídos em vários OSs subordinados dentro de um ambiente TMN.

6.2.3 Funcionalidade OSI para TMN

TMN define uma função de comunicação de mensagens: a MCF. Todos os blocos funcionais com interfaces físicas precisam ter uma MCF, a qual provê as camadas de protocolo de comunicações necessárias para conectá-los em uma rede de dados (camadas de 4 a 7).

Uma MCF pode prove todas as 7 camadas OSI e também as funções de convergência de protocolo para interfaces que usam outras configurações de camadas (outros protocolos de comunicação).

Os blocos funcionais podem atuar no papel de um agente ou de um gerente. Os conceitos TMN de agente e de gerente são os mesmos dos usados pelo CMIP e pela gerência OSI, de acordo com a figura a seguir.

![Figura 3: Interação entre Gerente, Agente e Objetos Gerenciáveis](image)

Uma aplicação TMN no papel de gerente faz requisições de operações e recebe notificações, enquanto que no papel de agente processa operações, envia respostas e emite notificações. Esta aplicação TMN pode fazer ao mesmo tempo o papel de gerente para uma outra aplicação agente e o papel de agente para uma outra aplicação gerente.

Além dos blocos funcionais e dos pontos de referência, a arquitetura funcional do TMN inclui alguns conceitos adicionais. Estes conceitos são:

- DCFs;
- Componentes funcionais.

Os DCFs são usados pelos componentes funcionais para permitir a troca de informações, não sendo modelados como se fossem blocos funcionais. O DCF provê apenas as camadas 1 a 3 do modelo da pilha OSI.

Cada bloco funcional do TMN é composto por um número de componentes funcionais, como representado na próxima figura. Os seguintes componentes funcionais já estão caracterizados:

- Função de aplicação de gerência (relacionada com gerenciamento);
- MIB (relacionada com gerenciamento);
- Função de conversão de informações (relacionada com gerenciamento);
- Adaptação homem-máquina (relacionada com gerenciamento);
- Função de apresentação (relacionada com gerenciamento);
- MCF (relacionada com o serviço de troca de informações de gerência).
Objetos gerenciáveis com mesmas propriedades são instâncias de uma classe de objetos. A MIB é um repositório conceitual de instâncias de objetos gerenciáveis. Uma classe de objeto gerenciável é definida pelos:

- **Atributos**: são elementos de dados e valores que caracterizam uma classe de objetos gerenciáveis;
- **Operações de gerência**: são as operações que podem ser aplicadas às instâncias de objetos gerenciáveis;
- **Comportamento**: é o que é apresentado por uma instância de objeto gerenciável baseado no recurso que ele representa;
- **Notificações**: são as mensagens que instâncias de objetos gerenciáveis podem emitir espontaneamente.

### 6.3 Interfaces Padrão do TMN

No modelo TMN, interfaces específicas entre dois componentes do TMN comunicam-se através das seguintes interfaces padrão:

<table>
<thead>
<tr>
<th>Interface TMN</th>
<th>Descrição</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q</td>
<td>A interface Q existe entre dois blocos funcionais padrão TMN que estejam dentro de um mesmo domínio TMN. A interface Q passa informação compartilhada entre o MD e os NEs que ela suporta e é a interface do OS: qualquer componente funcional que se comunica diretamente com um OS usa a interface Q.</td>
</tr>
<tr>
<td>F</td>
<td>A interface F existe entre uma WS e um OS ou entre uma WS e um MD.</td>
</tr>
<tr>
<td>X</td>
<td>A interface X existe entre dois OSs padrão TMN mas localizados em domínios TMN distintos.</td>
</tr>
</tbody>
</table>

Há ainda dois outros tipos de pontos de referência, g e m, mas que estão fora do escopo do TMN. Estes pontos de referência encontram-se entre entidades não-TMN.
A figura a seguir ilustra quais componentes se comunicam via a interface Q.

A interface Q é a forma de comunicação dos OSs: Q é a única interface que QAs, MDs ou NEs podem utilizar para comunicarem-se diretamente com os OSs. Se um QA ou um NE não suportar a interface Q, ele não pode se comunicar diretamente com um OS, devendo fazê-lo através de um MD.

Os padrões de gerenciamento de sistemas da OSI provêm a flexibilidade requerida na definição de interfaces padrão mas não são propriamente as interfaces requeridas pelos sistemas de gerenciamento TMN. Estas interfaces compreendem os padrões genéricos e os padrões dependentes de tecnologia.

Os padrões genéricos são supostos a se aplicarem para todas as tecnologias de telecomunicações e serviços. Um exemplo clássico é a recomendação ITU-T M.3100 [10] que contém a definição dos
objetos gerenciáveis suficientemente genéricos para descrever a informação compartilhada por todas as interfaces TMN, independente da tecnologia de telecomunicação referenciada.

Os objetos especificados nos padrões dependentes de tecnologia são normalmente importados dos padrões genéricos ou são subclasses de objetos genéricos.

Herança é o procedimento de especificar uma nova classe de objeto baseado em uma classe de objeto previamente definida. Desta forma, a nova classe de objeto possui todas as características da classe de objeto base com algumas novas características. Esta política de derivar classes de objetos de tecnologias específicas de classes de objetos genéricas garante um nível de similaridade entre diferentes modelos de informação de tecnologias específicas.

A herança, junto com os conceitos de padrões genéricos e dependentes de tecnologia, é o mecanismo que permite a generalidade e a consistência desejável entre interfaces TMN.

### 6.4 Modelo Lógico do TMN

TMN provê um modelo de camadas lógicas que define níveis gerências para funcionalidades específicas. Os mesmos tipos de funções podem ser implementados em várias outras camadas, desde um nível mais alto que gerência assuntos corporativos até um nível mais baixo que pode ser definido por uma rede ou por um recurso de rede.

Para lidar com a complexidade inerente ao gerenciamento ([13]), a funcionalidade de gerência, com sua informação associada, pode ser decomposta entre as camadas lógicas, como está representado na figura a seguir:

![Figura 7: Decomposição da Funcionalidade de Gerência](image)

A partir de camada mais inferior, estas hierarquias incluem NEs, gerência de elementos de rede (EML), gerência de redes (NML), gerência de serviços (SML) e gerência de negócios (BML):

<table>
<thead>
<tr>
<th>Camada</th>
<th>Responsabilidade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gerência de negócios</td>
<td>Coordena o planejamento de alto nível, as cobranças, as estratégias, as decisões executivas, os contratos de negócios (BLA).</td>
</tr>
<tr>
<td>Gerência de serviços</td>
<td>Utiliza informação disponibilizada pela camada de gerência de rede para gerenciar contratos de serviço de clientes existentes e em potencial, desde o aprovisionamento e a qualidade de serviço, até a gerência de falhas. A SML é também responsável pela interação com os provvedores de serviços e com outros domínios administrativos, mantendo dados estatísticos para garantir a qualidade do serviço prestado.</td>
</tr>
<tr>
<td>Gerência de rede</td>
<td>Tem a visibilidade de toda a rede baseada em informações de NEs disponibilizadas pelos OSs da camada de gerência de elemento de rede. A NML coordena todas as atividades de rede e suporta as requisições da SML.</td>
</tr>
</tbody>
</table>
Gerência de elemento de rede

Gerencia cada elemento de rede. A EML possui OSs, cada um dos quais é responsável pelas informações gerenciáveis de NEs específicos. De uma forma geral, um gerente de elemento de rede é responsável por um subconjunto dos elementos de rede, gerenciando os seus dados, atividades, registros, etc.

Elemento de rede

Apresenta as informações gerenciáveis TMN de cada um dos NEs individualmente. A NEL faz a interface entre a informação gerenciável proprietária e a infra-estrutura TMN.

Na figura a seguir é apresentado um exemplo de como componentes TMN de diferentes níveis lógicos de gerenciamento podem estar relacionados.

Figura 8: Exemplo de Componentes TMN Distribuídos nas Camadas Lógicas

### 6.5 Áreas Funcionais de Gerenciamento

A ITU-T dividiu o gerenciamento de redes de telecomunicações em áreas funcionais de gerenciamento permitindo desta forma identificar diferentes atividades características da gerência TMN. Foram definidas cinco áreas funcionais básicas:

- **Gerência de Falhas**: permite a detecção, isolamento e correção de anomalias de uma rede de telecomunicações e de seus equipamentos;
- **Gerência de Configuração**: permite o controle, a identificação e a coleta de dados de NEs e de conexões entre eles, assim como o planejamento, a instalação e a configuração dos equipamentos da rede de forma a garantir os serviços requeridos pelos clientes de uma rede de telecomunicações;
- **Gerência de Bilhetagem**: habilita o uso dos serviços da rede para medição e determinação de custos, provendo facilidades de definição de parâmetros de bilhetagem e de coleta de registros de cobrança do uso de uma rede de telecomunicações;
- **Gerência de Desempenho**: permite a geração e a avaliação de relatórios de dados coletados de uma rede de telecomunicações de acordo com requisitos de qualidade de serviço requeridos pelos usuários da rede e de seus equipamentos;
• **Gerência de Segurança**: permite prevenir e detectar o uso impróprio ou não autorizado de recursos de uma rede de telecomunicações, assim como administrar a sua segurança.

### 6.6 Serviços de Gerência TMN

Um serviço de gerenciamento pode ser definido como sendo uma oferta de funcionalidades compatíveis com as necessidades de gerenciamento de uma rede de telecomunicações para um usuário específico.

Vários serviços de gerência foram identificados, tais como administração de clientes e gerência de tráfego, assim como características que necessariamente devem ser identificadas para sua caracterização. Estes serviços e suas correspondentes características estão descritas na recomendação ITU-T M.3200 [9].

### 6.7 Soluções TMN

O objetivo chave para provedores de serviço, fornecedores de software e integradores de sistemas é desenvolver aplicações TMN robustas capazes de executar operações de gerência de redes variadas, em redes de várias plataformas, vários fornecedores e em constante atualização.

Provedores de serviços e fornecedores independentes de software precisam desenvolver soluções capazes de:

- Reduzir o tempo de desenvolvimento de aplicações TMN;
- Reduzir custos;
- Suportar demandas crescentes do aumento da qualidade do software;
- Incorporar novos equipamentos;
- Estar em conformidade com os padrões do TMN.

### 6.7.1 Integração de Gerência de Equipamentos

Sistemas e equipamentos de rede, os quais são normalmente não conformes com as normas do TMN, comunicam-se apenas através de protocolos de gerência proprietários.

Ao definir interfaces padrão, o TMN não obriga que elementos de rede sejam substituídos por hardware e sistemas de gerência conformes com CMIP. Mas, ao prever a utilização de elementos de mediação e de QAs, o TMN possibilita às empresas migrarem todos os seus sistemas e equipamentos para uma solução de gerência distribuída, escalável e interoperável.

Um QA pode impor uma estrutura padrão, legível pelo equipamento e orientada a objetos, a partir de um modelo de informação proprietário. Desta forma, ao derivar informação, interpretando-a a partir de um modelo proprietário, um desenvolvedor de sistemas de gerenciamento pode disponibilizar uma interface legível padrão para um OS.

Outro uso de um QA é a integração de toda uma rede LAN proprietária em uma infra-estrutura TMN. A maior parte das redes LAN gerenciam seus recursos através de serviços SNMP. Ao adaptar SNMP a uma rede de gerenciamento CMIP, uma rede LAN proprietária pode ser integrada a um ambiente de gerência TMN.

A adaptação provida por um QA costuma ser suficiente para converter a interface de NEs para outro padrão utilizada por um OS. Mas, várias vezes, é necessário algum tipo de refinamento adicional, o qual é obtido através da utilização de dispositivos de mediação. Logo, os MDs podem reconstituir ou impor estruturas na sua interface de tal forma que tornem-se compatíveis com o formato esperado por uma aplicação de um OS.
6.7.2 Interface de Programação Padrão Orientada a Objetos

O TMF introduziu uma API padrão baseada no paradigma da orientação a objetos para o TMN, a qual foi chamada TMN/C++ API ou TMF API. A arquitetura completa compreende três módulos de camadas de APIs:

- **Interface de objetos gerenciáveis GDMO/C++ API**: provê um ambiente de acesso e de implementação de objetos gerenciáveis em um modelo hierárquico de árvore;
- **Interface de serviço CMIS/C++ API**: provê os serviços básicos de gerenciamento de informações para o envio de pedidos e o recebimento de respostas, assim como para o envio de eventos que ocorram na rede de gerenciamento;
- **Interface de dados ASN.1/C++ API**: define a interface para todos os dados e para a sua codificação.

![Figura 9: API TMN/C++ da TMF](image)

Antes da API do TMF já haviam outras APIs padrão aceitas para as aplicações TMN. A mais comum foi a XMP/XOM da X/Open, a qual provia uma API C para o gerenciamento de objetos.

A utilização do XMP/XOM no domínio do TMN foi desestimulada pelo seu alto grau de abstração e dificuldade de uso, sendo que normalmente plataformas TMN que implementavam as APIs XMP/XOM também ofereciam outra API que as encapsule.

A API da TMF elimina as dificuldades das APIs como a da X/Open e provê um desenvolvimento baseado em interfaces orientadas a objetos, em C++, e independentes de plataforma. Esta API tem as seguintes vantagens:

- É independente de plataforma com código C++ reusável, de forma que códigos de interface possam ser utilizados por várias aplicações;
- Suporta dois tipos de aplicações: específico (implementam um modelo de informações estático) e genérico (dinamicamente interpretam mudanças dentro do modelo de informações de uma rede de gerenciamento);
- É fácil de ser usada pois, com o paradigma da orientação de objetos, somente a interface é exposta escondendo todas as complexidades inerentes aos tipos de dados e aos métodos de sua manipulação;
- Provê uma representação completa do modelo TMN para GDMO, ASN.1 e CMIS.
6.7.3 Ferramentas de Automação de Implementação de Aplicações TMN

Há ferramentas disponíveis que automatizam significativamente o desenvolvimento de aplicações TMN, tanto de agentes e como de gerentes. Estas ferramentas podem ser implementadas e personalizadas para representar uma MIB específica de uma empresa. Estes produtos devem fornecer várias das seguintes funcionalidades e assim maximizar as vantagens do TMN e suportar de forma produtiva sua infra-estrutura:

- **Modelagem de informação dinâmica**: ferramentas que têm a habilidade de adicionar, alterar ou remover a configuração ou funcionalidade da rede de gerenciamento sem precisar reinstalar ou recompilar as aplicações TMN;
- **Prototipação automática**: ferramentas que podem compilar modelos de informação GDMO/ASN.1 e produzir interfaces C++ específicas do modelo de objetos e da estrutura dos seus dados;
- **Funções de gerenciamento de sistemas**: ferramentas que permitem gerar, filtrar, encaminhar e registrar eventos e alarmes recebidos;
- **Ferramentas e interfaces independentes de plataforma**: ferramentas de testes e de simulação que assumem papéis de gerente e de agente, assim como ferramentas para a criação e o desenvolvimento de modelos de informação (MIBs);
- **Capacidade de adaptação Q**: ferramentas que têm a habilidade de se tornar interface e desta forma integrar NEs proprietários assim como sistemas de redes LAN;
- **Conformidade com padrões TMN**: ferramentas que permitem a implementação de serviços, dados e camadas de objetos gerenciáveis de uma API TMF, e suporte para o desenvolvimento de aplicações do tipo específicas e genéricas.
7. CORBA

Diferentes modelos de objetos têm surgido a partir de vários tipos distintos de requisitos. Como exemplos, OLE/COM provê interfaces para alguns sistemas operacionais e aplicações; Smalltalk provê um ambiente bem superior de aplicações orientadas a objetos; C++ provê o meio para gerar sistemas, infra-estruturas e componentes orientados a objetos.

Um novo padrão importante é o CORBA, especificado pela OMG. O CORBA está estruturado para permitir a integração de uma vasta variedade de sistemas de objetos através de uma arquitetura que viabiliza a comunicação entre objetos, os quais são definidos através de interfaces descritas com uma linguagem neutra de especificação de interfaces chamada IDL ([11]).

7.1 OMG

A OMG é uma organização internacional de mais de 800 membros. Fundada em 1989, a OMG tem promovido a teoria e a prática da tecnologia de orientação a objetos na análise e no desenvolvimento de software.

Os objetivos da organização são o estabelecimento de normas industriais e a criação de especificações de gerência de objetos para prover uma plataforma comum para o desenvolvimento de aplicações, através do uso da reusabilidade, portabilidade e interoperabilidade de software baseado em objetos em sistemas distribuídos e heterogêneos.

Conformidade com estas especificações tornam possível desenvolver um ambiente de aplicações heterogêneas entre a grande maioria dos hardwares e dos sistemas operacionais.

O objetivo da OMG é acompanhar o crescimento da tecnologia de objetos e influenciar a sua direção estabelecendo a especificação da arquitetura para o desenvolvimento de aplicações de gerência, a OMA. A OMA provê a infra-estrutura conceitual sobre a qual se baseiam as especificações da OMG.

7.2 Especificações Relacionadas

A documentação sobre o CORBA editada pela OMG inclui um conjunto de especificações reunidas nos seguintes livros:

- **Object Management Architecture Guide**: define os objetivos técnicos da OMG e a sua terminologia, além de descrever os modelos conceituais sobre os quais os padrões da OMG se baseiam. A OMA também fornece informações sobre as políticas e os procedimentos da OMG, tais como a forma que os padrões são propostos, avaliados e aceitos;
- **CORBA: Common Object Request Broker Architecture and Specification**: contém a arquitetura e a especificação do ORB;
- **CORBA Services: Common Object Services Specification**: contém a especificação dos serviços de objetos da OMG;
- **CORBA Facilities: Common Facilities Architecture and Specification**: descreve um arquitetura para as facilidades comuns do CORBA, além de incluir as especificações baseadas na sua arquitetura que tem sido adotada e publicada pela OMG.

As especificações são adotadas como padrões somente quando os representantes dos membros da OMG aceitam por votação.

7.3 OMA

A OMA é o centro de toda atividade em que a OMG está diretamente envolvida ([14]). A OMG foi formada para ajudar a reduzir a complexidade, reduzir os custos e suportar a introdução de um novo
conceito no desenvolvimento de aplicações de software. Isto tem sido feito através da introdução de um ambiente arquitetural (OMA) com a especificação de interfaces detalhadas.

As implementações propriamente ditas são da competência das empresas de software, de usuários finais e daqueles que estiverem desenvolvendo produtos de software e projetos para solução de problemas específicos.

As especificações são de responsabilidade dos membros da OMG. Elas encaminham a indústria para garantir a interoperabilidade, reusabilidade, portabilidade de componentes de software e padronização de interfaces orientadas a objetos.

7.3.1 Modelo de Referência da OMA

O Modelo de Referência da OMA partiona a proposta da OMG em um grupo prático de componentes arquiteturais que podem ser tratados por diferentes membros do grupo. Ele forma uma organização conceitual para compilar as tecnologias propostas resultantes, enquanto permite diferentes soluções de especificações.

O Modelo de Referência identifica e caracteriza os componentes, as interfaces e os protocolos que compõe a OMA, mas sem entrar em detalhes específicos de implementação. A OMA pode ser visualizada em três segmentos principais constituídos de cinco componentes críticos:

- **Orientado a aplicação**: a OMA caracteriza Interfaces e CORBAfacilities comuns como sendo componentes específicos de uma aplicação o mais próximo possível dos requisitos do usuário final;
- **Orientado a sistema**: ORBs e CORBAservices estão mais ligados ao conceito de sistema ou de aspectos de infra-estrutura dos objetos distribuídos;
- **Orientado a mercados verticais**: Interfaces de Dominios são aplicações verticais ou interfaces de algum domínio específico, que provêem ambientes de aplicações críticas para uma grande variedade de indústrias.

Independente de qual segmento se está particularmente interessado, é importante ressaltar que toda comunicação entre os componentes é gerenciada pelo ORB, o fundamento da OMA. A tecnologia de ORB (infra-estrutura representada na figura a seguir) é hoje considerada como sendo a mais importante forma de se alcançar o desenvolvimento de soluções computacionais realmente abertas, distribuídas e heterogêneas.

![Figura 10: Estrutura Típica do CORBA](image-url)
É igualmente importante notar que a OMA assume que os serviços de suporte são fornecidos pela plataforma do sistema operacional e pelos serviços básicos de baixo nível, tais como facilidades computacionais de rede, e estão disponíveis e utilizáveis pelas implementações especificadas pela OMA.

7.4 Estrutura de um ORB

O Cliente é a entidade que deseja executar uma operação sobre um objeto e a Implementação do Objeto é o código e os dados que implementam o objeto. O ORB ([12]) é responsável por todos os mecanismos inerentes e necessários para:

- encontrar a Implementação do Objeto para um determinado comando a ser executado;
- preparar a Implementação do Objeto para receber o comando;
- prover a comunicação dos dados relativos ao comando em execução.

A interface que o Cliente vê é completamente independente de onde o objeto está localizado ou de qual linguagem de programação foi utilizada para a sua implementação.

A figura a seguir apresenta a estrutura básica de um ORB. As interfaces do ORB estão representadas pelas quatro cores listadas na legenda e as setas indicam quando o ORB é chamado ou quando executa uma chamada através de uma interface.

![Figura 11: Estrutura das Interfaces de um ORB](image)

Para executar um comando, o Cliente pode utilizar a interface de Invocação Dinâmica (uma mesma interface independente da interface do objeto destino) ou um Stub IDL (um Stub específico dependente da interface do objeto destino). O Cliente também pode interagir diretamente com o ORB através de algumas funções específicas.

A Implementação de um Objeto pode receber um pedido de comando tanto através de um Skeleton gerado a partir de uma IDL quanto através de um Skeleton dinâmico. Além disso, ela pode, quando necessário, chamar o adaptador de objeto ou o ORB enquanto estiver processando um comando.

Definições de interfaces de objetos podem ser especificadas de duas maneiras. Interfaces podem ser definidas estaticamente através de uma linguagem de definição de interface chamada IDL...
(especificada pela OMG). Esta linguagem define as características dos objetos de acordo com as operações que podem ser executadas por eles e incluindo os parâmetros que fazem parte destas operações.

Alternativamente, interfaces podem ser adicionadas a um serviço de Repositório de Interfaces, o qual contém os componentes de uma interface representados como objetos, permitindo assim acessos dinâmicos a estes componentes.

Em qualquer implementação de ORB, a representatividade da IDL e do Repositório de Interfaces são equivalentes.

O Cliente faz um pedido de comando para um objeto através de um acesso a uma Referência de Objeto, sempre conhecendo o tipo do objeto e as características do comando a ser executado. O Cliente inicializa este pedido chamando rotinas de um Stub específicas do objeto ou construindo o pedido de comando dinamicamente (veja figura a seguir).

![Diagrama do ORB](image)

Figura 12: Cliente Utilizando um Stub ou uma Interface de Invocação Dinâmica

Tanto a interface através de um Stub quanto a interface dinâmica utilizadas na requisição de mandados devem ter a mesma semântica de tal forma que quem recebe esta requisição não é capaz de identificar qual foi a forma em que foi invocada.

O ORB localiza a implementação do código do objeto apropriada, transmite os parâmetros e transfere o controle para a Implementação do Objeto através de um Skeleton IDL ou de um Skeleton dinâmico, representado na figura a seguir.
Os Skeletons são específicos às interfaces e aos adaptadores de objetos. Ao executar um pedido de comando, a Implementação dos Objetos pode receber algumas requisições de serviço do ORB através do Adaptador de Objeto. Quando a requisição for completada, o controle e os valores dos resultados são retornados ao Cliente.

A Implementação do Objeto pode escolher qual dos Adaptadores de Objeto que será utilizado. Esta decisão baseia-se basicamente em quais dos tipos de serviços que a Implementação de Objeto requer.

A figura a seguir mostra como as informações de implementação e de interface são geradas e disponibilizadas aos Clientes e às Implementações de Objetos. A interface é definida em IDL e/ou obtida do Repositório de Interfaces. A definição IDL é utilizada para gerar os Stubs dos Clientes e os Skeletons das Implementações dos Objetos.

A informação da Implementação do Objeto é obtida em tempo de instalação dos objetos e armazenada no Repositório de Implementações para ser utilizada durante o envio das requisições de comandos.
7.4.1 ORB

Na arquitetura, o ORB não precisa necessariamente ser implementado como um componente único, mas, ao contrário, ele é definido por suas interfaces. Logo, qualquer implementação de ORB que provê interfaces apropriadas é válida. As interfaces estão organizadas em três categorias:

- Operações que são as mesmas em todas as implementações de ORBs;
- Operações que são específicas para alguns tipos especiais de objetos;
- Operações que são específicas para alguns estilos especiais de implementações de objetos.

ORBs diferentes podem optar por implementações ligeiramente diferentes e, junto com compiladores IDL, repositórios e Adaptadores de Objetos, prover um conjunto de serviços para Clientes e Implementações de Objetos que têm propriedades diferentes.

Podem haver múltiplas implementações de ORBs as quais têm representações diferentes para Referências a Objetos e significados diferentes na invocação de comandos. É possível para um Cliente ter acesso simultâneo para duas Referências de Objeto gerenciadas por implementações diferentes de ORBs. Quando dois ORBs precisam trabalhar juntos, estes ORBs devem ser capazes de distinguir as cada uma das suas Referências a Objetos, não sendo responsabilidade do Cliente de fazer esta distinção.

O núcleo de um ORB é a parte que provê a representação básica das requisições dos objetos e da comunicação. O CORBA foi especificado para suportar mecanismos distintos de objetos, o que é possível por estruturar o ORB em módulos componentes ligados aos seus núcleos através de interfaces que encapsulam as diferenças entre eles.

7.4.2 Clientes

Um Cliente de um objeto tem acesso a uma referência deste objeto e é capaz de chamar seus comandos. Um Cliente conhece somente a estrutura lógica de um objeto de acordo com a especificação de sua interface e com o comportamento dos seus comandos.

Clientes geralmente vêem objetos e interfaces ORB através de uma perspectiva de mapeamento de linguagem, trazendo desta forma o ORB diretamente ao nível do programador. Clientes são portáveis e devem ser capazes de trabalhar sem mudanças de código em nenhum ORB que suporte o mapeamento da linguagem utilizada nas instâncias dos objetos que implemente esta interface.

Clientes não têm conhecimento da Implementação do Objeto ou de qual Adaptador de Objetos está sendo utilizado ou de qual ORB está acessando o objeto.

7.4.3 Implementações de Objetos

A Implementação de Objetos provê a semântica do objeto, normalmente definindo os dados para as suas instâncias e o código para os seus métodos. Normalmente a implementação utiliza outros objetos ou softwares adicionais para implementar o comportamento do objeto.

Pode-se suportar várias Implementações de Objetos, inclusive servidores separados, bibliotecas, programas específicos para métodos, aplicações encapsuladas, banco de dados orientados a objetos, etc. Através do uso de Adaptadores de Objetos adicionais, é possível suportar qualquer estilo de Implementação de Objetos.

Geralmente a Implementação de Objetos não depende do ORB ou de como os Clientes acessam os objetos. A Implementação de Objetos pode selecionar determinadas interfaces de serviços específicos de ORB ao escolher um Adaptador de Objeto.
7.4.4 Referências a Objetos

Uma Referência a Objeto é a informação necessária para especificar um objeto em um ORB. Tanto os Clientes quanto as Implementações de Objetos têm noção das Referências a Objetos de acordo com o mapeamento da linguagem. Implementações distintas de ORBs podem ter diferentes representações de Referências a Objetos.

A representação a um objeto utilizado por um Cliente é válida somente durante o tempo em que este Cliente estiver sendo executado.

Todos os ORBs devem prover a mesma linguagem de mapeamento a uma Referência a Objetos para uma específica linguagem de programação. Isto permite que um programa escrito em uma linguagem específica acesse Referências a Objetos independentemente do ORB.

Há ainda uma Referência a Objeto distinta, que deve ser garantidamente diferente de todas as Referências a Objetos existentes, que denote a ausência de objeto.

7.4.5 IDL da OMG

A linguagem de definição de interfaces (IDL) da OMG define os tipos dos objetos especificando suas interfaces. A interface consiste de um conjunto de operações e de seus parâmetros.

Apesar de a IDL prover a infra-estrutura conceitual para descrever os objetos a serem manipulados pelo ORB, não é necessário estar disponível o código fonte da IDL completa para que o ORB possa funcionar. A medida que a informação estiver sendo disponibilizada na forma de rotinas de Stubs, um ORB pode estar sendo capacitado para funcionar completamente.

IDL é a forma pela qual uma Implementação de Objeto descreve aos seus Clientes quais operações estão disponíveis e como elas podem ser invocadas. Das definições IDL, pode-se mapear objetos CORBA em uma linguagem de programação específica.

7.4.6 Mapeamento de IDL para Linguagens de Programação

Diferentes linguagens de programação, orientadas a objetos ou não, podem ser utilizadas para acessar objetos CORBA de diferentes formas. Para linguagens orientadas a objetos, é desejável visualizar os objetos CORBA como objetos da linguagem de programação.

Um mapeamento de IDL para uma linguagem de programação deve ser o mesmo para todas as implementações de ORBs. O mapeamento de uma linguagem inclui a definição de tipos de dados e de métodos específicos da linguagem para acessar objetos através do ORB. Isto inclui a estrutura da interface dos Stubs para os Clientes, da interface de invocação dinâmica, das implementações dos Skeletons, dos Adaptadores de Objetos e, diretamente, da interface do ORB.

O mapeamento da linguagem de programação também define a interação entre as invocações aos objetos e o controle dos processos no Cliente e na Implementação do Objeto. O mapeamento mais comum provê chamadas síncronas, nas quais a rotina retorna quando a operação no objeto estiver completa.

7.4.7 Stubs de Clientes

Para o mapeamento de linguagens não orientadas a objetos, há uma interface de programação dos Stubs para cada tipo de interface. Geralmente, os Stubs acessam as operações definidas por uma IDL em um objeto de forma a tornar fácil para um programador identificar um mapeamento de uma linguagem de programação.

Os Stubs executam chamadas a outros ORBs através de interfaces privativas e otimizadas para um ORB específico. Quando mais de um ORB estiver disponível, devem haver Stubs diferentes
correspondentes aos ORBs diferentes. Neste caso, é necessário que o ORB e o mapeamento da linguagem sejam capazes de permitir a associação entre cada um dos Stubs com uma Referência a Objeto específica.

Linguagens de programação orientadas a objetos, tais como C++ e Smalltalk, não necessitam interfaces de Stubs.

7.4.8 Interface de Invocação Dinâmica

Também está definida uma interface que permite a construção dinâmica de invocações a objetos, isto é, ao invés de chamar uma rotina de Stub específica para uma operação em particular em um objeto predeterminado, o Cliente pode especificar o objeto a ser invocado, a operação a ser executada e o conjunto de parâmetros para esta operação através de uma sequência de chamadas em tempo de execução.

O código do Cliente deve buscar as informações necessárias sobre a operação a ser executada e sobre os tipos dos parâmetros a serem passados à operação a partir do Repositório de Interfaces.

A forma de implementação da Interface de Invocação Dinâmica pode variar substancialmente dependendo do mapeamento de cada linguagem de programação.

7.4.9 Implementação de Skeletons

Para um mapeamento de uma determinada linguagem de programação e também dependendo do Adaptador de Objetos, há necessariamente uma interface para os métodos a qual implementa cada tipo de objeto.

A interface é geralmente a especificação da implementação do objeto, descrevendo as rotinas e as chamadas que o ORB faz através dos Skeletons.

A existência dos Skeletons não implica na existência de um Stub cliente correspondente (Clientes podem fazer requisições através da interface de invocação dinâmica).

É possível implementar um Adaptador de Objeto que não utilize Skeletons para invocar a implementação dos métodos. Por exemplo, pode ser possível criar implementações dinamicamente para linguagens tais como Smalltalk.

7.4.10 Interface Dinâmica de Skeletons

Há uma interface disponível que permite lidar dinamicamente com as invocações de objetos. Isto significa que ao invés de uma Implementação de Objeto ser acessada através de um Skeleton específico de uma operação em particular, ela é acessada através de uma interface que provê acesso ao nome das operações e dos parâmetros de forma análoga à Interface de Invocação Dinâmica do Cliente.

O código da implementação deve prover as descrições de todos os parâmetros ao ORB enquanto que o ORB provê os valores de quaisquer parâmetros de entrada a serem utilizados pela operação. Por outro lado, o código da implementação provê os valores de quaisquer parâmetros de saída (ou uma exceção) ao ORB depois de a operação ter sido completada.

A natureza da Interface Dinâmica de Skeletons pode variar substancialmente de um mapeamento de linguagem de programação para outro.

Skeletons dinâmicos podem ser invocados tanto por Stubs de Clientes quanto por Interfaces de Invocação Dinâmica, mas ambos estilos de requisições de acesso devem necessariamente prover os mesmos resultados.
7.4.11 Adaptadores de Objetos

Um Adaptador de Objeto é uma forma básica em que uma Implementação de Objeto acessa serviços providos por um ORB. Serviços providos por um ORB através de uma Adaptador de Objetos normalmente incluem:

- Geração de Referências de Objetos;
- Interpretação de Referências de Objetos;
- Invocação de métodos;
- Segurança das interações;
- Ativação de objetos;
- Desativação de objetos;
- Ativação de Implementações de Objetos;
- Desativação de Implementações de Objetos;
- Mapeamento entre Referências de Objetos e Implementações de Objetos;
- Registro de Implementações de Objetos.

A variedade de granularidade de objetos, ciclos de vida, políticas de criação e de manutenção, estilos de implementação, entre outras propriedades, dificulta sobremaneira ao ORB prover uma interface única conveniente e eficiente para todos as Implementações de Objetos. Desta forma, é através dos Adaptadores de Objetos que torna-se possível ao ORB alcançar um grupo de Implementações de Objetos específico que possui características em comum.

7.4.12 Interface de um ORB

A Interface de um ORB é a interface que não depende da interface dos objetos ou do Adaptador de Objetos.

Como a maior parte das funcionalidades de um ORB é provida através dos Adaptadores de Objetos, Stubs, Skeletons ou de invocações dinâmicas, há apenas poucas operações que são comuns para todos os objetos. Estas operações comuns são úteis tanto para os Clientes quanto para as Implementações de Objetos.

7.4.13 Repositório de Interfaces

O Repositório de Interfaces é o serviço que provê objetos persistentes que representam a informação da IDL em tempo de execução.

A informação contida no Repositório de Interfaces pode ser utilizada pelo ORB para executar as requisições de comandos. Além disso, esta informação possibilita a validação de operações requisitadas pelos Clientes e a sua consequente invocação.

Uma função adicional do Repositório de Interfaces para o ORB é que ele torna-se o local comum para armazenar informações adicionais associadas às Interfaces dos Objetos do ORB. Por exemplo, informações de depuração, bibliotecas de Stubs e de Skeletons, rotinas de formatação e de navegação, etc., podem estar associados a este repositório.

7.4.14 Repositório de Implementações

O Repositório de Implementações contém informação que permitem ao ORB localizar e ativar as Implementações dos Objetos.

Embora a maior parte da informação contida no Repositório de Implementações é específica para um ORB ou ambiente de operação, ele é o local convencionalmente utilizado para armazenar este tipo de informação. Normalmente, a instalação das Implementações dos Objetos e o controle das
políticas relativas à ativação e à execução das implementações é feita através de operações específicas no Repositório de Implementações.

Adicionalmente a esta funcionalidade no ORB, o Repositório de Implementações é o lugar comum para armazenar informações adicionais associadas à Implementação dos Objetos do ORB. Por exemplo, informações de depuração, controle administrativo, alocação de recursos, segurança, etc., podem estar associadas a este repositório.

7.4.15 Exemplos de ORBs

Há várias arquiteturas possíveis para a implementação dos ORBs. Um ORB específico pode suportar várias opções de implementação e distintos protocolos de comunicação.

7.4.15.1 ORB de Cliente e de Implementação

Se existir um mecanismo de comunicação específico, um ORB pode ser implementado utilizando-se dele em rotinas residentes nos Clientes e nas Implementações de Objetos.

Os Stubs no Cliente podem usar um mecanismo de IPC do tipo de localização transparente ou acessar diretamente um serviço de localização para estabelecer a comunicação com as Implementações de Objetos. Este código gerado deve ser responsável por criar e manter um banco de dados apropriado para disponibilizar informações necessárias aos Clientes.

7.4.15.2 ORB Baseado em Servidores

Para centralizar o gerenciamento de um ORB, todos os Clientes e Implementações de Objetos podem se comunicar com um ou mais servidores cuja função é rotear as requisições dos Clientes para as respectivas implementações.

Um ORB pode ser um programa normal e o canal de comunicação pode ser um IPC específico suportado pelo sistema operacional.

7.4.15.3 ORB Baseado em Sistemas Operacionais

Para incrementar a segurança, a robustez e o desempenho, um ORB pode ser provido como um serviço básico do sistema operacional. Referências a Objetos poderiam ser persistentes, reduzindo o custo da autenticação de cada requisição.

Considerando-se ainda que o sistema operacional pode neste caso conhecer previamente a localização e a estrutura dos Clientes e das Implementações de Objetos, seria possível implementar vários tipos de otimizações.

7.4.15.4 ORB Baseado em Bibliotecas

Para objetos simples cujas implementações podem ser compartilhadas, as Implementações de Objetos podem ser reunidas em uma biblioteca.

Neste caso, os Stubs são os próprios métodos. Este procedimento assume que é possível para o programa de um Cliente acessar os dados dos objetos que se encontram na biblioteca e que as Implementações dos Objetos confiam que os Clientes não inutilizaremos estes dados.
7.5 Estrutura de um Cliente

Um Cliente de um objeto tem uma Referência a Objeto que o identifica. Uma Referência a Objeto é, portanto, uma identificação que pode ser invocada ou passada como parâmetro para ser invocada por um outro objeto.

A invocação a um objeto envolve especificar o objeto a ser invocado, a operação a ser executada e os parâmetros a serem enviados à operação e a serem recebidos ao seu término.

O ORB gerencia a transferência do controle e dos dados para a Implementação dos Objetos e de volta para o Cliente. Quando o ORB por algum motivo não é capaz de completar a invocação, uma resposta do tipo exceção é gerada. Normalmente, o Cliente chama uma rotina no seu código que executa a invocação e que retorna quando a operação estiver completa.

Os Clientes acessam os Stubs como se fossem rotinas de bibliotecas nos seus próprios programas. Desta forma, o programa do Cliente vê as rotinas requisitáveis normalmente como se fossem rotinas comuns da linguagem de programação.

Todas as implementações provêem um tipo de dado específico da linguagem de programação para representar as Referências a Objetos, como está simbolizado na figura a seguir. O Cliente passa a Referência de Objeto para as rotinas do Stub a fim de inicializar a invocação, enquanto que os Stubs têm acesso à representação da Referência a Objeto e interagem com o ORB para executar a invocação.

![Diagrama do Cliente e ORB](image)

Figura 15: Estrutura Típica de um Cliente

Pode haver ainda um conjunto de bibliotecas disponível para executar invocações a objetos quando estes forem definidos em tempo de compilação. Neste caso, o programa do Cliente provê a informação adicional necessária para nomear o tipo do objeto e o método a ser utilizado e executa uma sequência de chamadas para especificar os parâmetros requeridos e em sequida inicializar a invocação.

Clientes normalmente obtêm as Referências a Objetos recebendo-as como parâmetros de saída de invocações a outros objetos dos quais eles têm referências. Quando um Cliente também é uma Implementação de Objeto, ele recebe as Referências a Objetos como parâmetros de entrada nas invocações dos objetos que ele próprio implementa.

Uma Referência a Objeto também pode ser convertida, por exemplo, em uma cadeia de caracteres, a qual pode ser armazenada e posteriormente recuperada para ser então utilizada pelo ORB que a gerou.
7.6 Estrutura de uma Implementação de Objetos

Uma Implementação de Objeto fornece o estado atual e o comportamento do objeto, podendo ser estruturado de várias formas distintas.

Além da própria definição dos métodos para as suas operações, uma implementação define também os procedimentos para a ativação e desativação dos objetos e pode ainda utilizar outros objetos para implementar persistência, controle de acesso, além de outros métodos específicos.

A Implementação do Objeto interage com o ORB para estabelecer sua identificação, para criar no vos objetos e para obter serviços específicos do ORB. Basicamente, este processo é feito através de um Adaptador de Objeto, como pode ser visualizado na figura a seguir, o qual provê uma interface aos serviços do ORB conveniente para uma implementação específica do objeto.

**Figura 16: Estrutura Típica de um Implementação de Objetos**

Em função da possibilidade de haver uma grande variedade de Implementações de Objetos, torna-se muito difícil caracterizar de forma geral como uma implementação deve ser estruturada.

Quando ocorre uma invocação a um objeto, o núcleo de um ORB, o Adaptador de Objeto e o Skeleton são responsáveis por garantir que a chamada seja feita à implementação apropriada do método do objeto. Um parâmetro do método sendo chamado especifica o objeto a ser invocado, o que permite ao método localizar dados adicionais necessários à Implementação do Objeto. Parâmetros adicionais são fornecidos de acordo com a definição do Skeleton.

Quando a execução do método tiver sido completada, ele retorna com os parâmetros de saída, ou com uma exceção, para serem transmitidos de volta ao Cliente.

Quando um novo objeto é criado, o ORB pode ser notificado de forma que ele passe a conhecer como encontrar a implementação deste novo objeto. Normalmente, a Implementação de Objeto também registra-se como uma nova implementação de uma interface em particular, além de especificar como inicializar a implementação caso ela já não estiver em execução.

A maioria das Implementações de Objetos provê o seu comportamento através do uso de facilidades adicionais ao ORB e ao Adaptador de Objeto. Este procedimento possibilita que diferentes Implementações de Objetos usem um mesmo serviço.
7.7 Estrutura de um Adaptador de Objetos

Um Adaptador de Objetos é a base para a Implementação dos Objetos pois permite acessar os serviços disponibilizados por um ORB, por exemplo, a geração das Referência a Objetos. O Adaptador de Objetos exporta uma interface pública da Implementação do Objeto e uma interface privada exclusiva para o Skeleton.

Os Adaptadores de Objetos são responsáveis pelas seguintes funcionalidades:

- Geração e interpretação das Referências a Objetos;
- Invocação dos métodos;
- Segurança das interações;
- Ativação e desativação dos objetos e das suas implementações;
- Mapeamento entre as Referências a Objetos e as Implementações de Objetos;
- Registro das Implementações de Objetos.

Estas funcionalidades são executadas através do núcleo do ORB e, opcionalmente, através de outros componentes que vierem a ser necessários.

Normalmente, um Adaptador de Objeto mantém seu próprio estado corrente para que esteja disponível às suas tarefas. É possível que um Adaptador de Objetos específico delegue uma ou mais de suas responsabilidades para o núcleo do ORB com o qual ele foi desenvolvido.

Na figura a seguir, o Adaptador de Objetos está implicitamente envolvido na chamada dos métodos, embora a interface direta responsável por isso seja normalmente através dos Skeletons. Por exemplo, o Adaptador de Objetos pode estar envolvido com o processo de ativação das Implementações de Objetos ou de autenticação das requisições de Clientes.

![Figura 17: Estrutura Típica de um Adaptador de Objetos](image)

Um Adaptador de Objetos define a maior parte dos serviços de um ORB que as Implementações de Objetos precisam. Diferentes ORBs podem prover diferentes níveis de serviços e diferentes ambientes de operação podem implicitamente prover algumas propriedades adicionais.

Por exemplo, é comum que Implementações de Objetos queiram armazenar alguns valores nas respectivas Referências a Objetos para facilitar a sua identificação durante a invocação. Se o Adaptador de Objetos permitir que a implementação especifique esses valores já durante a criação do objeto, ele pode ser capaz de armazená-los na Referência a Objetos daqueles ORBs que assim o permitirem. Se o núcleo do ORB não implementa esta funcionalidade, o Adaptador de Objetos pode armazenar estes valores localmente através de um serviço próprio de armazenamento e provê-los à Implementação dos Objetos durante a sua invocação.
Através dos Adaptadores de Objetos é possível que uma Implementação de Objetos tenha acesso a um serviço sempre que ele estiver implementado no núcleo do ORB. Se o núcleo do ORB provê o serviço, o Adaptador de Objeto simplesmente oferece uma interface para ele; por outro lado, se o serviço não for suportado, o Adaptador de Objetos deve implementá-lo localmente.

Todas as instâncias de um Adaptador de Objetos específico deve disponibilizar a mesma interface e os mesmos serviços para todos os ORBs para os quais eles foram implementados.

Não é necessário que todos os Adaptadores de Objetos fornecem a mesma interface ou as mesmas funcionalidades.

Algumas Implementações de Objetos possuem requisitos especiais. Por exemplo, um sistema de banco de dados orientado a objetos pode requisitar o registro implícito para milhares de objetos sem que para isso seja necessário chamadas individuais para o Adaptador de Objetos.

Ao se utilizar a interface de um Adaptador de Objetos para acessar as Implementações dos Objetos, é possível obter vantagens quando se utilizar particularidades do núcleo de um ORB e assim obter um acesso mais efetivo ao ORB.

### 7.7.1 Exemplos de Adaptadores de Objetos

Apesar de poder haver uma grande variedade de Adaptadores de Objetos, já que as Implementações de Objetos dependem diretamente da definição de suas interfaces, é desejável que na prática haja apenas poucos.

A grande maioria dos Adaptadores de Objetos são desenvolvidos para abranger o maior número de Implementações de Objetos possível, por isso, somente quando uma implementação necessitar de um serviço ou uma interface significativamente diferente, pode-se considerar a possibilidade de se desenvolver um novo Adaptador de Objetos.

A seguir descrevem-se três Adaptadores de Objetos que são os mais comuns para a grande maioria das Implementações de Objetos.

#### 7.7.1.1 BOA

Esta especificação define um Adaptador de Objeto que pode ser utilizado pela maioria das Implementações convencionais de Objetos.

Neste Adaptador de Objetos, as Implementações de Objetos são normalmente programas separados, permitindo haver programas que sejam inicializados para cada objeto ou, até mesmo, para cada método. Ele fornece uma pequena capacidade de armazenamento persistente para cada objeto, o qual pode ser utilizado como o nome ou o identificador para outro sistema de armazenamento, para listas de controle de acesso, ou para outras propriedades dos objetos.

Quando uma Implementação de Objeto não estiver ativa ao ser invocada, o BOA é responsável por inicializá-la.

#### 7.7.1.2 POA

A utilização do BOA tem mostrado algumas limitações e ambiguidades na especificação, o que tem conduzido a implementações distintas implicando em uma fraca portabilidade entre aplicações desenvolvidas para diferentes implementações de ORBs.

Em função disso, a OMG incluiu na nova especificação do CORBA o POA, uma extensão do BOA que incrementa o suporte à portabilidade entre ORBs implementados por diferentes desenvolvedores.
Outra característica importante deste novo Adaptador de Objetos é a habilidade de mapear várias Referências a Objetos em uma única Implementação de Objeto, dispensando desta forma a necessidade de se criar e se manter Implementações de Objetos separadas para cada um mesmo objeto utilizado.

O POA também permite ativar somente aqueles objetos de uma Implementação de Objetos que estiverem realmente envolvidos em um determinado processamento de uma requisição, adaptando assim os recursos locais de implementação às necessidades reais do sistema.

7.7.1.3 LOA

Este Adaptador de Objetos é basicamente utilizado para Implementações de Objetos agrupados em bibliotecas. Ele acessa o armazenamento persistente através de arquivos e não suporta a ativação de objetos ou a sua autenticação, já que os objetos são supostos de serem encontrados localmente nos programas dos Clientes.

7.7.1.4 OODA

Este Adaptador de Objetos utiliza uma conexão a um banco de dados orientado a objetos para fornecer acesso aos objetos armazenado neles.

Como o OODB provê os métodos e o armazenamento persistente, os objetos são registrados implicitamente e não há a necessidade de manter o estado corrente dos objetos.

7.8 Integração com Outros Sistemas de Objetos

O CORBA está especificado para permitir a interoperabilidade com vários sistemas de objetos, de acordo com o que está representado na figura a seguir. Em função de já existirem alguns sistemas de objetos, um requisito comum tem sido que os objetos destes sistemas possam ser acessados através de um ORB. Para os sistemas de objetos que já são ORBs, eles devem permitir a conectividade com outros ORBs através de mecanismos específicos para este propósito.

![Figura 18: Formas de Integração com Outros Sistemas de Objetos](image-url)

Para os sistemas de objetos que simplesmente desejam mapear seus objetos em um ORB a receber invocações através dele, basta transformar estes sistemas de objetos em Implementações de Objetos. Neste caso, o sistema de objetos deve registrar os objetos no ORB e saber lidar com as requisições que receber dele.
Em alguns casos, é inviável que outro sistemas de objetos venha a se tornar uma Implementação de Objetos de um BOA ou de um POA. Um Adaptador de Objetos deve ser desenvolvido visando um ORB previamente especificado e deve preferencialmente ser invocado através do próprio ORB.

Um outro sistema de objetos pode desejar criar objetos sem previamente consultar o ORB e pode ter a expectativa de que a maioria das invocações ocorram de dentro dele mesmo ao invés de ocorrer através de um ORB.

7.9 CORBA services

Os componentes especificados pelo CORBA services padronizam basicamente o gerenciamento do ciclo de vida dos objetos.

Provêm-se interfaces de criação de objetos, de controle de acesso de objetos, de recuperação de objetos previamente alocados e de controle do relacionamento entre estilos de classes. Desta forma, CORBA services garante a consistência das aplicações e auxilia no aumento da produtividade do processo de programação.

7.9.1 Serviço de Nomeação

O Serviço de Nomeação fornece a habilidade de ligar um nome a um objeto relativamente a um contexto de nomeação. O contexto de nomeação é um objeto que contém um conjunto de relações nas quais cada nome é único.

Resolver um nome é determinar qual o objeto que é associado com um nome em um determinado contexto.

Através do uso de um modelo genérico de nomeação, a implementação de um Serviço de Nomeação pode ser específica de uma determinada aplicação ou pode ser baseada em sistemas de nomeação fornecidos por sistemas operacionais específicos.

7.9.2 Serviço de Eventos

O Serviço de Eventos fornece capacidades básicas para o gerenciamento, o controle e a configuração de eventos, que podem ser configuradas de acordo com os requisitos específicos de uma determinada aplicação.

Este serviço foi especificado para ser escalonável e para suportar ambientes distribuídos, não havendo necessidade de estar em um servidor centralizado ou de depender de nenhum serviço global de eventos.

Há dois tipos de disponibilização de eventos: o tipo *push* e o tipo *pull*. O funcionamento destes tipos de suporte a eventos permite que os consumidores possam requisitar eventos ou que sejam notificados da presença de eventos, dependendo das características requeridas pela implementação de uma dada implementação. É possível, ainda, haver múltiplos consumidores de eventos e múltiplos geradores de eventos.

Geradores de eventos não necessitam conhecer os seus consumidores, assim como consumidores de eventos não necessitam obrigatoriamente conhecer os seus geradores.

7.9.3 Serviço de Ciclo de Vida

O Serviço de Ciclo de Vida define convenções de criação, remoção, cópia e mudança dos objetos. Como os ambientes baseados em CORBA suportam a distribuição de objetos nativamente, este
serviço é responsável por permitir que os Clientes executem operações de ciclo de vida em objetos em diferentes localidades.

O modelo de um Cliente para a criação de objetos é definido em termos de Fábrica de objetos. Uma Fábrica é um objeto, definido através de uma interface IDL, que é capaz de criar outros objetos.

7.9.4 Serviço de Persistência de Objetos

O Serviço de Persistência de Objetos fornece um conjunto de interfaces especificando mecanismos usados para o armazenamento persistente dos estados de objetos.

Uma importante característica deste serviço é que ele é aberto, o que implica que pode haver vários diferentes Clientes e Implementações de Objetos sendo capazes de operar conjuntamente.

7.9.5 Serviço de Transação

O Serviço de Transação suporta vários modelos de transação, interoperabilidade entre diferentes modelos de programação (procedimental e orientado a objetos), interoperabilidade entre redes (comunicação entre sistemas diferentes).

Este serviço permite tanto a propagação dos mecanismos de transação implícita ( coordenada pelo sistema) quanto a explícita (coordinada pela aplicação).

7.9.6 Serviço de Controle de Concorrência

O Serviço de Controle de Concorrência permite que vários Clientes coordenem seus acessos a recursos compartilhados.

Um acesso concorrente a um recurso é controlado através de chaves. Cada chave é associada a um único recurso e a um único Cliente. O controle do acesso é conseguido evitando que mais de um Cliente com diferentes chaves de acesso usem o mesmo recurso quando suas atividades forem conflitantes.

7.9.7 Serviço de Relacionamento

O Serviço de Relacionamento permite a representação explícita de entidades e relacionamentos, incluindo a representação de restrições.

Este serviço permite que, através do uso de grafos de objetos relacionados, objetos sejam percorridos sem serem ativados.

7.9.8 Serviço de Externalização

O Serviço de Externalização define protocolos e convenções para externalizar e internalizar objetos. Externalizar um objeto significa armazenar o estado de um objeto em um buffer acessível e internalizar um objeto significa recuperar um objeto (novo ou já existente) a partir de um buffer contendo um estado de objeto.

Este serviço está relacionado com o Serviço de Relacionamento e com o Serviço de Ciclo de Vida por definir protocolos de externalização de objetos.
7.9.9  Serviço de Comando

O Serviço de Comando permite que usuários e objetos invoquem um comando sobre um conjunto de objetos, comando este que é composto por uma especificação de valores de atributos e de operações.

Este serviço permite indexação e mapeia os mecanismos de comandos utilizados pelos sistemas de banco de dados.

7.9.10 Serviço de Licença

O Serviço de Licença provê um mecanismo para os procedimentos serem capazes de controlar o uso de sua propriedade intelectual.

Produtores de software podem implementar este serviço de acordo com suas necessidades e de seus clientes, já que o serviço não impõe nenhuma política de licenciamento.

7.9.11 Serviço de Propriedade

O Serviço de Propriedade provê a habilidade de dinamicamente associar nomes e valores com objetos fora de um domínio.

O serviço define operações para criar e manipular conjuntos de nomes e valores, provendo acesso e controle dos Clientes em relação as suas restrições e propriedades.

7.9.12 Serviço de Temporização

O Serviço de Temporização permite ao usuário obter informações de data e hora corrente de uma fonte centralizada.

Este serviço define tanto a gerência de informações de tempo e de intervalo de tempo quanto a geração de informação de tempo para a emissão de eventos.

7.9.13 Serviço de Segurança

O Serviço de Segurança define várias funcionalidades relativas à segurança no acesso e na manipulação dos objetos, tais como:

- **Identificação e autenticação** de usuários;
- **Autorização e controle de acesso** de objetos e/ou atributos;
- **Auditoria** de segurança;
- **Segurança de comunicação** entre objetos;
- **Administração** das características inerentes aos procedimentos segurança.

7.9.14 Serviço de Negociação de Objetos

O Serviço de Negociação de Objetos provê a habilidade de registrar a disponibilização de um novo serviço passando como parâmetros informações sobre os serviços oferecidos.

Este serviço é normalmente restrito a um domínio específico mas pode também ser configurado para permitir a disponibilização de serviços entre vários domínios.
7.9.15 Serviço de Coleção de Objetos

O Serviço de Coleção de Objetos provê uma forma uniforme de criar e manipular genericamente os tipos de coleções mais comuns.

Exemplos de coleções são os conjuntos, as filas, as pilhas, as listas, as árvores e os grafos. Operações típicas que são suportadas por este serviço são: inserção de novos elementos, teste de presença, união, intercessão, cardinalidade, entre outras.

7.10 CORBAfacilities

CORBAfacilities é a última área da OMA a ser definida. Esta área supre o espaço entre a tecnologia definida pelo CORBA, os CORBAservices e as aplicações específicas.

CORBAfacilities incluem especificações alto nível de serviços específicos para mercados verticais. Esta área é especialmente apropriada para garantir a padronização e consequente interoperabilidade entre produtores de softwares independentes.

7.10.1 CORBAfacilities Horizontais

Os CORBAfacilities Horizontais incluem funções compartilhadas pela maioria dos sistemas, independente do conteúdo das suas aplicações. Foram identificados basicamente quatro domínios principais:

- **Interface de Usuários**: torna um sistema de informações acessível para seus usuários;
- **Gerência de Informações**: refere-se à modelagem, definição, armazenamento, recuperação, gerência e compartilhamento de informações;
- **Gerência de Sistemas**: refere-se ao gerenciamento de sistemas complexos de várias plataformas;
- **Gerência de Processos**: refere-se à automação de procedimentos, tanto de usuários quanto de sistemas.

7.10.2 CORBAfacilities de Mercados Verticais

Os CORBAfacilities de Mercados Verticais representam tecnologias que suportam vários segmentos de mercado. A medida que grupos de empresas com interesses em comum desenvolvem especificações para novos segmentos, a OMG é responsável por garantir a sua integração na sua arquitetura. Alguns segmentos que já contém especificações próprias são:

- **Imagem**: suporta a interoperabilidade entre objetos de imagem, informações relativas a imagens e aplicações de manipulação de imagens;
- **Viárias de Informação**: suporta os serviços de informação de aplicações através de WANs;
- **Manufatura**: suporta a interoperabilidade entre objetos de manufatura;
- **Simulação Distribuída**: suporta a interação de vários objetos de simulação em ambientes virtuais;
- **Exploração e Produção de Petróleo**: suporta a interoperabilidade no mercado petrolífero;
- **Contas**: suporta as transações comerciais;
- **Desenvolvimento de Aplicações**: suporta a interoperabilidade entre objetos de desenvolvimento de aplicações;
- **Mapeamento**: suporta a interoperabilidade entre objetos de mapeamento.

7.11 Interfaces de Domínios

Interfaces de Domínios representam áreas verticais que provêm funcionalidades de interesse específico de usuários finais em domínios de aplicações particulares.
Interfaces de Domínio podem combinar alguns CORBA facilities e CORBA services mas são basicamente especificados para executar operações para usuários dentro de um mercado vertical, ou indústria, específico.
8. Implementação de TMN através de CORBA

Em resposta aos requisitos do mercado, os esforços para integrar as funcionalidades da infraestrutura do CORBA no ambiente TMN têm sido dirigidos tanto por consórcios industriais e por organismos internacionais de padronização quanto também por algumas empresas.

O CORBA, um sistema orientado a objetos com APIs bem definidas e padronizadas, simplifica a implementação de aplicações distribuídas e, combinado com o CMIP, um protocolo de transferência de informações de gerenciamento TMN, permite que as empresas de desenvolvimento de aplicações de gerência de redes de telecomunicações desenvolvam rapidamente sistemas robustos baseados em normas rígidas de modelagem de informação de acordo com os conceitos definidos pela ITU-T e amplamente aceitos pelo mercado.

Mas esta integração de CORBA em um ambiente TMN tem trazido alguns problemas relativos à interoperabilidade entre protocolos e aplicações de gerenciamento de redes de telecomunicações heterogêneas.

8.1 Considerações Tecnológicas Relativas à Integração TMN e CORBA

8.1.1 Comparação entre as Características do CORBA, do TMN e do IETF SNMP

Os padrões para o gerenciamento de sistemas e de redes têm sido especificados por vários organismos de normatização, baseados basicamente em diferentes domínios característicos de gerência ([15]). Por exemplo, o ITU-T está encarregado dos serviços e das redes de telecomunicações através do padrão TMN, enquanto que a ISO está responsável pelo gerenciamento do OSI e a IETF pelo gerenciamento da Internet através do SNMP.

As principais técnicas de gerenciamento usadas para o gerência de redes de comunicação são o SNMP proposto pelo IETF e o CMIP proposto pela ISO e pelo ITU-T. O CMIP é adotado como o protocolo de gerenciamento padrão para a gerência de redes de telecomunicações de acordo com as recomendações do TMN da ITU-T.

Tanto o CMIP quanto o SNMP têm características particulares em termos de funcionalidade e de desempenho, como apresentado na tabela a seguir:

<table>
<thead>
<tr>
<th>Característica</th>
<th>IETF SNMP</th>
<th>TMN CMIP</th>
<th>OMG CORBA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linguagem de Modelagem de Informação</td>
<td>SMI</td>
<td>GDMO</td>
<td>IDL</td>
</tr>
<tr>
<td>Primitivas de Gerenciamento</td>
<td>Get</td>
<td>M-GET</td>
<td>Request</td>
</tr>
<tr>
<td></td>
<td>GetNext</td>
<td>M-SET</td>
<td>Response</td>
</tr>
<tr>
<td></td>
<td>Set</td>
<td>M-CANCEL-GET</td>
<td>CancelRequest</td>
</tr>
<tr>
<td></td>
<td>Trap</td>
<td>M-CREATE</td>
<td>LocateReply</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M-DELETE</td>
<td>CloseConnection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M-EVENT-REPORT</td>
<td>MessageError</td>
</tr>
<tr>
<td>Notificação de Eventos</td>
<td>Trap</td>
<td>M-EVENT-REPORT</td>
<td>Serviço de Eventos</td>
</tr>
<tr>
<td>Arquitetura de Gerenciamento</td>
<td>Gerente/Agente/Gerente</td>
<td>Gerente/Agente/Gerente</td>
<td>Comunicação entre Objetos Distribuídos</td>
</tr>
<tr>
<td>Funções e Serviços de Gerenciamento</td>
<td>Não há</td>
<td>Funções de Gerenciamento TMN</td>
<td>CORBA services</td>
</tr>
<tr>
<td>Extensibilidade</td>
<td>Fraca</td>
<td>Fraca</td>
<td>Robusta</td>
</tr>
<tr>
<td>Respostas Encadeadas Múltiplas</td>
<td>Não</td>
<td>Sim</td>
<td>Não</td>
</tr>
<tr>
<td>Escopo e Filtragem</td>
<td>Não</td>
<td>Sim</td>
<td>Não</td>
</tr>
<tr>
<td>Transparência de Localização</td>
<td>Não</td>
<td>Não</td>
<td>Sim</td>
</tr>
</tbody>
</table>

O CMIP tem uma grande capacidade de recuperação de informações de gerenciamento através da utilização das funcionalidades de escopo, filtragem e respostas encadeadas múltiplas, assim como
uma poderosa capacidade de modelagem destas informações de gerência através da utilização das linguagens de modelagem GDMO e ASN.1. Por outro lado, o CMIP é relativamente lento em relação ao seu desempenho e não é fácil de ser instalado e utilizado. E ainda, as plataformas de desenvolvimento de aplicações TMN são normalmente complexas e caras.

Já o SNMP é relativamente fácil de ser utilizado em função da simplicidade dos seus modelos de informação e de comunicação. Por isso o SNMP tem sido amplamente utilizado no desenvolvimento de aplicações de gerência de diversas redes de computadores, principalmente LAN, FDDI e redes privadas de ATM. Mas o SNMP não é indicado para o gerenciamento de redes muito grandes e complexas porque a sua arquitetura é baseada em mecanismos de pooling. Além disso, os seus mecanismos de segurança são fracos e não há nenhuma especificação para aplicações de gerenciamento padrões, o que implica em problemas de interoperabilidade em redes heterogêneas de larga escala.

A nova maneira que tem permitido de uma forma eficiente gerenciar não somente redes mas também outros sistemas distribuídos tem sido proposta com sucesso pela OMG através da utilização da infra-estrutura do CORBA.

Os benefícios diretos obtidos pela utilização do CORBA são que seus conceitos já estão bem estabelecidos e amplamente adotados. A modelagem através da interface IDL do CORBA é relativamente fácil de ser definida e implementada e ainda pode ser mapeada para diferentes linguagens de programação, permitindo, desta forma, a interconectividade entre aplicações desenvolvidas com diferentes linguagens de programação, entre diferentes sistemas operacionais, entre diferentes sistemas de banco de dados, ou ainda, entre diferentes tipos de redes de comunicação.

Com a infra-estrutura de distribuição do CORBA, mesmo sistemas de computação simples são capazes de serem os servidores das aplicações complexas, sendo capazes de manter muitos objetos e habilitando a comunicação direta entre eles e entre os CORBA services requeridos. Esta estrutura permite tem permitido acomodar o gerenciamento integrado de redes de comunicação independentemente do seu tamanho.

Por um lado, o CORBA suporta a transparência de localização e a habilidade de integração de serviços e informações de gerência. Mas por outro lado, o CMIP utilizado pelo TMN tem uma poderosa capacidade de acesso a bases de informações de gerência em função de características suportadas como escopo, filtragem e respostas encadeadas. O CMIP também provê serviços de notificação de eventos mais bem elaborados através do uso de EFDs e de funções de controle de registros.

A integração destes ambientes de desenvolvimento de sistemas de gerenciamento de redes deve extrair as características positivas de cada uma destas arquiteturas.

8.1.2 Questões Relativas à Integração de CORBA e TMN

A integração de ambientes de gerenciamento distintos é um processo relativamente difícil já que a interoperabilidade destes sistemas de gerência depende das opções escolhidas na especificação da interface de comunicação e da especificação da definição dos objetos gerenciáveis adotada.

O CORBA tem sido proposto para garantir a interoperabilidade dos domínios computacionais enquanto que a arquitetura de gerenciamento OSI do TMN tem sido a opção proposta para a comunicação das informações de gerência dos ambientes de gerenciamento de redes.

Para possibilitar a integração destas arquiteturas em um sistema de gerenciamento integrado, deve haver um relacionamento entre as interfaces externas e internas dos sistemas, ou seja, objetos gerenciáveis definidos através de GDMO/ASN.1 devem ser mapeados de forma que possam ser entendidos pelo IDL do CORBA.

A seguir estão listados as principais questões relativas à integração da arquitetura do CORBA com o ambiente de gerenciamento TMN.
8.1.2.1 Resolução de Nomes e de Endereços dos Sistemas e Objetos Gerenciáveis

Uma das grandes vantagens que o CORBA tem é a característica de transparência de localização e a capacidade de distribuição dos objetos. O ORB é responsável por resolver o nome dos objetos e automaticamente encontra o endereço da localização da implementação destes objetos.

Por outro lado, a localização de um objeto gerenciável TMN é identificada de forma não transparente através de vários passos e operações. Um gerente TMN primeiro deve encontrar a localização do agente requerido através do serviço de diretórios ITU-T X.500 ou através do uso de listas de endereços OSI previamente conhecidos. A partir de então, o gerente TMN estabelece uma associação OSI com o agente e, finalmente, é capaz de enviar o comando CMIP desejado.

A integração da arquitetura do CORBA com o TMN deve ser capaz de resolver as dificuldades inerentes a estes mecanismos de resolução de endereçamento de objetos gerenciáveis.

Outra questão relativa a resolução de nomes é o mapeamento entre a nomeação dos objetos TMN (árvore de nomeação) e as Referências de Objetos do CORBA. O Serviço de Nomeação da OMG (CORBA services) não é capaz de suportar univocamente as convenções de nomeação utilizadas pela gerência TMN.

8.1.2.2 Integração de Diferentes Modelos de Informação de Gerência

Tanto CORBA quanto TMN baseiam-se em modelos de informação orientados a objetos. Na prática, a maior parte das sintaxes dos atributos definidos através do GDMO/ASN.1 para o gerenciamento TMN são facilmente mapeadas para sintaxes correspondentes do IDL do CORBA.

Porém, no modelo de informações utilizado pelo TMN, há alguns mecanismos de abstração para a definição de relacionamento entre objetos gerenciáveis, tais como relacionamento de contenimento e alomorfismo, que o CORBA não é capaz de representar diretamente. Esta discrepância precisa ser resolvida para a perfeita integração do CORBA no ambiente TMN.

8.1.2.3 Métodos Diferentes de Acesso e de Troca de Informações de Gerenciamento

O TMN provê métodos eficientes de acesso a vários objetos gerenciados em um relacionamento de contenimento através de técnicas de acesso tais como escopo, filtro e respostas múltiplas.

Por outro lado, o CORBA basicamente se baseia em um protocolo de acesso do tipo pergunta e resposta de forma que somente um único objeto por vez pode ser acessado em uma requisição de uma aplicação Cliente.

Estas discrepâncias em relação aos métodos e aos protocolos de acesso devem ser resolvidas sem que haja perda da semântica da informação de gerenciamento. Além disso, quando o número de objetos for muito grande e estiver muito espalhado em um sistema distribuído de larga escala, o CORBA pode ter alguns problemas em relação à escalabilidade e ao desempenho.

8.1.2.4 Suporte à Notificação de Eventos

Na gerência TMN, os EFDs são utilizados para coordenar o destino dos eventos e o controle do seu registro de forma a armazenar as informações de gerência necessárias e assim evitar uma possível perda de informações.

Por outro lado, o Serviço de Eventos do CORBA não provê serviços ponto a ponto com confirmação e também não armazena a informação de gerência, o que pode causar a perda de informações vitais para um sistema de gerenciamento.
A integração CORBA com TMN deve levar em conta estas características em relação à notificação de eventos e a sua consistência.

**8.1.2.5 Associação dos Serviços de Gerenciamento TMN com os CORBA services**

Várias funcionalidades e serviços de gerenciamento do TMN estão descritos nas recomendações M.3400 do ITU-T, apesar de as MIBs que implementam estes serviços não serem especificadas pelo ITU-T.

Há algumas diferenças na forma com que os gerentes CORBA gerenciam serviços baseados em implementações CORBA e a forma com que gerentes TMN gerenciam objetos gerenciados especificados em GDMO/ASN.1.

Desta forma, sistemas de gerência TMN baseados em CORBA devem ser capazes de interagir com os serviços especificados pela OMG (CORBA services). O relacionamento entre estes dois conjuntos de serviços deve ser bem identificado na integração CORBA com TMN.

**8.2 Integração da OMG**

A OMG está propondo uma arquitetura para sistemas TMN baseada em CORBA. Esta arquitetura inclui uma implementação alternativa dos conceitos de gerência de sistemas da ISO utilizando o paradigma do CORBA.

O objetivo desta forma de implementação da arquitetura é reutilizar o conhecimento adquirido ao longo dos anos durante o estabelecimento dos padrões de gerenciamento da ISO e do ITU-T, e assim, garantindo a compatibilidade com sistemas já desenvolvidos baseados nos padrões destes organismos de normatização.

**8.2.1 CORBA para os Sistemas de Gerenciamento**

A OMG está propondo que os sistemas de gerenciamento de redes de telecomunicações sejam centrados na arquitetura CORBA, suportando elementos de mediação para comunicação com sistemas baseados em outras arquiteturas.

Baseado nos atuais ambientes TMN, há uma série de tópicos em relação ao desenvolvimento destes sistemas operacionais e de sistemas de gerência baseados em outros padrões já existentes que estão servindo de referência nesta padronização.

No ambiente proposto pela OMG, uma MAF é apresentada com o mesmo significado das funções de gerência do TMN, já que também implementa um serviço de gerenciamento. A implementação das funções de gerenciamento não são padronizadas nem pela ITU-T no TMN e nem pela OMG, podendo ser consideradas como sendo casos específicos de aplicações.

A figura a seguir apresenta um ambiente de gerenciamento baseado em um sistema operacional proposto pela OMG.
Estes ambientes de desenvolvimento de sistemas operacionais baseados em CORBA incluem CORBAfacilities, que estão sendo definidos pelo grupo de sistemas de gerência da X/Open, e serviços de proxy, que são propostos pelo grupo do JIDM.

O grupo de telecomunicações da OMG está incumbido de definir estes CORBAfacilities que permitirão o desenvolvimento de serviços requeridos pelos ambientes TMN.

8.2.2 CORBA para os Sistemas Gerenciados

Os sistemas de gerência propostos pela OMG partem do pressuposto de que cada objeto CORBA representa recursos de sistemas computacionais. E como a tradução entre GDMO e IDL está definida pelo grupo JIDM, o CORBA tem sido o candidato natural para implementar a nova geração das interfaces do TMN.

Cada classe de Objeto Gerenciável é definida por uma interface IDL. É necessário ainda definir uma classe comum para todos os sistemas de gerenciamento baseados em CORBA a qual contivesse toda a informação básica sobre a classe do Objeto Gerenciável assim como a identificação de suas instâncias. Todas as classes componentes do sistema de gerência devem ser uma especialização da interface desta classe básica.

O sistema de gerenciamento pode usar de forma eficiente ambos tipos de interfaces do CORBA (estática ou dinâmica) para chamar os serviços requeridos pelos Objetos Gerenciados.

8.3 Interação e Especificação do JIDM

O grupo JIDM, apoiado pelos organismos TMF e X/Open, desenvolveu a especificação dos métodos de interação e de tradução para permitir a interoperabilidade das tecnologias SNMP, CMIP e CORBA.

A Especificação de Tradução descreve o mecanismo de tradução entre o GDMO/ASN.1 (linguagem de definição do modelo de informação utilizado pelo CMIP), o SMI/ASN.1 (linguagem de definição do modelo de informação utilizado pelo SNMP) e o IDL (linguagem de definição de interfaces utilizado pelo CORBA).

A Especificação de Interação descreve os mecanismos de conversão dinâmica entre os protocolos em um domínio e os protocolos em outro domínio, sem que cada um deles tenha ciência do tipo do domínio com que está se comunicando.
Estas especificações tem como objetivo permitir que objetos de um domínio de implementação sejam representados em um outro domínio e que as interações entre os objetos destes dois domínios ocorram de forma transparente. Por exemplo, um objeto em um domínio CORBA deve ser capaz de interagir com um objeto GDMO como se este último estivesse também em um domínio CORBA.

8.3.1 Especificação da Tradução

A Especificação de Tradução das linguagens de modelagem de informação provê os algoritmos de tradução das especificações dos Objetos Gerenciados usados pelos sistemas de gerenciamento baseados em CMIP (GDMO/ASN.1) e em SNMP (ASN.1) em especificações IDL usadas nos ambientes CORBA.

Um módulo ASN.1 é traduzido para um módulo IDL de acordo basicamente com as seguintes regras:

- Cada módulo ASN.1 é mapeado para um módulo IDL em arquivos separados;
- Os tipos importados do ASN.1 são mapeados para *typedef* no início do módulo IDL;
- Os identificadores do ASN.1 são mapeados para identificadores IDL, acrescidos de um número de diferenciação no caso de repetição de nomes;
- O mapeamento para IDL de tipos primitivos do ASN.1 é feito de acordo com a tabela apresentada em seguida;
- Tipos complexos do ASN.1 são mapeados para IDL como sendo uma composição de tipos primitivos.

<table>
<thead>
<tr>
<th>Tipo ASN.1</th>
<th>Tipo IDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOLEAN</td>
<td>typedef boolean ASN1_Boolean</td>
</tr>
<tr>
<td>INTEGER</td>
<td>typedef long ASN1_Integer</td>
</tr>
<tr>
<td>REAL</td>
<td>typedef double ASN1_Real</td>
</tr>
<tr>
<td>ENUMERATED</td>
<td>enum &lt;enumName&gt; { &lt;elem&gt;, ...};</td>
</tr>
<tr>
<td>BIT STRING</td>
<td>typedef sequence&lt;octet&gt; ASN1_BitString</td>
</tr>
<tr>
<td>OCTET STRING</td>
<td>typedef sequence&lt;octet&gt; ASN1_OctetString</td>
</tr>
<tr>
<td>OBJECT IDENTIFIER</td>
<td>typedef string ASN1_ObjectIdentifier</td>
</tr>
<tr>
<td>ANY DEFINED BY</td>
<td>ASN1_Any</td>
</tr>
<tr>
<td>EXTERNAL</td>
<td>ASN1_External</td>
</tr>
</tbody>
</table>

A especificação GDMO é traduzida para o IDL de acordo basicamente com as regras abaixo, representado pela figura apresentada em seguida:

- Um elemento GDMO é mapeado em várias interfaces: uma interface primária e interfaces específicas para suportar as ações e as notificações;
- Cada classe de Objeto Gerenciável especificado em GDMO é traduzido para uma interface IDL do CORBA, mantendo as informações de hierarquia;
- Cada atributo é traduzido para operações da interface primária de acordo com a lista de suas propriedades;
- Cada ação é traduzida para operações da interface primária;
- As notificações são traduzidas para operações a serem utilizadas pelos mecanismos de transmissão de eventos da OMG.
8.3.2 Especificação da Interação

A Especificação de Tradução das classes de Objetos Gerenciados define somente o mapeamento das linguagens e o comportamento relativo aos atributos e às notificações. A Especificação de Tradução também não define como e quando usar os CORBAservices da OMG.

Algumas características relativas ao comportamento dos Objetos Gerenciados são intencionalmente deixadas para a fase de interacção dos objetos, o que inclui o suporte para o uso dos pacotes condicionais, a distribuição dos eventos através do Serviço de Eventos da OMG e o repositório de informações IDL a partir do modelo de informações GDMO/ASN.1.

A Especificação de Interação é responsável por descrever os processos através dos quais as interações de um domínio são mapeadas em uma ou mais interações em outro domínio, eliminando desta forma a necessidade de um conhecimento prévio da arquitetura utilizada para o desenvolvimento de cada domínio de implementação que tiver que interagir.

8.4 ODMA

A ISO e a ITU-T iniciaram um projeto conjunto relativo à gerência de sistemas abertos com objetivo de criar um conceito genérico para uma arquitetura aberta de gerenciamento distribuído, denominada ODMA ([16] e [17]), e para um modelo baseado em ODP de gerenciamento de sistemas OSI.

O ODP é uma arquitetura integrada de suporte à distribuição, interoperabilidade e portabilidade de ambientes de processamento distribuídos. Esta arquitetura e o seu modelo de referência são especificados pela ISO/ITU-T e utilizados também por outros organismos de normatização, tais como a TINA-C ([18]).

A ODMA é uma extensão da Gerência de Sistemas da ISO. Ela descreve como uma arquitetura de gerenciamento pode ser reutilizada em um ambiente distribuído, especialmente em relação a sua utilização na integração com a arquitetura de distribuição CORBA da OMG. A figura a seguir apresenta os documentos relativos ao ODMA já definidos e as áreas em que ainda há o que ser especificado:
Figura 21: Documentos Relativos à ODMA

O ambiente da ODMA aplica as referências do ODP para o gerenciamento distribuído e define um conjunto de conceitos adicionais para algumas destas referências.

O modelo de referência do ODP é um padrão da ISO/ITU-T que provê um ambiente e uma estrutura de especificação para sistemas distribuídos e heterogêneos de larga escala. Ele define uma arquitetura com um conjunto de cinco referências para diferentes características do problema da distribuição de sistemas assim como um conjunto de funções e de mecanismos de transparência capazes de suportar esta distribuição. A ODMA é, portanto, uma arquitetura especializada deste modelo de referência do ODP para o gerenciamento distribuído de recursos, sistemas e aplicações distribuídos.

A figura a seguir apresenta um sistema de gerência baseado em CORBA que suporta vários protocolos de gerenciamento. Os Objetos Gerenciados definidos através de GDMO/ASN.1 a serem acessados através do CMIP podem ser implementados como objetos CORBA, e, desta forma, as implementações destes objetos gerenciados podem também ser acessados usando protocolos interoperáveis com o CORBA. Mecanismos de infra-estruturas proprietárias podem também interoperar com objetos CORBA de uma forma similar.
Figura 22: Sistema Aberto de Gerenciamento Baseado em CORBA

O suporte ao gerenciamento OSI para ODMA é uma extensão do gerenciamento de sistemas definido pela ISO, descrevendo como um SMA pode ser reutilizado em um ambiente distribuído. Além disso, através do uso de um ambiente ODMA tem-se a vantagem de que vários aspectos de um sistema de gerenciamento OSI podem ser encapsulados simplificando-se, desta forma, o sistema.

O objetivo do mapeamento da ODMA para CORBA é integrar o ambiente distribuído orientado a objetos com o ambiente baseado no paradigma agente/gerente do sistema de gerenciamento definido pela ISO/ITU-T, em especial, o TMN.
9. Conclusões

As empresas precisam continuar a operar incorporando as novas tecnologias que emergem a cada dia. Ao entender os conceitos, implementar e implantar aplicações de gerência e construir uma infraestrutura TMN, elas poderão certamente maximizar o potencial dos seus sistemas de gerenciamento e, consequentemente, dos seus equipamentos atuais.

O gerenciamento TMN baseia-se basicamente na definição de dois tipos de entidades: o Gerente (uma aplicação que implementa as funcionalidades de gerenciamento) e os Objetos Gerenciados (entidades de software que provêem uma interface padrão de representação de recursos de sistemas). Os Gerentes, através de um conjunto de serviços oferecidos pelo ambiente de gerenciamento, executam operações sobre os Objetos Gerenciados.

A implementação de TMN através de uma arquitetura CORBA tem sido proposta por vários organismos de normatização, tais como OMG, JIDM, ISO/ITU-T e TINA-C, em função da simplicidade e flexibilidade das aplicações baseadas nesta arquitetura em comparação com as arquiteturas mais específicas e complexas, como as baseadas em SNMP ou CMIP.

A arquitetura especificada para o desenvolvimento de aplicações de gerenciamento baseadas em SNMP ou CMIP provê métodos de acesso eficientes baseado no paradigma Gerente/Agente da ISO/ITU-T. Por outro lado, o CORBA provê um alto grau de flexibilidade na distribuição e implementação de Gerentes e de Objetos Gerenciados.

Este artigo apresentou como os principais organismos de normatização envolvidos na padronização de ambientes de gerenciamento distribuído de sistemas têm tratado a integração do ambiente TMN com a arquitetura CORBA, cada qual procurando utilizar da melhor maneira as vantagens oferecidas por cada uma destas arquiteturas.

Esta preocupação dos organismos internacionais de normatização reflete a importância que as empresas estão dando para esta integração de arquiteturas e a necessidade de se implementar os conceitos do TMN já absorvidos pelo mercado através do CORBA.
10. Referências Bibliográficas