



CARLOS EDUARDO BERTHOLDI

**ARQUITETURA BASEADA EM CORBA PARA GERENCIAMENTO
INTEGRADO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES**

Dissertação apresentada ao Curso de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná, Centro de Ciências Exatas e Tecnológicas, Departamento de Informática, como requisito parcial para a obtenção do grau de Mestre em Ciências.

Orientador: Prof. Dr. Edgard Jamhour

Dis
004
B5920-
1999

Curitiba

1999



PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ
Pró-Reitoria de Pós-Graduação, Pesquisa e Extensão

ATA DA SESSÃO PÚBLICA DE EXAME DE DISSERTAÇÃO DO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA APLICADA DA PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ.

Exame de dissertação nº 009

Aos 14 dias do mês de setembro de 1999, realizou-se a sessão pública de defesa de dissertação "ARQUITETURA BASEADA EM CORBA PARA GERENCIAMENTO INTEGRADO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES", apresentada por Carlos Eduardo Bertholdi, ano de ingresso 1996, para obtenção do título de Mestre em Ciências. A Banca Examinadora foi composta pelos seguintes professores:

MEMBROS DA BANCA	ASSINATURA
Presidente: Prof. Dr. Edgard Jamhour (PUCPR)	
Prof. Dr. Luiz Augusto de P. Lima Junior (PUCPR)	
Prof. Dr. Manoel Camillo de O. Penna Neto (PUCPR)	
Prof. Dr. Luiz Nacamura (CEFET)	
Prof. Dr. José Marcos Nogueira (UFMG)	

De acordo com as normas regimentais a Banca Examinadora deliberou sobre os conceitos a serem atribuídos e que foram os seguintes:

MEMBROS DA BANCA	CONCEITOS
Presidente: Prof. Dr. Edgard Jamhour (PUCPR)	A
Prof. Dr. Luiz Augusto de P. Lima Junior (PUCPR)	A
Prof. Dr. Manoel Camillo de O. Penna Neto (PUCPR)	A
Prof. Dr. Luiz Nacamura (CEFET)	A
Prof. Dr. José Marcos Nogueira (UFMG)	A
Conceito Final	A

Observações da Banca Examinadora

Profº Júlio Cesar Nievola
Coordenador do Programa de Pós-Graduação em Informática Aplicada-PUC-PR

Agradecimentos
possibilitaram

A minha
apoio durante o

Ao
trabalho, o

A todos
estudo e o

A
deste trabalho

À minha família,
Andréa, Juliana e Luiza

AGRADECIMENTOS

Agradeço primeiramente aos meus pais que em seus esforços possibilitaram a minha formação.

À minha família, em especial à minha esposa Andréa, pelo incentivo e apoio durante a realização de todo o curso.

Ao professor Edgard Jamhour, pela orientação na realização deste trabalho, e principalmente pelo companheirismo e amizade.

A todos os professores, pelos ensinamentos que transmitiram.

Agradecimentos ao Sr. Antônio Motta da Luz, pela oportunidade de estudo e aperfeiçoamento concedida, assim como pelo apoio prestado.

A todos os demais que de alguma forma contribuíram para a realização deste trabalho.

SUMÁRIO

1 INTRODUÇÃO	1
1.1 OBJETIVOS DO TRABALHO.....	1
1.2 IMPORTÂNCIA DO TRABALHO	1
1.3 LIMITAÇÃO DO TRABALHO	2
1.4 REVISÃO DE LITERATURA	2
1.5 ESTRUTURA DO TRABALHO	3
2 DESCRIÇÃO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES.....	5
2.1 INTRODUÇÃO.....	5
2.2 CARACTERÍSTICAS GERAIS DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES	5
2.2.1 Arquitetura distribuída	5
2.2.2 Equipamentos e sistemas heterogêneos.....	6
2.2.3 Interdependência.....	7
2.2.4 Interfaces proprietárias.....	8
2.3 ARQUITETURA FÍSICA DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES.....	9
2.3.1 Subsistemas do Grupo I.....	10
2.3.1.1 Subsistema de Corrente Contínua.....	10
2.3.1.2 Subsistema de energia em corrente alternada	11
2.3.1.3 No-Break	12
2.3.1.4 Subestações.....	12
2.3.1.5 Subsistema de iluminação	13
2.3.1.6 Subsistemas hidráulicos	14
2.3.2 Subsistemas do Grupo II	16
2.3.2.1 Elevadores	17
2.3.2.2 Controle de acesso.....	17
2.3.2.3 Detecção e alarme de incêndio	17
2.3.2.4 Condicionadores de Ar	18
2.3.3 Requisitos para gerenciamento informatizado da infra-estrutura de telecomunicações	19
2.3.4 Funcionalidades do sistema de gerenciamento de infra-estrutura.....	19
2.3.4.1 Gerenciamento de falhas	20
2.3.4.2 Gerenciamento de configuração.....	20
2.3.4.3 Gerenciamento de desempenho	20
2.3.4.4 Gerenciamento de segurança	21
2.3.5 Requisitos para arquiteturas de sistemas de gerenciamento	22
2.3.5.1 Sistema aberto	22
2.3.5.2 Sistema modular	22
2.3.5.3 Sistema escalonável.....	22
2.3.5.4 Sistema distribuído	23
2.3.5.5 Gerenciamento integrado	23
2.3.5.6 Interface amigável com o usuário	23
2.3.6 Requisitos oriundos de Práticas Telebrás	24
2.3.7 Níveis de gerenciamento	26
2.3.8 Sistemas Comerciais.....	29
3 ESTUDO DAS TECNOLOGIAS APLICADAS AO GERENCIAMENTO DE INFRA-ESTRUTURA.....	31

3.1 INTRODUÇÃO	31
3.2 O MODELO TMN- TELECOMMUNICATION MANAGEMENT NETWORK	31
3.2.1 Arquitetura orientada à objetos	33
3.2.2 Conceito Gerente/Agente	34
3.2.3 TMN e o gerenciamento da infra-estrutura	37
3.3 O MODELO SNMP – SIMPLE NETWORK MANAGEMENT PROTOCOL	38
3.3.1 SNMP e o gerenciamento da infra-estrutura	41
3.4 TECNOLOGIAS DE SISTEMAS DISTRIBUÍDOS	42
3.4.1 Sockets	44
3.4.2 Chamada de procedimentos remotos: RPC	44
3.4.3 RMI – Remote Method Invocation	46
3.4.4 D-COM	46
3.4.5 CORBA	46
 4 A ARQUITETURA CORBA E SERVIÇOS APLICÁVEIS AO GERENCIAMENTO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES.....	 49
4.1 INTRODUÇÃO	49
4.2 CONCEITOS DE CLIENTE, SERVIDOR E MIDDLEWARE.....	49
4.3 A ARQUITETURA CORBA – OMA	50
4.4 COMPONENTES E INTERFACES DA ARQUITETURA CORBA	54
Aplicação do cliente	55
4.4.2 Evocação estilo stub	56
4.4.3 Evocação dinâmica	57
4.4.4 Objeto de contexto	57
4.4.5 Repositórios de interface.....	58
4.4.6 Repositórios de implementação	60
4.4.7 Aplicação do Servidor.....	61
4.4.8 Basic Object Adapter - BOA	61
4.4.9 Esqueleto do servidor.....	63
4.5 CORBA E OS CONCEITOS DE ORIENTAÇÃO À OBJETOS.....	63
4.6 MODELO DE COMUNICAÇÃO CORBA	65
4.6.1 OMG IDL	67
4.7 SERVIÇOS DE OBJETOS CORBA APLICÁVEIS NO GERENCIAMENTO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES.	68
4.7.1 Serviço de nomes.....	68
4.7.2 Serviço de eventos.....	68
4.7.3 Serviço de persistência de objetos	70
4.7.4 Serviço de controle de concorrência	70
4.7.5 Serviço de tempo	71
4.7.6 Serviço de segurança.....	71
 5 ARQUITETURA PARA GERENCIAMENTO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES BASEADA EM CORBA.....	 73
5.1 INTRODUÇÃO	73
5.2 ARQUITETURA FÍSICA	73
5.3 ARQUITETURA FUNCIONAL.....	75
5.4 ARQUITETURA DE INFORMAÇÃO	80
5.4.1 Modelo geral para os subsistemas	87
5.4.2 Arquitetura de informação do gerenciador	90

5.4.3	Especialização do modelo geral	92
6	ESTUDO DE CASO.....	94
6.1	INTRODUÇÃO.....	94
6.2	DEFINIÇÃO DOS SUBSISTEMAS A SEREM GERENCIADOS.....	94
6.2.1	Casos de uso do sistema de gerenciamento	96
6.3	CONCEPÇÃO DA SIMULAÇÃO DO SISTEMA DE GERENCIAMENTO	97
6.3.1	Projeto das interfaces.....	99
6.3.2	Projeto do Sistema Gerenciador	100
6.3.3	Projeto do Sistema Gerenciado.....	100
6.4	IMPLEMENTAÇÃO DAS INTERFACES IDL – <i>INTERFACE DEFINITION LANGUAGE</i>	101
	Implementação do sistema	104
6.4.2	Limitações da simulação	105
6.5	RESULTADOS PRINCIPAIS DA SIMULAÇÃO.....	106
7	CONCLUSÃO.....	108
7.1	SUGESTÕES PARA TRABALHOS FUTUROS.....	110
ANEXO 1	– Outros Serviços CORBA	111
ANEXO 2	– Códigos fonte da simulação	115
ANEXO 3	– Glossário UML	124
REFERÊNCIAS BIBLIOGRÁFICAS		130

LISTA DE FIGURAS

Figura 1 - Visão geral da infra-estrutura de telecomunicações	7
Figura 2 - Elementos do subsistema de corrente contínua	10
Figura 3 - Elementos do subsistema de fornecimento de energia em corrente alternada	11
Figura 4 - Elementos gerenciáveis do No-Break	12
Figura 5 - Elementos da subestação de energia	13
Figura 6 - Elementos do subsistema de iluminação	14
Figura 7 - Elementos gerenciáveis do subsistema hidráulico	15
Figura 8 - Integração dos subsistemas do tipo 2 com o sistema de gerenciamento da infra-estrutura	16
Figura 9 - Arquitetura funcional do SAIE[37]	25
Figura 10 - Níveis de gerenciamento	27
Figura 11 - Modelo TMN e seus blocos funcionais	32
Figura 12 - Modelo Gerente - Agente	35
Figura 13 - Modelo de comunicação TMN	36
Figura 14 - Modelo de comunicação SNMP	39
Figura 15 - Integração de CORBA com TMN e SNMP	48
Figura 16 - Interação entre cliente e servidor por intermédio de middleware	49
Figura 17 - OMA - <i>Object Management Architecture</i>	50
Figura 18 - Seleção de implementação pelo ORB	52
Figura 19 - Componentes e interfaces CORBA	55
Figura 20 - Modelo de comunicação CORBA	66
Figura 21 - Canal de eventos estilo <i>push</i>	69
Figura 22 - Arquitetura física genérica	74
Figura 23 - Arquitetura para gerenciamento baseada em CORBA - lado do subsistema	76
Figura 24 - Arquitetura para gerenciamento baseada em CORBA - lado do gerenciador	78
Figura 25 - Relacionamento funcional cliente/servidor	79
Figura 26 - Diagrama de classe geral	81
Figura 27 - Funcionalidades de gerenciamento e parâmetros gerenciados	83
Figura 28 - Diagrama de classes para sistemas gerenciados	88
Figura 29 - Mecanismo para notificação de falhas	90
Figura 30 - Diagrama de classes do sistema gerenciador	91
Figura 31 - Modelo para acesso aos parâmetros de alarmes	91
Figura 32 - Diagrama de classes do subsistema de energia CC	92
Figura 33 - Subsistema local	95
Figura 34 - Subsistema remoto	95
Figura 35 - Diagrama de caso de uso simulado	96
Figura 36 - Visão geral do sistema simulado	98
Figura 37 - Interfaces IDL para a simulação	99
Figura 38 - Diagrama de classes do sistema cliente simulado	100
Figura 39 - Diagrama de classes do sistema servidor	101
Figura 40 - Diagrama das classes geradas a partir do compilador IDL	103
Figura 41 - Metodologia para implementação do sistema simulado	104

LISTA DE TABELAS

Tabela 1 – Parâmetros de identificação.....	84
Tabela 2 - Parâmetros de controle.....	84
Tabela 3 - Parâmetros de estado funcional.....	85
Tabela 4 - Parâmetros de medição.....	86
Tabela 5 - Parâmetros de alarme.....	87

LISTA DE SIGLAS

ANATEL	Agência Nacional de Telecomunicações
API	Application Programming Interface
BOA	Basic Object Adapter
CA	Corrente Alternada
CC	Corrente Contínua
CT	Controlador
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
CORBA	Common Object Request Broker Architecture
COSS	Common Object Service Specification
D-COM	Distributed COM
DCE-RPC	Distributed Computer Environment – RPC
DCF	Data Communication Function
GDMO	Guidelines for the Definition of Managed Objects
GMG	Grupo Motor Gerador
IDL	Interface Definition Language
ITU	International Telecommunication Union
JIDM	Joint Inter Domain Management
LAN	Local Area Network
M	Mediation function
MIB	Management Information Base
NBR	Normas Brasileiras
NEF	Network Element Function
OMA	Object Management Architecture
OMG	Object Management Group
OQL	Object Query Language
OSF	Operations System Function
PDU	Protocol Data Unit
PLC	Programador Lógico Controlável
RMI	Remote Method Invocation
RPC	Remote Procedure Call
SAIE	Sistema de Supervisão e Automação da Infra-Estrutura
SIS	Sistema Integrado de Supervisão
SMK	Base Compartilhada de Conhecimentos Gerenciais
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
TCP-IP	Transmission Control Protocol/ Internet Protocol
TELEMIG	Telecomunicações de Minas Gerais
TMN	Telecommunication Management Network
UDP	User Datagram Protocol
UML	Unified Modeling Language
UPS	Uninterrupted Power Supply
WAN	Wide Area Network
WLL	Wireless Local Loop
WSF	Work Station Function

RESUMO

O objetivo desta dissertação é a proposta de uma arquitetura baseada em CORBA – *Common Object Request Broker Architecture* para ser aplicada no gerenciamento da infra-estrutura de telecomunicações. Deve ser evidenciado que este trabalho não se refere ao gerenciamento de sistemas de telecomunicações em geral, mas sim a um campo muito menos explorado, correspondente a infra-estrutura de telecomunicações. A infra-estrutura de telecomunicações é composta por sistemas distribuídos e heterogêneos como: sistemas de controle de acesso, iluminação, demanda de energia, sistema de detecção e alarme de incêndio, sistemas hidráulicos, ar condicionado e ventilação mecânica, grupos geradores, fontes de corrente contínua, *no-breaks* e elevadores. Como pode ser facilmente identificado, estes sistemas não são utilizados exclusivamente para suportar equipamentos e serviços de telecomunicações. Mas, no contexto das telecomunicações eles devem ser gerenciados de forma muito especial, objetivando a redução dos custos de operação e manutenção, aumentando também confiabilidade das telecomunicações.

Dois fatos principais contribuem para a motivação desta pesquisa. Primeiramente, na área de telecomunicações existe uma grande demanda por sistemas informatizados capazes de gerenciar a ampla gama de plataformas e equipamentos de infra-estrutura como um sistema integrado[35]. Em segundo, devido a sua grande flexibilidade, as tecnologias de objetos distribuídos, mais especificamente CORBA, é uma forte candidata para ser adotada como uma tecnologia adicional no gerenciamento das telecomunicações.

São descritos os sistemas que compõe a infra-estrutura de telecomunicações, assim como seus requisitos de gerenciamento. Também são apresentadas algumas soluções importantes utilizadas atualmente no gerenciamento de telecomunicações, como TMN-Telecommunication Management Network, SNMP - Simple Network Management Protocol e algumas tecnologias de sistemas distribuídos. A importância desta abordagem é a compreensão das características dos equipamentos, requisitos de gerenciamento e soluções atuais.

A contribuição mais relevante deste trabalho é a proposta de uma arquitetura baseada em CORBA para o gerenciamento da infra-estrutura de telecomunicações. São definidas as arquiteturas física, funcional e de informação. A arquitetura de informação é modelada em termos de objetos gerenciáveis distribuídos, utilizando-se UML-Unified Modeling Language.

A dissertação valida a proposta através da implementação de um estudo de caso representativo, aplicando a linguagem JAVA e um broker comercial. As

interfaces são projetadas de acordo com a IDL -Interface Definition Language, definida pela OMG – Object Management Group. Uma simulação sobre uma rede de computadores mostra como os aspectos fundamentais de gerência de infra-estrutura são tratados pela arquitetura proposta.

Espera-se que este trabalho sirva de base para o desenvolvimento de sistemas de gerenciamento de infra-estrutura verdadeiramente abertos. As interfaces homem-máquina, assim como a integração entre CORBA e TMN, e entre CORBA e SNMP são sugeridas como trabalhos complementares.

ABSTRACT

The objective of this dissertation is the proposition of a CORBA (*Common Object Request Broker Architecture*) based architecture applied to the telecommunication's infrastructure management. It must be evidenced that this work does not refer to telecommunication systems in general, but to a much less explored field corresponding to the telecommunication's infrastructure. Telecommunication's infrastructure is composed by heterogeneous and distributed systems like: access control systems, enlightenment, energy demand, fire detection and alarming, hydraulic systems, air conditioning and mechanic ventilation, generators set, direct current systems, no-breaks and elevators. As easily identified, these systems are not used exclusively to support telecommunication's equipment and services. But, in the telecommunication's context, they must be managed in very special ways, aiming the reduction of operation and service costs, also increasing the telecommunication's reliability.

Two main facts contribute to the motivation of this research. First, in the telecommunication's field there is a great demand for computer systems capable to manage the wide range of infrastructure and equipment platforms as an integrated system[35]. Second, because of its great flexibility, distributed objects technology, more specifically CORBA, is a strong candidate to be adopted as an additional technology for the telecommunication's management.

The systems that compose the telecommunication's infrastructure are described, as well as their management requirements. It is also presented some important solutions used nowadays in the telecommunication's management, like TMN – Telecommunication's Management Network, SNMP – Simple Network Management Protocol and some distributed systems technologies. The importance of this approach is the understanding of the equipment characteristics, management requirements and current solution.

The most relevant contribution of this work is the proposal of CORBA based architecture for the telecommunication's infrastructure management. The physical, functional and information architectures are defined. The information architecture is modelled in terms of managed objects, using UML – Unified Modelling Language.

The dissertation validates the architecture by implementing a representative study case, using the JAVA language and a commercial broker. The interfaces are designed according to the IDL - Interface Definition Language, defined by the OMG – Object Management Group. A simulation over a computer network shows how the fundamental aspects of the infrastructure management are threatened by the proposed architecture.

It is expected that this work serve as base for the development of true opened infrastructure management systems. The man-machine interfaces, as well as the integration between CORBA and TMN or between CORBA and SNMP are suggested as complementary researches.

1 INTRODUÇÃO

1.1 OBJETIVOS DO TRABALHO

Neste trabalho objetiva-se definir e avaliar uma proposta de arquitetura de gerenciamento de infra-estrutura de telecomunicações baseada em CORBA. Para isso são inicialmente pesquisados os elementos que fazem parte da infra-estrutura, suas características, as exigências para um gerenciamento informatizado e as tecnologias de informática aplicadas atualmente.

Os objetivos do trabalho são:

- a) pesquisar os elementos que compõe a infra-estrutura de telecomunicações e seus parâmetros gerenciáveis
- b) identificar os requisitos importantes para o gerenciamento da infra-estrutura de telecomunicações a partir da revisão de normas específicas[37], publicações científicas e das soluções atualmente propostas.
- c) identificar características e serviços CORBA que possam ser aplicadas diretamente em sistemas de gerenciamento da infra-estrutura de telecomunicações.
- d) propor uma arquitetura apropriada baseada em CORBA para o gerenciamento da infra-estrutura de telecomunicações
- d) avaliar a arquitetura proposta através de um estudo de caso simulado em uma rede de computadores, utilizando objetos distribuídos.

1.2 IMPORTÂNCIA DO TRABALHO

Com a evolução dos sistemas de telecomunicações aumentou-se a expectativa dos usuários pela confiabilidade, segurança e disponibilidade dos serviços prestados. Isso trouxe reflexos significativos aos sistemas pertencentes à infra-estrutura de telecomunicações. Contar com uma infra-estrutura confiável e com boa relação custo benefício é fundamental para as operadoras de

telecomunicações, pois os equipamentos de telecomunicações e seus serviços se apoiam totalmente nesta infra-estrutura[24].

O gerenciamento informatizado da infra-estrutura objetiva justamente o aumento da confiabilidade e a redução de custos de operação e manutenção. Entretanto, a urgência por soluções levou à soluções e equipamentos heterogêneos, tanto em termos de avanço tecnológico quanto em diversidade de equipamentos. Isso dificulta a construção sistemas capazes de gerenciar a infra-estrutura de telecomunicações de forma integrada.

A pesquisa de novas tecnologias de informática aplicáveis aos sistemas de gerenciamento de infra-estrutura de telecomunicações busca, entre outros objetivos, novos caminhos que simplifiquem a implementação de tais sistemas. O fato de CORBA estar sendo apontada como tecnologia complementar para o gerenciamento de sistemas de telecomunicações[32], motivou o desenvolvimento deste trabalho.

1.3 LIMITAÇÃO DO TRABALHO

Não é finalidade deste trabalho abordar tópicos como interface de usuário, e a integração de CORBA com TMN – Telecommunication Management Network e SNMP – Simple Network Management Protocol. Estes são sugeridos como trabalhos futuros.

No estudo de caso não são implementados todos os subsistemas. Entretanto as funcionalidades de gerenciamento foram implementadas e testadas.

1.4 REVISÃO DE LITERATURA

Na revisão bibliográfica foram identificados poucos trabalhos relacionados com o tema abordado nesta dissertação. Os mais importantes são apresentados a seguir:

Motta e Guimarães[24], no artigo *Panorama da Infra-estrutura para Telecomunicações e Perspectivas da Supervisão e Automação sob o Ponto de Vista de Padrões e Protocolos*, citam que plataformas de software distribuídos, entre eles CORBA, irão influenciar a evolução dos sistemas de gerenciamento da infra-estrutura de telecomunicações. Afirmam ainda que o uso de arquiteturas distribuídas podem facilitar a implementação de tais sistemas e que espera-se que os equipamentos possuam interfaces mais simples para se comunicarem como os sistemas de gerência de equipamento.

Nogueira, Andrade e Menezes, no seu artigo *Integração de interfaces Q3 ao SIS: Estratégia e Prática*[27], descreveram o sistema SIS – Sistema Integrado de Supervisão como um sistema cujo maior objetivo é a integração à sua plataforma, de sistemas de supervisão e gerência com alto grau de heterogeneidade funcional e tecnológica. Neste trabalho descreve-se uma metodologia para integração à este sistema de uma interface Q3, definida nas especificações ITU. Os serviços de gerenciamento foram implementados utilizando-se RPC, sendo que as mudanças necessárias para a adoção do CMIP sobre TCP-IP demonstram ser uma dificuldade.

Outros trabalhos correlatos podem ser citados: *Monitoring and Control of Telecommunications Energy Systems*, apresentado por Mathiensen[21]; especificações gerais para sistemas de automação de infra-estrutura predial, de Pinto[34]; *Experiences of Energy Management Systems in Telecommunications*[45], de Wikström e Englung, onde são descritas as experiências do uso de SNMP no gerenciamento de sistemas de energia para telecomunicações.

1.5 ESTRUTURA DO TRABALHO

A infra-estrutura de telecomunicações é descrita em termos de subsistemas no capítulo 2. Neste capítulo procura-se enfatizar a heterogeneidade e a característica distribuída dos subsistemas que formam a infra-estrutura de telecomunicações, sendo identificados os principais parâmetros gerenciáveis e os requisitos para o gerenciamento integrado.

No capítulo 3 são apresentadas as tecnologias de gerenciamento atualmente aplicadas no gerenciamento da infra-estrutura e os seus modelos de comunicação. São explorados ainda os principais conceitos e mecanismos de comunicação utilizados por estas tecnologias, como o modelo gerente/ agente.

A arquitetura CORBA é apresentada no Capítulo 4. São identificados os serviços CORBA definidos no *COSS – Common Object Service Specification* possíveis de serem usados em sistemas de gerenciamento de infra-estrutura.

No Capítulo 5 propõe-se uma arquitetura genérica para gerenciamento de infra-estrutura de telecomunicações, sendo esta modelada em termos de objetos gerenciáveis. Para isso, os parâmetros de gerenciamento são classificados em termos de tabelas, possibilitando a análise dos subsistemas. Esta análise destina-se a construção de modelos de forma que a problemática do gerenciamento possa ser melhor compreendido. Na modelagem em objetos é utilizada a notação UML – Unified Modeling Language[29][12].

No Capítulo 6 é desenvolvido um estudo de caso apoiado na arquitetura sugerida no capítulo 5. Para isso, primeiramente é definido um caso de uso. O sistema foi implementado e conclusões puderam ser obtidas. Os mecanismos de comunicação CORBA foram testados, ou seja, invocação de métodos remotos por intermédio de *stubs* e invocações dinâmicas.

Finalmente, no Capítulo 7, é realizada a análise dos resultados e são citadas as sugestões para trabalhos futuros.

2 DESCRIÇÃO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES

2.1 INTRODUÇÃO

A função da infra-estrutura de telecomunicações é servir de apoio para a operação e funcionamento dos equipamentos provedores de telecomunicações, como sistemas de comutação, transmissão e telefonia celular entre outros. A infra-estrutura desempenha um papel tão importante quanto os próprios equipamentos de telecomunicações, visto que seu funcionamento afeta diretamente os serviços prestados pelas operadoras de telecomunicações[24].

Este capítulo tem por objetivo oferecer uma visão abrangente da infra-estrutura de telecomunicações sob a óptica do gerenciamento integrado. Veremos que ela é composta por um conjunto de subsistemas heterogêneos e que apresenta-se cada vez mais distribuída fisicamente, devido principalmente aos grandes esforços para redução de custos e maior escalabilidade, ou seja, a possibilidade de expansão.

2.2 CARACTERÍSTICAS GERAIS DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES

Entre as características da infra-estrutura de telecomunicações, as que mais impactam na implementação de sistemas de gerenciamento integrado são a distribuição física, a diversidade de equipamentos e interfaces proprietárias. Essas três características aliadas tornam a implementação de sistemas para gerenciamento da infra-estrutura de telecomunicações um desafio muito grande, fazendo-se imperativo o uso de novas tecnologias de informática. Veremos que CORBA pode solucionar boa parte dos problemas de distribuição e heterogeneidade.

2.2.1 Arquitetura distribuída

Por muito tempo alguns dos subsistemas que compõe a infra-estrutura de telecomunicações tenderam a ser implementados de forma centralizada. Entretanto, identificam-se três forças que se somam no sentido de levar a infra-estrutura de telecomunicações a estar cada vez mais distribuída:

A primeira delas originou-se com o aumento da densidade de potência dos sistemas de energia e com o advento das baterias seladas. Essa evolução permite que estes equipamentos possam ser instalados próximos dos sistemas de telecomunicações, chamados também de consumidores, eliminando a necessidade de salas especiais para estes tipos de equipamentos.

Outra tendência de descentralização de equipamentos de infra-estrutura são os sistemas de ar condicionado. A utilização de sistemas de condicionamento de ar dedicados e isolados, normalmente de menor porte, têm demonstrado serem operacionais e atraentes em termos de custos. O emprego de unidades modulares e distribuídas permitem otimizar os gastos com energia, sendo em várias ocasiões a única solução devido a problemas de espaço.

E finalmente, as estações rádio base para sistemas celulares, os estágios de linha remotos[2] e WLL - *Wireless Local Loop* forçaram o uso de arquiteturas de infra-estrutura inerentemente distribuída, pois nestes casos muitos dos equipamentos de telecomunicações são instalados remotamente às grandes estações, não podendo usufruir de uma infra-estrutura comum. Nestes casos a infra-estrutura segue as tendências de distribuição dos equipamentos que ela suporta.

2.2.2 Equipamentos e sistemas heterogêneos

Os equipamentos e sistemas que compõe a infra-estrutura de telecomunicações são heterogêneos[14]. Ou seja, devido a diversidade de funções exigidas na infra-estrutura, são empregados várias implementações e equipamentos distintos. Mesmo dois equipamentos que realizam funções iguais se diferem na forma como eles as executam.

Atualmente, nem todos os equipamentos são constituídos por componentes que estejam preparados para serem gerenciados. Mesmo os equipamentos dotados dos recursos necessários são provenientes de mercados multifornecedores e não apresentam interoperabilidade. Isso tem levado ao

desenvolvimento não padronizado de sistemas de gerenciamento que não estão preparados para funcionarem em conjunto.

2.2.3 Interdependência

A figura 1 ilustra de forma resumida a arquitetura física da infra-estrutura de telecomunicações, de acordo com a visão apresentada nas normas Telebrás[37].

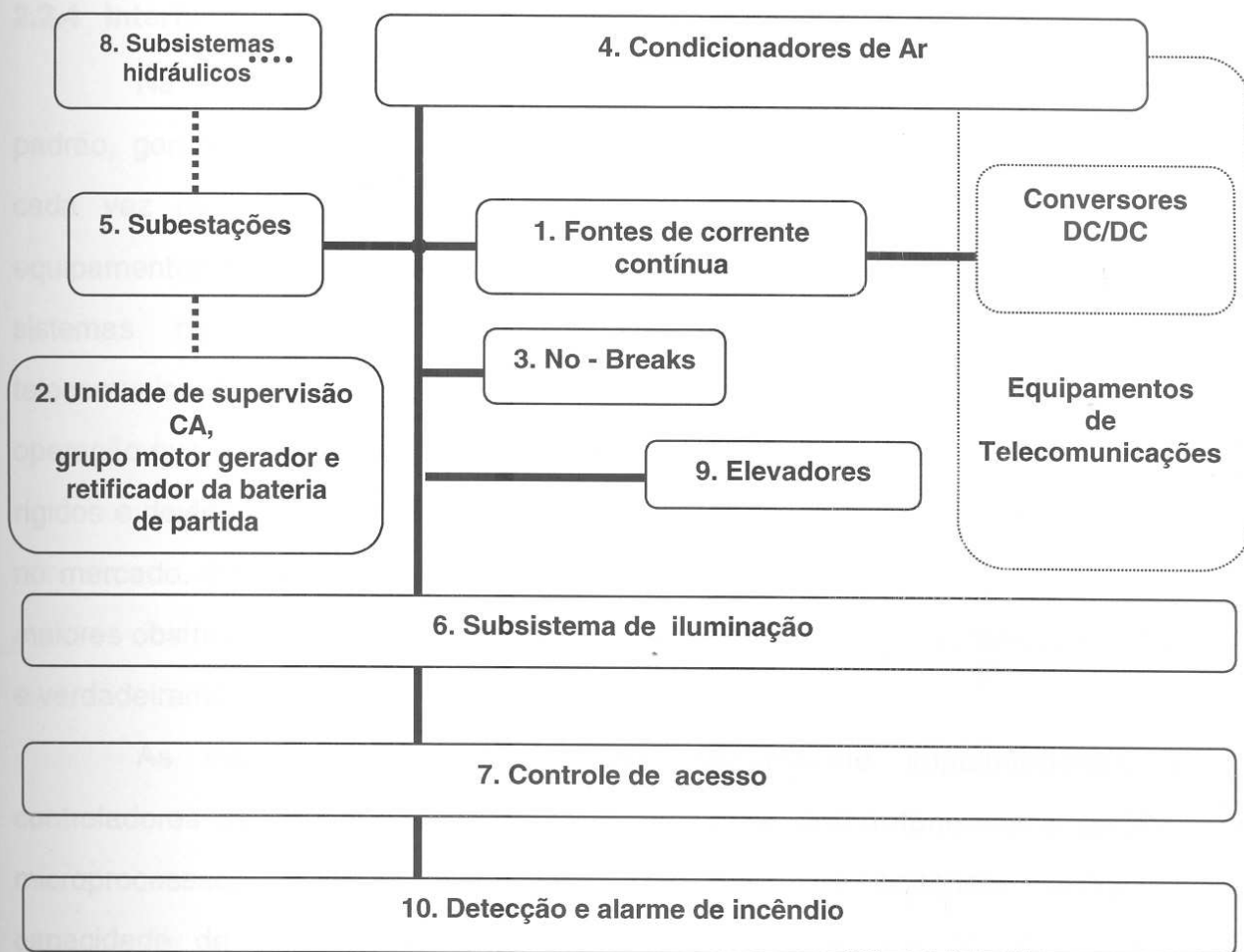


Figura 1 - Visão geral da infra-estrutura de telecomunicações

É importante ressaltar que os subsistemas não operam de forma independente, sendo que as conexões representam a ligação funcional entre os diversos subsistemas que formam a infra-estrutura. Portanto, um sistema de gerenciamento deve ser capaz de otimizar este relacionamento. Por exemplo, o controle da temperatura ambiente pode levar em conta as informações obtidas do

subsistema de controle de acesso, ou uma falha detectada em uma subestação pode ser minimizada ao se propagar para todos os demais subsistemas. Muitos outros exemplos poderiam ser formulados, mas o mais importante é ressaltar que existem vantagens econômicas e operacionais no gerenciamento integrado da infraestrutura.

2.2.4 Interfaces proprietárias

Na situação ideal existira apenas um sistema, que através de uma interface padrão, gerenciaria todo tipo de equipamento. Essa situação tem se distanciado cada vez mais da realidade porque, além da diversidade de padrões, os equipamentos de telecomunicações estão cada vez mais complexos e exigem sistemas dedicados e especializados. Em todas as operadoras de telecomunicações, existe grande quantidade e diversidade de equipamentos em operação que não possuem interface padronizada. Isso ocorre pela falta de padrões rígidos e devido a existência de vários tipos diferentes de equipamentos e soluções no mercado. Esta característica da infra-estrutura de telecomunicações é um dos maiores obstáculos a serem superados por um sistema de gerenciamento integrado e verdadeiramente aberto[24].

As interfaces proprietárias são normalmente implementadas entre controladores de pequeno porte, que são equipamentos autônomos e geralmente microprocessados. Estes dispositivos opcionalmente possuem relógio com capacidade de tempo real e são alocados próximo dos sistemas de infraestrutura[34]. Uma de suas funções é fazer a aquisição dos parâmetros e controlar equipamentos específicos. Os controladores possuem conexões digitais e analógicas específicas com os subsistemas de infra-estrutura e são capazes de exteriorizar parâmetros para gerenciamento. Tantos os sinais que entram no controlador quanto os que saem, são oriundos da interação com os subsistemas e de entidades externas. Estas entidades podem ser modems, Lap-tops, estações de trabalho, outros controladores ou sistemas de informática.

2.3 ARQUITETURA FÍSICA DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES

A arquitetura física da infra-estrutura de telecomunicações não trata-se de uma estrutura rígida, sendo que diferentes operadoras podem estabelecer critérios diferenciados de projeto. Estes critérios são dependentes do porte da instalação, localização e espaço físico disponível. O montante de capital disponível para investimento afeta igualmente a opção por equipamentos mais sofisticados, dotados de algum tipo de interface de gerenciamento, ou mais simples sem esta opção.

Optou-se neste trabalho pela divisão da infra-estrutura em subsistemas. Estes subsistemas se caracterizam por um conjunto de equipamentos que desempenham funções relacionadas e tendem a estar fisicamente próximos. Por exemplo, um subsistema de corrente contínua é composto por equipamentos como módulos retificadores, baterias, controladores e disjuntores, entre outros.

Visando ainda simplificar a compreensão da proposta de arquitetura de gerenciamento deste trabalho, a infra-estrutura de telecomunicações será dividida neste trabalho em dois grupos: Considera-se que o grupo I é composto por subsistemas que normalmente não possuem um sistema de gerenciamento legado e que necessitam de dispositivos controladores específicos comunicando-se diretamente com o sistema de gerenciamento. O grupo II tende a ser composto por sistemas legados, necessitando de algum tipo de conexão com o sistema de gerenciamento integrado.

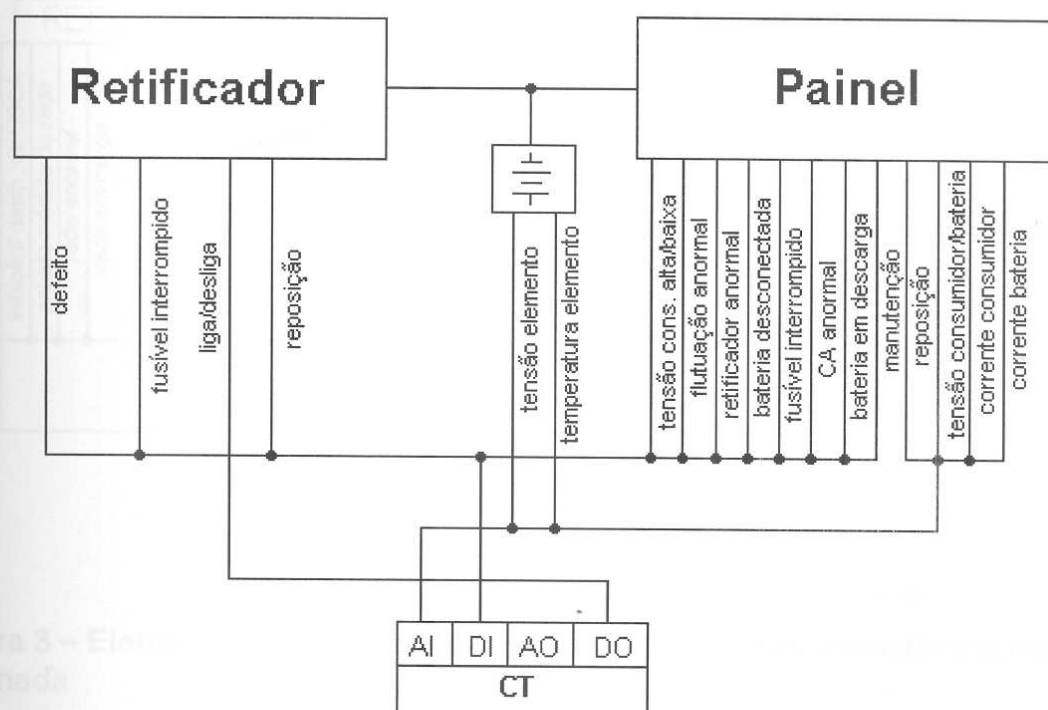
A arquitetura de gerenciamento a ser proposta mais adiante suporta tanto os subsistemas pertencentes ao grupo I quanto ao grupo II, pois para a implementação CORBA isso é transparente. Salienta-se que qualquer subsistema classificado como do grupo I pode vir a se tornar do tipo II, bastando que ele já seja representado de forma integrada por algum dispositivo ou sistema pré-existente.

Os parâmetros gerenciados são apresentados nos diagramas em blocos estabelecidos para os subsistemas pertencentes ao grupo I e serão utilizados para a análise realizada no capítulo 5, onde a arquitetura de informação para o gerenciamento da infra-estrutura é definida.

2.3.1 Subsistemas do Grupo I

2.3.1.1 Substistema de Corrente Contínua

As fontes de energia em corrente contínua, ou CC, são responsáveis pela retificação e conversão dos níveis em corrente alternada, ou CA, provenientes de subestações ou grupos motor gerador.



AI - entrada sinais analógicos
 DI - entrada sinais digitais
 AO - saída sinais analógicos
 DO - saída sinais digitais

Figura 2 – Elementos do subsistema de corrente contínua

A figura 2 apresenta o diagrama em blocos de um subsistema de corrente contínua e os parâmetros gerenciáveis. Verifica-se que vários parâmetros devem ser monitorados e ajustados. Ou seja, falhas ou anomalias de funcionamento devem ser sinalizadas e tratadas adequadamente. Por exemplo, um defeito ocorrido no retificador deve resultar no envio de um sinal de falha para um sistema de gerenciamento.

2.3.1.2 Subsistema de energia em corrente alternada

Este subsistema é composto pela unidade de supervisão CA, grupo motor gerador e retificador de bateria de partida, sendo responsável pelo fornecimento de energia em corrente alternada confiável para os sistemas de telecomunicações.

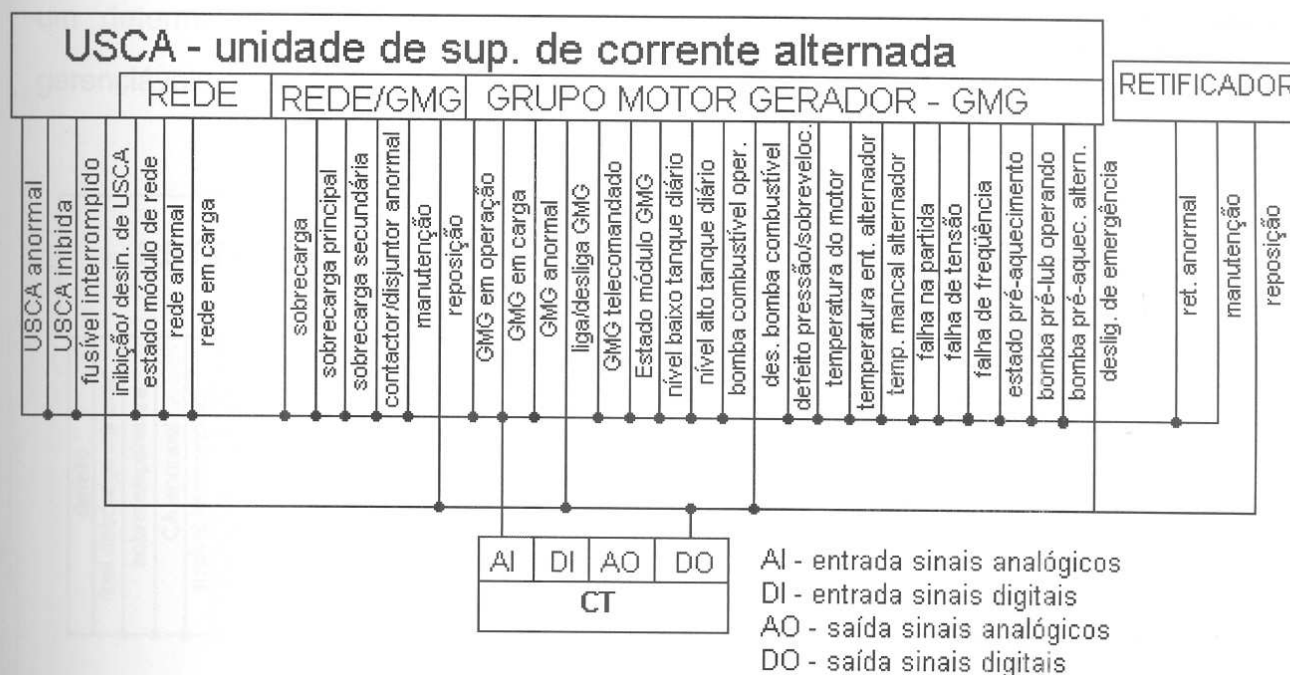


Figura 3 – Elementos do subsistema de fornecimento de energia em corrente alternada

A figura 3 mostra um diagrama em blocos com os parâmetros que devem ser gerenciados. Pretende-se com o gerenciamento deste subsistema monitorar o nível de combustível do GMG, falha de frequência, estado do retificador, sobrevelocidade, entre outros. Sempre que houver sinalizações de parâmetros fora de níveis, ações automáticas devem ser tomadas visando recuperar a falha ou evitar que esta se propague para os demais sistemas.

São previstas lógicas de controle que permitem partir e colocar em carga o Grupo Motor Gerador, a título de teste do equipamento. Isso também pode ser feito em períodos de alta demanda de energia CA, visando a redução dos custos com energia elétrica.

2.3.1.3 No-Break

Os sistemas de *No-Break* ou UPS (*Uninterrupted Power Supply*) são utilizados para garantir a estabilidade e segurança das ferramentas de informática administrativas aplicadas aos sistemas de telecomunicações, como por exemplo a própria rede corporativa da empresa e os sistemas de gerenciamento de equipamentos. Obviamente, um sistema de gerenciamento que seja atendido por um determinado UPS não deve gerenciá-lo. Ressalta-se que os parâmetros gerenciáveis apresentados na figura 4 são válidos para *No-Breaks* de grande porte.

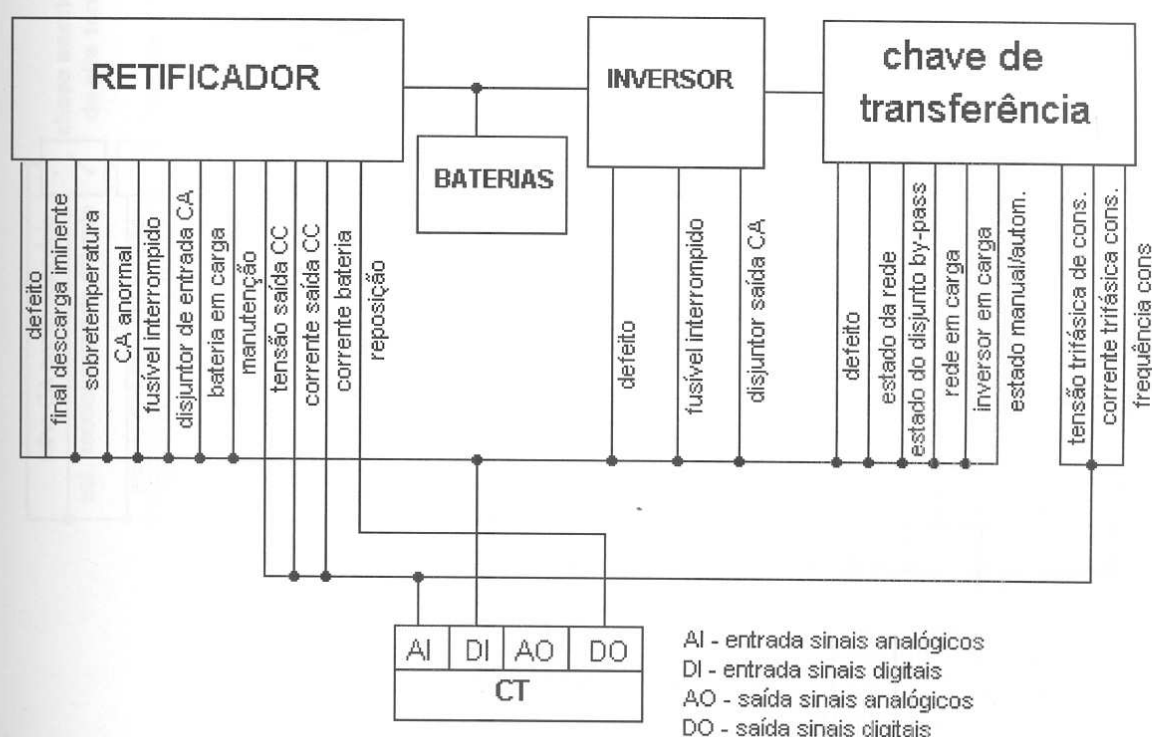


Figura 4 - Elementos gerenciáveis do No-Break

Como pode-se observar são especificados vários parâmetros que resultam em alarmes, como sobre-temperatura e tensão ou corrente de saída fora dos limites estabelecidos.

2.3.1.4 Subestações

Dependendo do porte da estação telefônica é necessário a existência de uma subestação dedicada devido à elevada demanda de energia. A subestação é composta por vários elementos que necessitam ser gerenciados. A partir do

gerenciamento de uma subestação, que alimente uma instalação de telecomunicações, pode-se verificar se as projeções de demanda estão devidamente sincronizadas com o relógio de medição da concessionária de energia local. Isso possibilita que a estimativa de demanda média em um período de tempo seja rigorosamente idêntico ao da concessionária. Caso os valores resultem na ultrapassagem dos valores permitidos, o sistema de gerenciamento pode proceder,

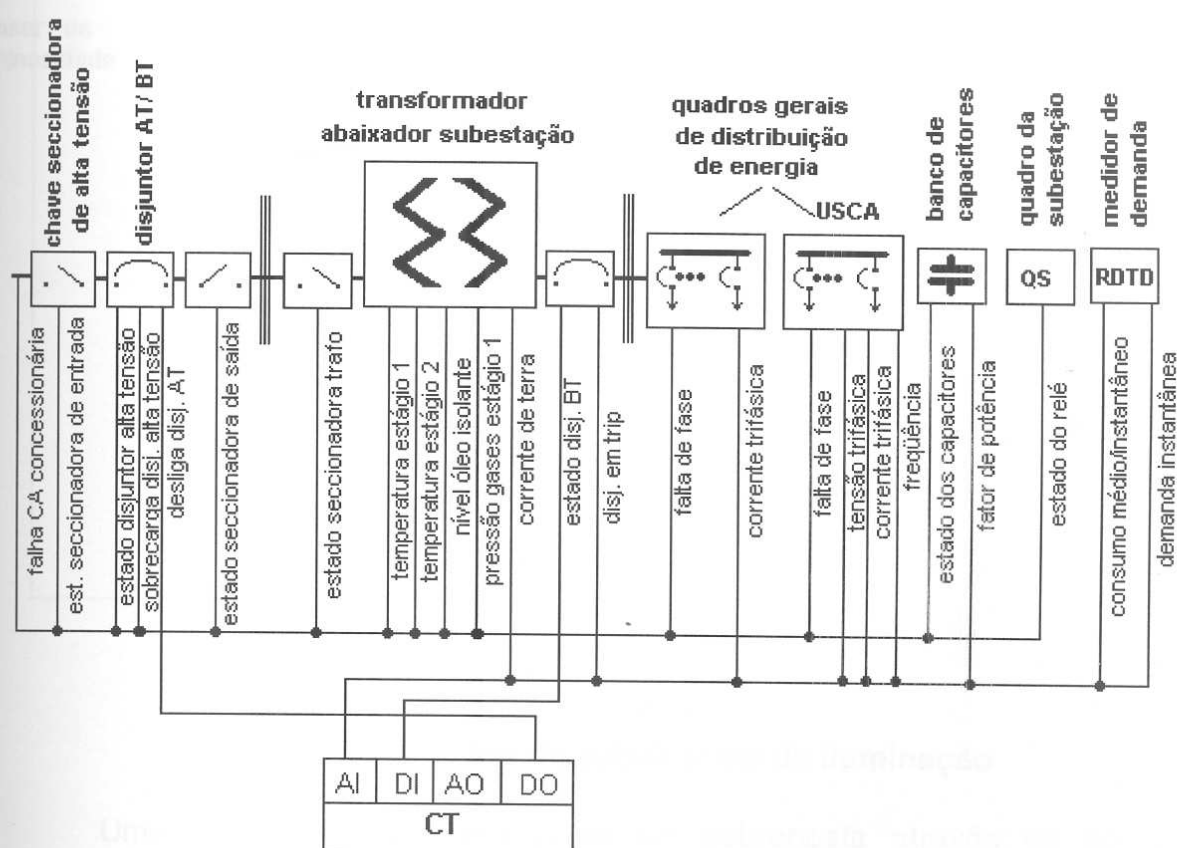


Figura 5 - Elementos da subestação de energia

por exemplo, o desligamento seletivo de cargas e /ou alteração de ajustes de temperaturas de centrais de água gelada e/ou condicionadores de ar, isoladamente ou em grupo, objetivando diminuir a energia consumida instantaneamente pela instalação. A figura 5 apresenta os elementos gerenciáveis deste subsistema.

2.3.1.5 Subsistema de iluminação

O controle da iluminação externa pode ser feito via programação horária ou pela atuação de foto-células visando basicamente a economia de energia elétrica. O

controle do nível de iluminação interna pode ser estabelecido via programação horária ou por intermédio de reatores eletrônicos controlados, a partir da medição da luminosidade externa, visando sempre otimizar o consumo de energia elétrica..

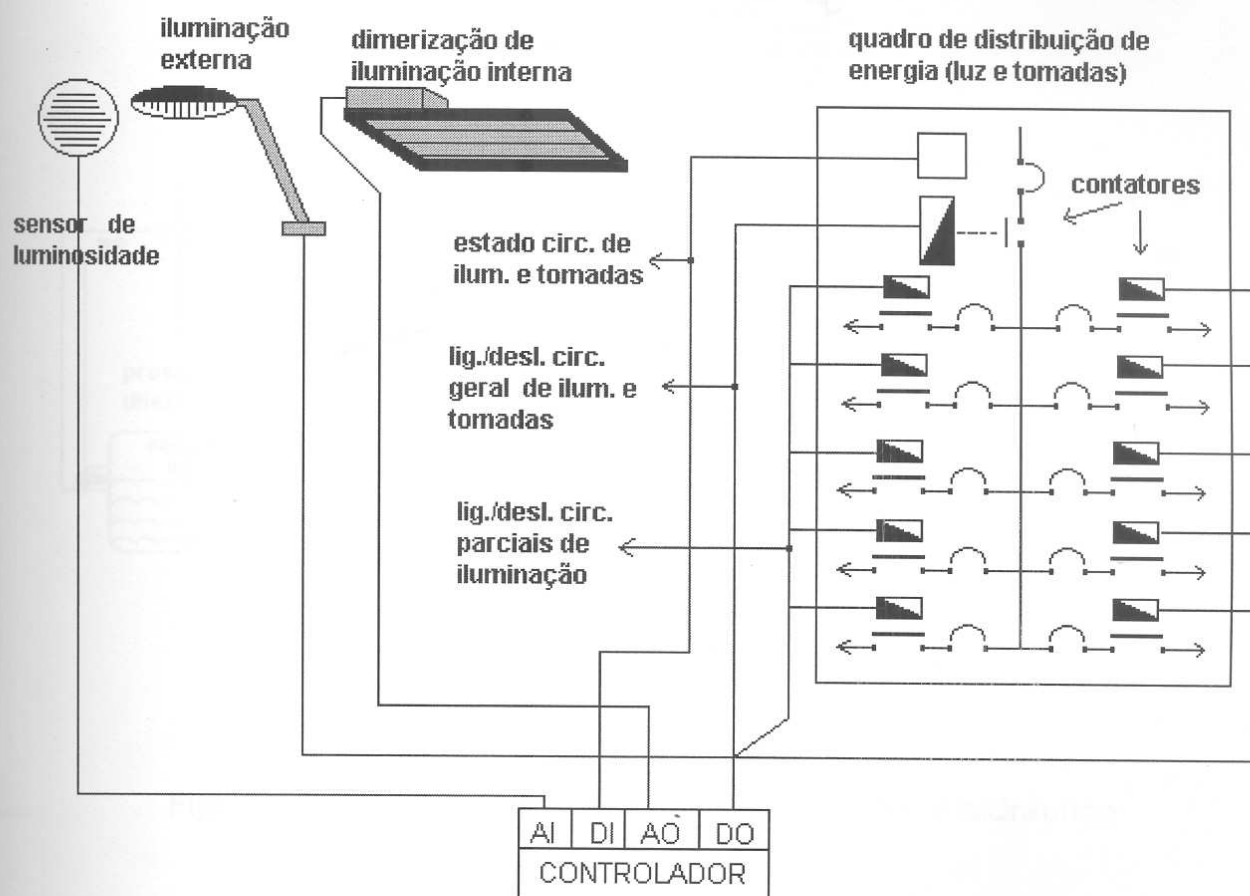


Figura 6 - Elementos do subsistema de iluminação

Uma programação horária pode ser sobreposta através de comando manual, sensores de presença, ou por intervenção direta do sistema de gerenciamento. Deve ser possível o controle total ou parcial dos circuitos do quadro de distribuição de iluminação através de chaves contactoras. A figura 6 apresenta de forma resumida os parâmetros gerenciáveis.

2.3.1.6 Subsistemas hidráulicos

Os subsistemas devem ser gerenciados para otimizar os gastos de energia e para garantir o funcionamento do sistema de ar condicionado. Os sistemas hidráulicos apresentados na figura 7 são compostos por bombas, caixas e poços de

água, entre outros. São utilizados medidores de nível nas caixas e poços, que levarão ao ligamento e desligamento das bombas, gerando alarmes de nível de água baixo ou alto.

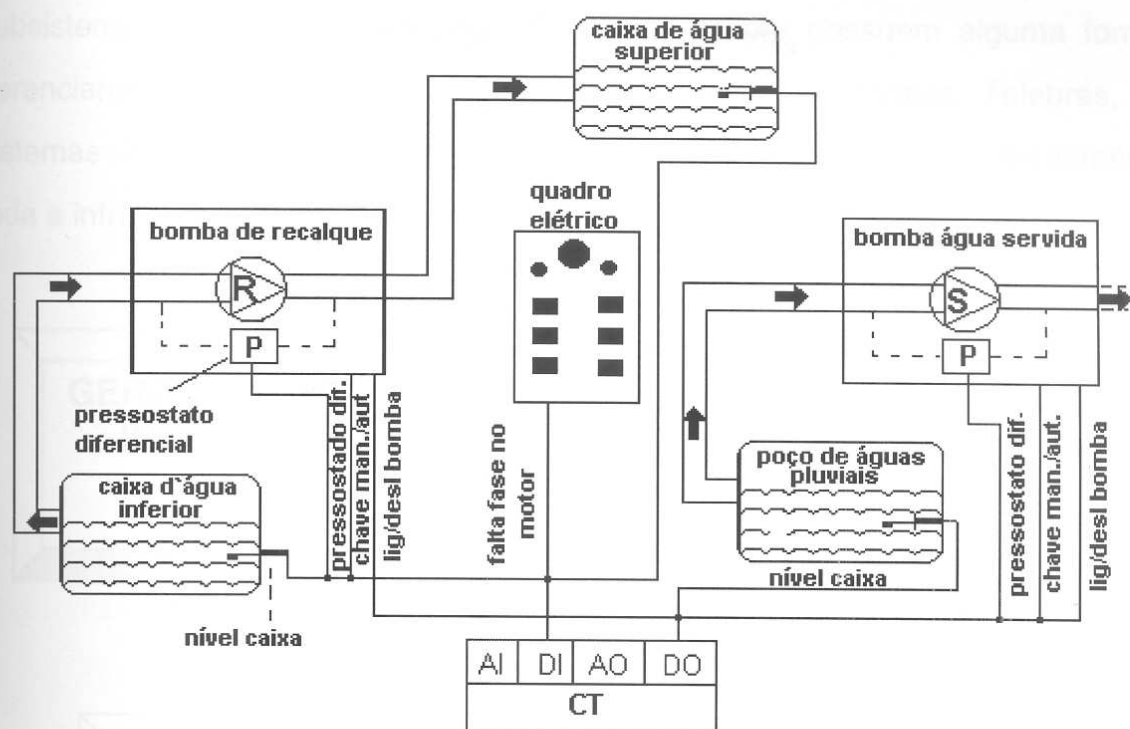


Figura 7 - Elementos gerenciáveis do subsistema hidráulico

Os estados das bombas devem ser notificados para o sistema de gerenciamento. No caso de falta de energia as informações de defeitos devem ser inibidas. Quando existirem mais de uma bomba, sempre que a que estiver em operação apresentar defeito, deverá ser acionada imediatamente a bomba reserva. Sempre que houver sinalização de falta de fase nos quadros de alimentação das bombas, o sistema deverá emitir imediatamente o comando para desligar as bombas. A partir dos dados históricos da infra-estrutura, pode-se estabelecer rodízio no funcionamento das bombas em função do tempo acumulado, visando equalizar desgastes mecânicos nas mesmas.

2.3.2 Subsistemas do Grupo II

Qualquer subsistema classificado anteriormente pode vir a se tornar do tipo II. Basta que ele já seja gerenciado por algum sistema isolado. Portanto, os subsistemas pertencentes ao segundo grupo sempre possuem alguma forma de gerenciamento independente. Entretanto, segundo as normas Telebrás, estes sistemas devem ser integrados a um outro sistema que efetue o gerenciamento de toda a infra-estrutura[37]. A figura 8 ilustra esta conexão.

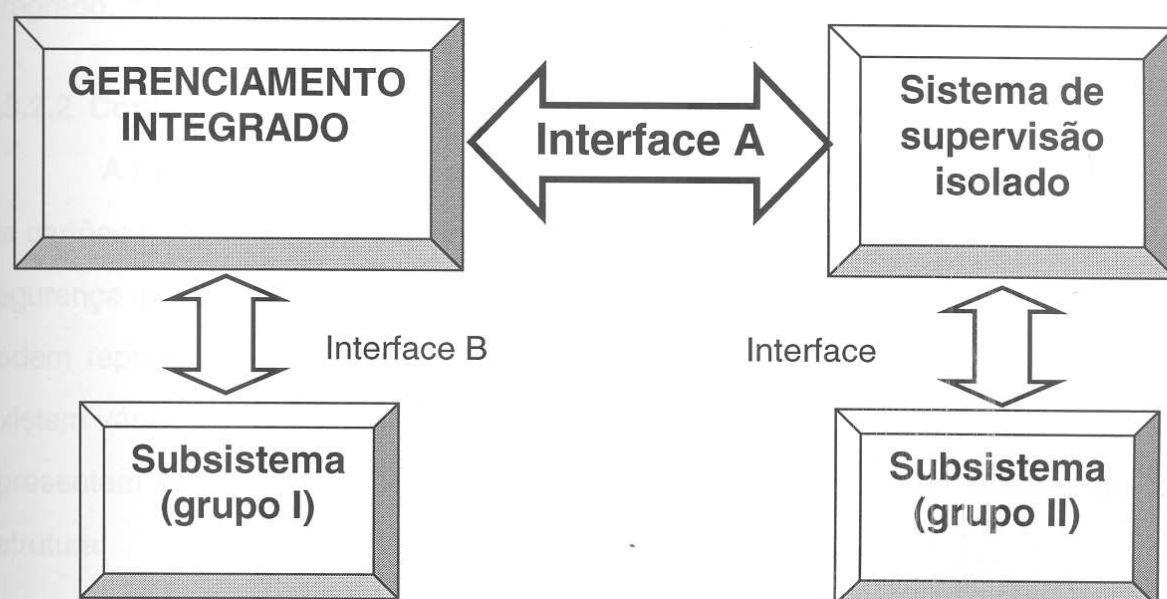


Figura 8 - Integração dos subsistemas do tipo 2 com o sistema de gerenciamento da infra-estrutura

Como pode-se observar um subsistema do grupo II, dotado de uma interface C, é gerenciado por um sistema de supervisão isolado. Para que este subsistema seja gerenciado por um gerenciamento integrado é necessário estabelecer um interfaceamento (interface A) entre o sistema de supervisão isolado e o gerenciamento integrado. Já o subsistema do grupo I, possui conexão direta com o gerenciamento integrado.

2.3.2.1 Elevadores

Um sistema de gerenciamento de infra-estrutura deve possuir interface com o sistema de elevadores com a finalidade de apresentar uma forma redundante de gerenciamento de todos os alarmes e/ou funções consideradas de relevância. Esta integração fornece mais segurança, tanto para os usuários dos elevadores quanto para a operadora de telecomunicações. Uma integração deste tipo pode até mesmo salvar vidas. Por exemplo, o subsistema de elevadores pode tomar ações baseadas no processamento de informações do sistema de gerenciamento integrado, oriundas do subsistema de detecção e alarme de incêndio.

2.3.2.2 Controle de acesso

A maioria das grandes empresas possuem sistemas de controle de acesso via cartões magnéticos ou código de barras. As razões são basicamente relativas a segurança e controle de horário dos funcionários, pois janelas e portas abertas podem representar brechas na segurança física e exposição a danos ambientais. Existem vários fornecedores para estes subsistemas, sendo que quase sempre não apresentam possibilidade de integração direta com outros subsistemas de infra-estrutura.

A interligação do subsistema de controle de acesso com um sistema de gerenciamento integrado visa otimizar o consumo de ar condicionado e iluminação das edificações pela interação com os demais subsistemas. As normas brasileiras[37] especificam que no caso de invasão em áreas que possam afetar a planta de telecomunicações deve sempre ocorrer envio de alarmes ao sistema de gerenciamento integrado.

2.3.2.3 Detecção e alarme de incêndio

A fumaça e os gases perigosos podem alertar previamente riscos para equipamentos e para pessoas. Portanto, a monitoração de parâmetros de ambientes e de prédios de telecomunicações proporciona maior segurança. O sistema de gerenciamento integrado de infra-estrutura deve possuir interface com o subsistema de detecção e alarme de incêndio com a finalidade de apresentar uma

forma redundante na estação central de todos os alarmes e/ou funções consideradas de relevância. O sistema de detecção e alarme de incêndio, por norma, é independente do sistemas de gerenciamento integrado, porém a troca de informações com o mesmo aumenta o grau de segurança de todos os subsistemas.

2.3.2.4 Condicionadores de Ar

Os equipamentos de telecomunicações são especificados para operarem em faixas de temperatura e umidade determinadas, pois estes parâmetros afetam a confiabilidade dos equipamentos eletrônicos. Porém, estes sistemas são os maiores responsáveis pelos gastos de energia nas estações de telecomunicações. O gerenciamento destes dentro de um contexto que englobe toda a infra-estrutura de telecomunicações é fundamental para o sucesso das operadoras.

A relação de pontos de condicionamento de ar varia em função das características dos equipamentos a serem utilizados. Isso torna os requisitos de funcionamento particulares para cada instalação, devendo estes serem considerados por ocasião do projeto. Portanto, cada instalação necessita de um gerenciamento específico.

Pelo gerenciamento dos subsistemas de ar condicionado é sempre possível corrigir condições anormais muito antes que de outra forma. Assim sendo, o gerenciamento dos sistemas de condicionamento de ar são fundamentais para garantirem a correta operação e a vida útil equipamentos de telecomunicações e da própria infra-estrutura.

2.3.3 Requisitos para gerenciamento informatizado da infra-estrutura de telecomunicações

O gerenciamento informatizado da infra-estrutura de telecomunicações objetiva a redução dos custos de operação e manutenção e o aumento da confiabilidade dos serviços ofertados aos usuários de telecomunicações. Veremos que para atingir estes objetivos é necessário levar em conta alguns requisitos importantes.

Pretende-se a seguir o levantamento dos requisitos para o gerenciamento informatizado da infra-estrutura de telecomunicações. O levantamento destes requisitos baseia-se na prática Telebrás “Especificações Gerais de Sistema de Supervisão e Automação de Infra-estrutura”[37], nos seminários de infra-estruturas e energia do sistema de telecomunicações brasileiro, em congressos internacionais de infra-estrutura para telecomunicações e informações obtidas de fornecedores de soluções para gerenciamento dos equipamentos de infra-estrutura. Obviamente os requisitos e especificações apontadas são passíveis de alterações futuras, visando sempre o aprimoramento contínuo.

Durante a elaboração desta pesquisa ocorreu a privatização do sistema de telecomunicações brasileiro. Apesar disso, as normas brasileiras continuarão a ser exigidas pela ANATEL - Agência Nacional de Telecomunicações. Existem atualmente esforços para que as normas e práticas venham a se tornar normas NBR - Normas Brasileiras. Portanto, as considerações apontadas por este trabalho, decorrentes das normas Telebrás, continuam válidas.

2.3.4 Funcionalidades do sistema de gerenciamento de infra-estrutura

Genericamente, os sistemas de gerenciamento, tais como o sistema de gerenciamento de infra-estrutura de telecomunicações abordado nesta dissertação, devem atender a um conjunto de funcionalidades[5] classicamente definidas como gerenciamento de falhas, desempenho, configuração, segurança e contabilização. Em princípio o gerenciamento de contabilização preocupa-se com a manutenção e

monitoração de quais e quantos recursos estão sendo utilizados. Estas informações podem ser utilizadas para estatísticas ou faturamento (bilhetagem). Esta funcionalidade apresenta uma papel secundário no gerenciamento de sistemas de infra-estrutura, não sendo incluída nesta abordagem.

2.3.4.1 Gerenciamento de falhas

Esta função de gerenciamento objetiva a detecção, o isolamento e a correção de falhas. Isto é feito por meio da monitoração, quase que em tempo real, sobre os subsistemas da infra-estrutura. Como consequência da monitoração de alarmes, a gerência destes sinais deve identificar a gravidade e a sua exata localização dentro do sistema, e então apresentar o resultado ao usuário do sistema. Normalmente os sinais de falhas são rápidos e apresentam poucos detalhes.

2.3.4.2 Gerenciamento de configuração

Esta é a função que proporciona a identificação do estado operacional dos equipamentos e simultaneamente o controle dos mesmos. Essas informações são importantes na determinação de novos investimentos e na tomada de decisões com relação ao provisionamento novos equipamentos. Pode-se inclusive realizar ações preventivas, evitando possíveis falhas futuras. Normalmente este tipo de gerenciamento exige uma largura de banda mais larga, devido à grande quantidade de dados coletados.

2.3.4.3 Gerenciamento de desempenho

A função gerência de desempenho encarrega-se da monitoração e análise de desempenho dos sistemas. Esta gerência deve manter a monitoração permanente sobre a qualidade dos equipamentos e reconfigurá-los quando necessário, visando manter a eficácia da rede e o desempenho dos serviços ofertados. Exemplificando, os sistemas de demanda de energia e subestações devem ter condições de realizar medição e cálculo de consumo e demanda de forma a poder avaliar detalhadamente as características da instalação.

2.3.4.4 Gerenciamento de segurança

Alterar estratégias de carga de baterias ou do funcionamento do subsistema de ar condicionado de forma incorreta pode vir a danificar equipamentos e consequentemente comprometer os serviços prestados aos usuários. Um sistema que ofereça acesso a parâmetros importantes da infraestrutura deve garantir que pessoas não autorizadas e despreparadas acessem o sistema, ou seja, deve-se preservar a segurança do sistema.

A gerência de segurança proporciona garantias de acesso às informações estratégicas somente aos usuários autorizados, mantendo a integridade dos dados, assim como o processamento destes isentos de interferências e violações.

2.3.5 Requisitos para arquiteturas de sistemas de gerenciamento

Os requisitos básicos para qualquer arquitetura de sistema de gerenciamento são:

2.3.5.1 Sistema aberto

Um sistema aberto garante a capacidade de mudanças para sistemas mais complexos sem perder os recursos alocados. Veremos mais adiante que alguns sistemas de gerenciamento integrado de infra-estrutura apresentam algum grau de abertura, entretanto estão longe de serem considerados abertos. Sistemas abertos são caracterizados pelo fato de suas interfaces serem publicadas. Para o gerenciamento da infra-estrutura esta abertura está aliada à necessidade de interfaces tão simples quanto os próprios equipamentos gerenciados, sendo que apenas o uso de arquiteturas abertas não garantem a interoperabilidade, pois é fundamental que as interfaces sejam de domínio público.

2.3.5.2 Sistema modular

A modularidade garante que soluções particulares atendam as necessidades dos clientes em suas aplicações, assim como a inclusão de novas funções de forma rápida e objetiva. Para a infra-estrutura de telecomunicações esta característica é muito importante, pois permite a solução de problemas específicos através da inclusão de novas facilidades.

2.3.5.3 Sistema escalonável

A força que conduz as novas tecnologias é a necessidade da escalabilidade dos sistemas[38], sendo que o gerenciamento integrado da infra-estrutura de telecomunicações também necessita desta característica. Geralmente a infra-estrutura de telecomunicações é expandida de acordo com a demanda dos equipamentos de telecomunicações. Um sistema escalonável permite que o usuário possa iniciar a implantação de forma progressiva, aumentando as dimensões e abrangências de acordo com as necessidades.

2.3.5.4 Sistema distribuído

Sistemas distribuídos garantem o suporte às distribuições físicas e funcionais de acordo com as necessidades geográficas e organizacionais de uma empresa. Conforme visto anteriormente, a infra-estrutura de telecomunicações é composta por subsistemas distribuídos. Portanto, uma arquitetura que se preste para o gerenciamento da infra-estrutura deve suportar esta característica.

Sistemas distribuídos que apresentam características de abertura são baseados no fornecimento de mecanismo de comunicação uniforme entre processos distribuídos e interfaces públicas para acesso aos recursos compartilhados.

2.3.5.5 Gerenciamento integrado

No gerenciamento da infra-estrutura é fundamental obter a máxima utilização da instalação, armazenamento de dados e recursos de comunicação. O gerenciamento integrado proporciona uma visão ampla dos subsistemas gerenciados, possibilitando considerar as influências entre eles.

O gerenciamento integrado leva a uma maior sinergia entre os subsistemas da infra-estrutura, resultando em ganhos econômicos e maior confiabilidade, pois a correção rápida de condições indevidas reduz a exposição de equipamentos a situações de risco.

Salienta-se que apesar dos subsistemas de infra-estrutura serem distribuídos, é desejável que o seu gerenciamento seja integrado.

2.3.5.6 Interface amigável com o usuário

Uma interface com o usuário deve ser preferencialmente gráfica e apresentar hierarquias, símbolos, gráficos e mapas do sistema. Além disso, é fundamental que esta interface seja eficiente, suportando a mesma vista dos sistemas com ícones e símbolos que representem a realidade. É interessante que o usuário encontre um objeto visualmente ou procurando pelo equipamento a partir de uma janela de busca.

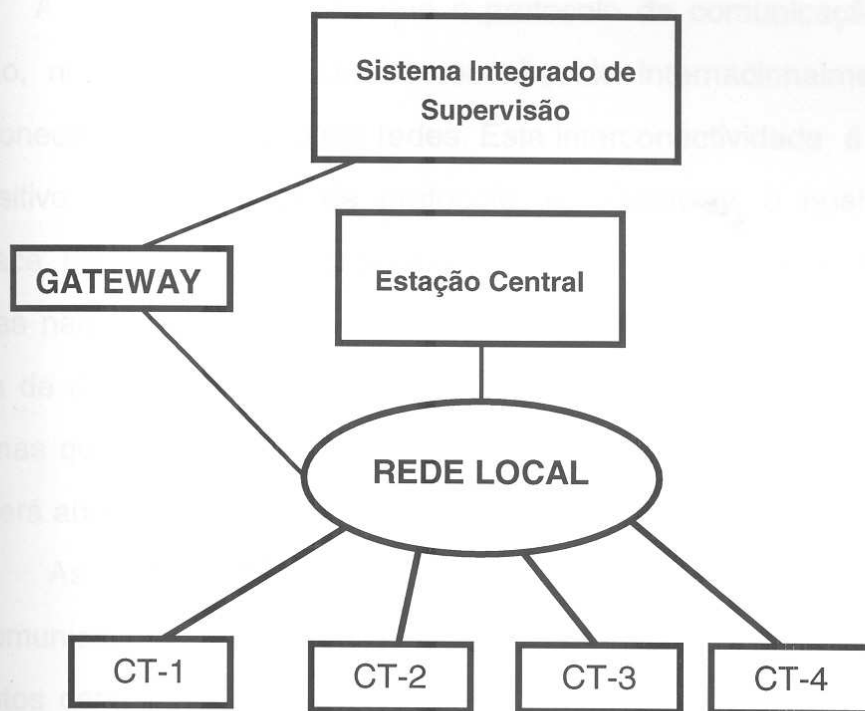
2.3.6 Requisitos oriundos de Práticas Telebrás

A Telebrás especificou um Sistema de Supervisão e Automação da Infra-Estrutura, denominado SAIE, como sendo um sistema informatizado que objetiva realizar o gerenciamento de uma instalação de telecomunicações, exercendo a supervisão, controle e operação dos subsistemas que compõe a infra-estrutura[37]. Ao operador do SAIE é necessário oferecer a possibilidade de acionamento remoto dos equipamentos selecionados, modificações de pontos de ajuste, modificação de programação horária, habilitação e inibição de execução de processos e alteração de data e hora, entre outras facilidades.

Neste sistema, para se executar o gerenciamento da infra-estrutura é necessário coletar todas as informações de alarmes gerados por todos os subsistemas e enviá-los ao menos para um local de armazenamento, tornando possível um controle estatístico do número de defeitos que deixam o equipamento inoperante para o usuário num dado tempo.

A TELEBRÁS especificou uma arquitetura centralizada, dotada de uma Estação Central – EC, que tem como função gerenciar a infra-estrutura, mediante comandos do operador e/ou controladores, além de gerar relatórios, gráficos, quadros sinóticos e alarmes, agilizando as ações de manutenção preventivas, corretivas e de segurança. Esta estação central deve realizar o gerenciamento das funções de alarmes baseados em classes de prioridades de atendimento e criticidade, como por exemplo, utilizando filtros para visualização seletiva. Os eventos devem estar associados a visualização e manipulação de alarmes relacionado-os com data e hora da ocorrência.

A figura 9 apresenta a arquitetura física proposta pelas práticas Telebrás. Para que se possa gerenciar os diversos subsistemas de forma integrada, a Telebrás especificou uma rede de telecomunicações. Esta pode ser uma rede dedicada que interliga os diversos elementos de controle, ou seja, controladores à estação central, permitindo desta forma ações de gerenciamento.



CT - controlador

Figura 9 - Arquitetura funcional do SAIE[37]

As normas definem que esta rede de comunicação deve apresentar pelo menos as seguintes características:

- ♦ Constituir-se em uma rede que permita a comunicação direta entre os elementos a ela conectados.
- ♦ Possuir condições de configuração automática, permitindo que todos os equipamentos operacionais desempenhem as funções designadas da forma mais eficiente possível.
- ♦ Detecção, correção de erros e/ou re-transmissão para garantir a integridade dos dados.
- ♦ Possibilitar a sincronização obrigatória de todos os relógios de tempo real em todos os subsistemas.
- ♦ Taxa de transmissão na rede local, bem como através de modem, compatível com a qualidade de informação desejada, isto é, não poderá haver retardo na transmissão que comprometa a agilidade nas providências por parte do operador da estação central.

A Telebrás especifica que o protocolo de comunicação de rede deve ser padrão, normalizado por comitê reconhecido internacionalmente, visando maior interconectividade com outras redes. Esta interconectividade é efetuada através de dispositivo de adaptação de protocolo, ou *Gateway*, o qual deve disponibilizar interface padrão para a comunicação. Portanto, apenas o uso de arquiteturas abertas não garantem a interoperabilidade, pois é fundamental que as interfaces sejam de domínio público. Este requisito objetiva possibilitar a integração com os sistemas que seguem a filosofia TMN - *Telecommunication Management Network* que será abordada mais adiante.

As normas e práticas referentes ao gerenciamento da infra-estrutura de telecomunicações visam permitir a compatibilidade entre os vários fornecedores de produtos dentro das necessidades das operadoras. Porém, o estabelecimento de padrões rígidos pode, por outro lado, inviabilizar a evolução tecnológica. Esse é um dos piores impedimentos com relação ao estabelecimento de padrões proprietários. Assim sendo, são desejáveis soluções que permitam um ambiente multifornecedor e que promovam a evolução tecnológica.

Veremos que a arquitetura proposta neste trabalho é mais abrangente que a arquitetura centralizada definida pela Telebrás. Contudo, muitos dos requisitos apontados nas práticas são considerados fundamentais e contemplados na proposta.

2.3.7 Níveis de gerenciamento

MOTA e GUIMARÃES classificaram as arquiteturas para sistemas de gerenciamento de infra-estrutura e sugeriram os níveis mostrados na figura 10[24].

O nível mais alto é constituído pelos sistemas de gerenciamento corporativos. Estes sistemas gerenciam não somente a planta de telecomunicações e sua infra-estrutura, mas a empresa operadora como um todo. Esse nível de gerenciamento é atendido pela arquitetura TMN, a qual será abordada mais adiante.

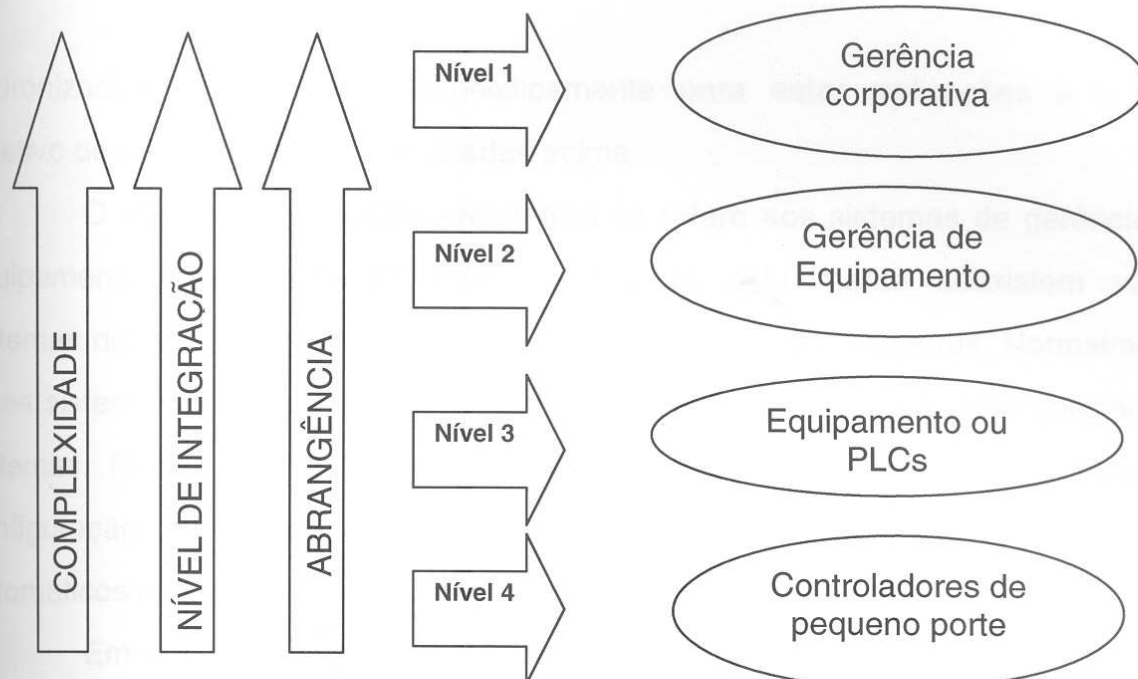


Figura 10 - Níveis de gerenciamento

O ITU – *International Telecommunication Union* considera como funções de gerência corporativa:

- ♦ Administração e gerência de pessoal.
- ♦ Gerência de marketing.
- ♦ Gerência de finanças corporativas.
- ♦ Contabilização e controle.
- ♦ Planejamento empresarial de serviços.
- ♦ Planejamento de tarifas.
- ♦ Planejamento a longo prazo das redes de telecomunicações.
- ♦ Controle de operações.
- ♦ Gerência de Contas de grandes clientes.
- ♦ Gerência da interface com outras operadoras e provedores de serviços.
- ♦ Gerência de serviços (Tarifação, Administração de clientes, ordens de serviços, gerência de informações da planta, gerência da força de trabalho).

Como pode-se observar, o nível de gerência corporativa não consiste apenas de um único sistema, mas vários sistemas que se integram por interfaces

padronizadas, desenvolvidos especificamente para estas aplicações e com o objetivo de viabilizar as funções citadas acima.

O segundo nível desta arquitetura se refere aos sistemas de gerência de equipamentos da planta de telecomunicações. Atualmente coexistem vários sistemas de gerenciamento provenientes de vários implementadores. Normalmente estes sistemas de gerenciamento não estão preparados para conexão com outros sistemas. Neste nível estão as funções de gerência de falhas e gerência de configuração e desempenho, assim como as ferramentas para execução de testes automáticos na planta e alterações de software dos equipamentos.

Em geral os sistemas de gerenciamento de infra-estrutura possuem níveis de complexidade menores do que o gerenciamento de equipamentos de telecomunicações. Além disso, não há necessidade de uma gerência de rede, uma vez que os equipamentos de infra-estrutura tem uma atuação pontual. Normalmente a inteligência reside nos próprios equipamentos ou em equipamentos auxiliares, como controladores microprocessados.

O gerenciamento realizado no nível 3 necessita de equipamentos com microprocessadores incorporados e interface de comunicação. Uma alternativa de implementação de um sistema de gerenciamento neste nível pode ser feita com o uso de programadores lógicos controláveis - PLCs ou rede de PLCs atuando no controle e automação de equipamentos e utilidades da infra-estrutura. Um sistema nestes moldes não é indicado para aplicações de grande porte e que deseje ser suportada por um mercado multifornecedor.

O nível 4 refere-se efetivamente aos equipamentos e utilidades de infra-estrutura que devem ser gerenciados. Este nível utiliza controladores mais simples, mais baratos e de uso geral. Sob o ponto de vista de protocolos de comunicação, este nível tem como característica uma troca rápida de pequenas quantidades de informações. Por outro lado, os protocolos de nível 3 são protocolos mais sofisticados, permitindo a troca de blocos de informações.

A decisão do nível em que se pretende implementar o sistema de gerenciamento deve levar em consideração o aproveitamento dos diversos sistemas legados, preservando os investimentos realizados.

2.3.8 Sistemas Comerciais

Para a gerência verdadeiramente integrada da infra-estrutura seria necessário que os fabricantes desenvolvessem seus produtos dentro dos padrões estabelecidos por normas e com interfaces padronizadas, proporcionando a conectividade com os sistemas de gerenciamento. Entretanto na prática as soluções diferem devido a fatores específicos de mercado, nível de gerenciamento utilizado e tecnologia empregada. Isso ocorre principalmente porque a concorrência leva à busca pela liderança tecnológica que, apesar dos benefícios imediatos obtidos, promove uma grande diversidade de tecnologias que prejudicam a interoperabilidade dos sistemas.

As soluções proprietárias não são interessantes para as operadoras de telecomunicações, pois tornam a gerência da infra-estrutura ineficiente, já que envolvem problemas como a presença de múltiplas interfaces para subsistemas semelhantes, geralmente não integráveis e insuficiência de informações coletadas.

Segundo um estudo comparativo efetuado pela TELEMIG - Telecomunicações de Minas Gerais[24], pertencente ao sistema Telebrás, das soluções de mercado e as suas características, constatou-se que grande parte dos subsistemas de infra-estrutura, que apresentam unidades de supervisão e controle microprocessadas, não são integráveis facilmente por se tratarem de soluções proprietárias.

Os sistemas comerciais avaliados oferecem algum tipo de acesso remoto e são capazes de emitir relatórios, sendo que todos eles objetivam fornecer uma visão abrangente dos subsistemas, possibilitando que as equipes de manutenção sejam devidamente acionadas na correção dos defeitos. Entretanto, observa-se que os recursos fornecidos pelos vários tipos de sistemas de gerenciamento não são os mesmos. Cada um possui características diferenciadas de software de supervisão,

protocolos de transmissão, e métodos de acesso. Isso traz enormes dificuldades para o gerenciamento integrado da infra-estrutura. Como muitos destes sistemas já estão desenvolvidos e operando em muitas localidades, apenas uma padronização mais detalhada não resolve a maioria dos problemas de conectividade. Por outro lado, migrar toda a estrutura atual para uma arquitetura padrão seria economicamente inviável.

O desenvolvimento de interfaces adaptadoras baseadas em CORBA podem facilitar a integração, sendo necessário estabelecer um mapeamento dos dados dos parâmetros gerenciáveis e das funcionalidades executadas. Veremos que o emprego de CORBA pode proporcionar a integração dos sistemas legados, assim como o desenvolvimento de novos sistemas.

3 ESTUDO DAS TECNOLOGIAS APLICADAS AO GERENCIAMENTO DE INFRA-ESTRUTURA

3.1 INTRODUÇÃO

Neste capítulo objetiva-se apresentar as principais tecnologias que estão sendo propostas para o gerenciamento da infra-estrutura de telecomunicações. Não pretende-se provar que uma tecnologia seja melhor que a outra, mas sim identificar as suas características e verificar a sua aplicação no gerenciamento da infra-estrutura de telecomunicações.

A implementação de sistemas de gerenciamento de telecomunicações sofre atualmente a influência de três frentes fundamentais, que são: os requisitos estabelecidos nas especificações TMN – *Telecommunication Management Protocol*, o uso de SNMP – *Simple Management Protocol* no gerenciamento da infra-estrutura de telecomunicações[2], e finalmente, a aplicação das novas tecnologias de computação distribuída na busca de soluções técnicas inovadoras que proporcionem melhores oportunidades para os usuários e fornecedores dos serviços de gerenciamento[22].

3.2 O MODELO TMN- TELECOMMUNICATION MANAGEMENT NETWORK

A TMN oferece uma estrutura organizada que possibilita conectar e operar diferentes sistemas e equipamentos de telecomunicações, visando a troca de informações de gerenciamento por intermédio de interfaces padronizadas e constituídas por protocolos específicos. O gerenciamento da infra-estrutura para telecomunicações faz parte dos objetivos da TMN.

A figura 11 apresenta de forma sucinta o modelo TMN e seus blocos funcionais, sendo que as suas especificações são resumidas a seguir:

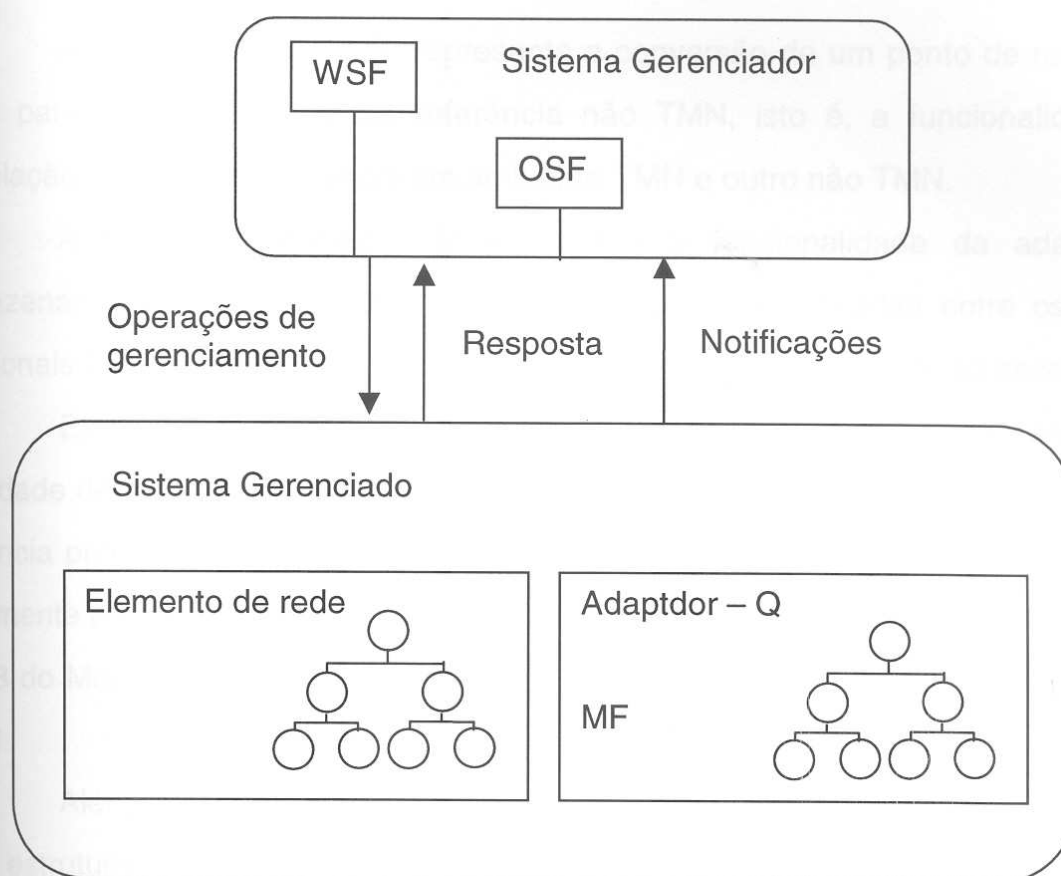


Figura 11 - Modelo TMN e seus blocos funcionais

O OSF - *Operations System Function* (função sistema operacional) realiza o processamento de informações gerenciais com propósito de controlar, monitorar e/ou coordenar funções.

O bloco funcional estação de trabalho ou WSF, representa a funcionalidade da troca de informações gerenciais entre a TMN e os operadores, através de uma estação de trabalho que pode ser constituída por um terminal de computador. Por definição o bloco WSF deve incluir o suporte à Interface Homem-Máquina.

Embora os recursos e equipamentos não façam parte da TMN, eles são representados para TMN através da NEF - função elemento de rede. A NEF abrange as funcionalidades gerenciáveis dos equipamentos ou recursos da rede de telecomunicações, devendo estes trocarem informações com a TMN para fins de gerenciamento.

A função adaptador Q representa a conversão de um ponto de referência TMN para um outro ponto de referência não TMN, isto é, a funcionalidade da translação de informações entre um ambiente TMN e outro não TMN.

A função mediação - M representa a funcionalidade da adaptação, armazenamento e filtragem das informações gerenciais trocadas entre os blocos funcionais NEF e/ou QAF – Q Adaptor Function com uma OSF, quando necessário.

Existe ainda a função comunicação de dados - DCF, que representa a facilidade de transporte de informações gerenciais. O transporte das informações de gerência processadas e trocadas entre os blocos da TMN devem ocorrer de forma totalmente transparente. A tarefa básica da DCF é suprir as funções das camadas 1, 2 e 3 do Modelo de Referência OSI para transporte das informações gerenciais da TMN.

Além de possibilitar o acesso do operador às informações de gerência, uma estrutura TMN prevê a troca de informações com outras redes de gerência. A definição efetiva das informações trocadas na TMN utiliza a metodologia apresentada na recomendação M.30200. As recomendações ITU M3010 abordam três aspectos no planejamento e projeto TMN, que são a arquitetura funcional, a arquitetura física e arquitetura de informação. A arquitetura de informação é apresentada a seguir, sendo que as demais fogem ao enfoque deste trabalho:

3.2.1 Arquitetura orientada à objetos

O modelo da arquitetura de informação da TMN é totalmente baseado em princípio de orientação à objetos. Portanto, todas as informações trocadas entre os sistemas de gerenciamento são modeladas em termos de objetos gerenciados. Os objetos representam a visão conceitual dos recursos a serem gerenciados. Portanto, um objeto é uma abstração que representa as propriedades físicas de um recurso, oferecendo a sua “visibilidade” ao sistema de gerenciamento. Um objeto gerenciado pode também representar uma relação entre os recursos ou, ainda, uma combinação destes.

Embora os objetos representem as abstrações dos respectivos recursos físicos gerenciados, isto não implica na obrigatoriedade de um mapeamento direto um para um. Isto é, um recurso real pode ser representado por um objeto ou mais, dependendo somente do ponto de vista abstrato dos recursos físicos quanto às suas múltiplas facilidades dentro do contexto TMN.

3.2.2 Conceito Gerente/Agente

O ambiente de gerenciamento de telecomunicações é basicamente um ambiente de processos aplicativos e informações gerenciais distribuídas. Esta distribuição é uma característica inerente dos recursos gerenciados. Portanto, há uma grande necessidade de troca de informações gerenciais entre processos de gerenciamento e objetos gerenciados.

Em função das necessidades de intercâmbio de dados modelou-se uma estrutura lógica Gerente-Agente, onde o processo que solicita informações de gerenciamento assume o papel do Gerente e aquele que responde à solicitação o do Agente. Assim, o Gerente é a parte da aplicação distribuída que emite diretivas para uma operação de gerenciamento e deve receber as notificações correspondentes. O Agente é a parte do processo aplicativo que gerencia os objetos gerenciados associados, devendo obedecer e responder às diretivas enviadas pelo Gerente. A figura 12 ilustra esta interação.

Para cada processo de gerenciamento, dentro de um contexto de troca de informações, é designado um relacionamento Gerente/Agente. Uma determinada relação Gerente/Agente pode envolver um Gerente e vários Agentes. Neste caso deve-se definir a existência de um relacionamento Gerente/Agente para cada contexto de comunicação. Da mesma forma é possível encontrar o caso oposto, isto é, pode-se apresentar vários Gerentes para um único Agente em um intercâmbio de informações gerenciais.

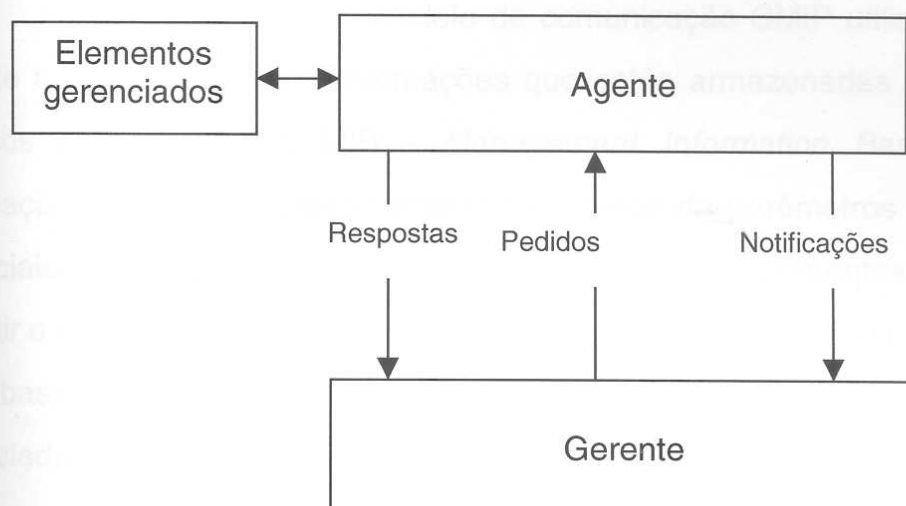


Figura 12 - Modelo Gerente - Agente

Pode acontecer do Agente negar a diretiva de um Gerente no processo de comunicação quando a segurança do sistema estiver em jogo ou a consistência do modelo de informação do sistema for ameaçada. Portanto, o modelo do Gerente é capaz de lidar com este tipo de situação, pois todo intercâmbio de gerenciamento entre o Gerente e o Agente obedece a um conjunto consistente de operações gerenciais. Estas operações são executadas por meio dos Serviços Comuns para Gerenciamento de Informação - CMIS e do Protocolo Comum para Gerenciamento de Informação - CMIP.

A adoção do conceito dos Serviços Comuns para Gerenciamento de Informação e Protocolo Comum para Gerenciamento de Informação permite que em um relacionamento Gerente-Agente, tanto o Gerente quanto o Agente possam compartilhar uma base comum de conhecimentos. Este compartilhamento só é possível quando ambas as entidades adotarem a mesma base de protocolo de comunicação e as mesmas funções e facilidades no seu contexto de comunicação. A Base Compartilhada de Conhecimentos Gerenciais -SMK é fundamental para que se possa ter uma relação Gerente-Agente eficaz.

A figura 13 mostra o modelo de comunicação CMIP utilizado em TMN. O gerente e o agente trocam informações que estão armazenadas na MIB sobre os recursos gerenciados. A MIB - *Management Information Base*, ou base de informação de gerenciamento, representa a base de parâmetros das informações gerenciais sobre a qual os elementos gerenciados e os elementos gerentes podem garantir o mútuo reconhecimento da base de informações da TMN. Estão mapeados nesta base de informações de gerenciamento os parâmetros utilizados pelos objetos gerenciados que os gerentes também possuem conhecimento.

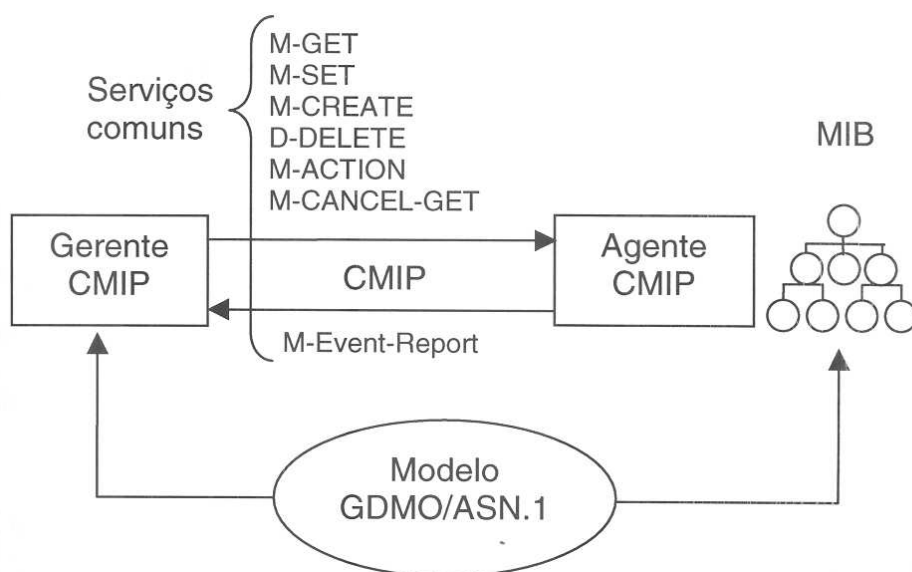


Figura 13 - Modelo de comunicação TMN

O CMIP comporta vários tipos de unidades de protocolos de dados - PDUs que são mapeados em operações equivalentes sobre os objetos gerenciados. Estas PDUs são basicamente as seguintes[5]:

- ♦ M-GET: leitura dos atributos dos objetos gerenciados;
- ♦ M-CANCEL-GET: cancela M-GET
- ♦ M-SET: modificação dos atributos dos objetos gerenciados;
- ♦ M-ACTION: execução de uma ação qualquer sobre um objeto gerenciado;
- ♦ M-CREATE: criação de uma instância de um objeto gerenciado;
- ♦ M-DELETE: remoção de uma instância de um objeto gerenciado;

- ◆ MEVENT-REPORT: emissão de notificação sobre a ocorrência de um evento associado a um objeto gerenciado.

No gerenciamento OSI de redes, os objetos gerenciados são definidos utilizando-se o GDMO – Guidelines for the Definition of Managed Objects. O GDMO é uma ferramenta aplicável na especificação dos objetos e suas operações, comportamentos e notificações[16]. As mensagens do CMIP são especificadas em ASN.1 – Abstract Syntax Notation One. O ASN.1 é projetado para definir informações estruturadas (mensagens) em uma forma independente de máquina[23].

O processo de troca de informações gerenciais é efetivo quando o gerenciador e o gerenciado apresentarem uma visão idêntica das informações trocadas entre si. Isto é, tanto o gerente quanto o agente devem compartilhar uma mesma base de referências para a troca e o processamento das informações gerenciais da TMN.

3.2.3 TMN e o gerenciamento da infra-estrutura

As ferramentas de gerenciamento informatizado de infra-estrutura tiveram o início de seu desenvolvimento atrasados em relação aos demais aplicativos, pois este gerenciamento não se demonstrava fundamental devido a sua simplicidade. Assim sendo, durante o período de elaboração das normas de gerenciamento de redes pela ISO, não foram desenvolvidos trabalhos de tradução dos documentos sugeridos em produtos de software que atendessem os requisitos de gerenciamento para estes tipos de equipamentos.

Muitos esforços estão sendo feitos para a integração dos vários equipamentos à rede de gerência TMN. Entretanto, menos esforços têm sido realizados para o gerenciamento dos sistemas de infra-estrutura. A implementação de interfaces TMN para o gerenciamento da infra-estrutura são difíceis devido a complexidade desta arquitetura.

3.3 O MODELO SNMP – SIMPLE NETWORK MANAGEMENT PROTOCOL

No final dos anos 80, o IAB (*Internet Architecture Board*) formou um grupo para desenvolver ferramentas, protocolos e uma base de dados comum - MIB para gerenciamento de redes[2]. Como resultado nasceu o SNMP - Simple Network Management Protocol para o TCP/IP, com considerações apoiadas no modelo de estruturas OSI. Atualmente as especificações e o código fonte do SNMP estão disponíveis na Internet e foram incorporados em muitos produtos que rodam tanto em *mainframes* quanto em processadores distribuídos de comunicações, como por exemplo concentradores e roteadores.

Embora o SNMP tenha sido originalmente concebido para gerenciar redes TCP/IP, este protocolo é aplicado atualmente no gerenciamento de alguns subsistemas da infra-estrutura[45]. O SNMP, em conjunto com plataformas de gerenciamento do tipo HP-OPEN VIEW, tem sido empregado com relativo sucesso na gerência de equipamentos de suprimento e conversão de energia para telecomunicações. Esta solução é considerada de rápida implementação, enquanto outras soluções mais apropriadas ainda estão em fase de desenvolvimento.

Assim como na TMN, o SNMP utiliza os conceitos básicos de gerente, agente e *Management Information Base* (MIB). O gerente pode ser visto como um cliente que roda na estação de gerenciamento e o agente como um servidor que roda em um elemento de rede, sendo que ambos pertencem à camada de aplicação do modelo OSI. A figura 14 apresenta o modelo de comunicação SNMP.

De forma análoga à arquitetura TMN, o agente SNMP é um programa de aplicação de rede que tem a função de coletar os dados do equipamento a ser gerenciado e interagir com o gerenciador e o sistema operacional de rede. O elemento de rede pode ser um *host*, um *gateway*¹, um servidor de dados, um subsistema de infra-estrutura, ou um equipamento.

¹O *gateway* pode ser implementado por um roteador que execute a conversão de protocolos, possibilitando a integração com outras redes.

Um gerente pode requisitar ao seu agente para reportar dados de seu estado e do seu desempenho a partir de sua MIB. A informação é codificada em uma mensagem SNMP ou em uma cadeia de bits. O gerenciador também fornece uma interface para o usuário, que pode acessar e visualizar a informação, analisando os dados, revisando o estado dos componentes de rede e tomando decisões. O gerenciador é um outro programa de aplicação que tem como funções supervisionar e comunicar com seus agentes. As informações que os usuários estiverem utilizando estão armazenadas no gerente, como por exemplo, estatísticas e interfaces gráficas de usuário, sendo que as definições do sistema podem ser compartilhadas e distribuídas.

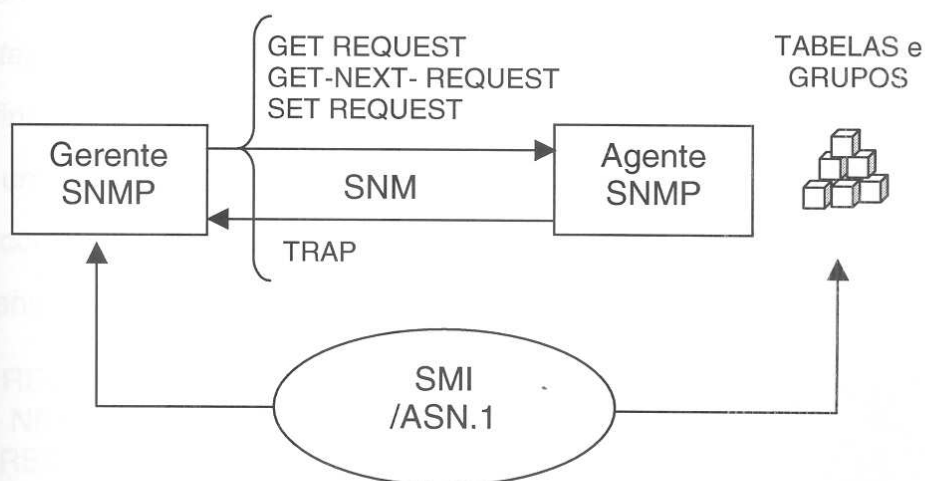


Figura 14 - Modelo de comunicação SNMP

A base de dados criada no elemento de rede é vista como uma MIB - *Management Information Base*, que contém objetos arranjados em uma estrutura de árvore[33]. Cada vez que o gerente necessitar de qualquer informação, ele irá procurá-la a partir da MIB do agente SNMP. O uso de agentes faz o sistema parecer homogêneo, pois o gerente apresenta as informações a partir das MIBs. Como todas as informações de gerenciamento relativas aos equipamentos são salvas na MIB, ela representa o equipamento para efeitos de gerenciamento.

Os agentes SNMP podem ser implementados como agentes do tipo PROXY[45]. O agente proxy pode comunicar-se, ou diretamente com o equipamento

instalado por um protocolo proprietário, ou por intermédio de um modem. Portanto, o agente proxy é um programa especial de aplicação que roda em um computador, em um hardware específico, ou mesmo no equipamento gerenciado caso este tenha a função incorporada e que representa o recurso gerenciado.

As funções de um agente proxy são aplicadas em dispositivos que não suportam SNMP e que devem ser gerenciados por uma estação de gerenciamento. O agente Proxy é uma solução adaptativa que recupera a informação do dispositivo e a traduz para o formato MIB, objetivando efetuar a comunicação com o gerente responsável pelo dispositivo.

As mensagens SNMP possuem formato próprio, definido no ASN.1. Cada mensagem é um subcomando especial transmitido através de uma porta UDP – *User Datagram Protocol*. Em geral, as mensagens de requisição do SNMP são tanto para definir (SET) o valor de parâmetros quanto para ler parâmetros (GET). Após receber uma requisição, o agente pede por serviços de autenticação do sistema. Apenas com aprovação a requisição será atendida. Existem ao todo cinco tipos de mensagens:

- ♦ GET REQUEST
- ♦ GET NEXT REQUEST
- ♦ SET REQUEST
- ♦ GET RESPONSE
- ♦ TRAP

Uma mensagem de GET REQUEST é enviada pela rotina do gerente que deseja ler informações da MIB, sendo que esta mensagem contém uma variável ou uma lista de variáveis. Uma variável é reconhecida como um objeto gerenciado na base de dados. O GET NEXT é uma mensagem de acompanhamento enviado pelo gerente após receber a resposta de uma requisição anterior ou GET REQUEST. Esta mensagem requisita o valor do próximo objeto na seqüência em relação ao objeto em questão. Baseado no endereço do objeto atual, o agente percorre a tabela da MIB e retorna o valor do próximo objeto na seqüência do seu endereço. Se o objeto em questão for o último objeto da tabela, então o próximo objeto será o primeiro objeto da tabela. O SET REQUEST também é enviado pelo gerente para

pedir ao agente para atribuir valores para as variáveis da MIB. Todas as mensagens de requisição são enviadas do gerente para o agente que a executa plenamente, se o acesso for correto e permitido. A mensagem de GET RESPONSE é enviada do agente para o gerente como resposta às mensagens de GET REQUEST, GET NEXT REQUEST ou SET REQUEST. A quinta mensagem, TRAP, não é uma mensagem solicitada. Após a ocorrência de um evento especial no elemento gerenciado o agente envia uma mensagem de TRAP ao seu gerente informando o evento. Esta característica pode ser habilitada por programação se um TRAP for desejável. Eventos especiais de TRAP incluem basicamente as falhas e a autenticação destas. Todas as mensagens de requisição e resposta são transmitidas ou recebidas via chamadas UDP – User Datagram Protocol.

Cada mensagem SNMP é composta de um cabeçalho de mensagem e um PDU – Protocol Data Unit. O cabeçalho da mensagem contém principalmente o número da versão e o nome da comunidade. O nome do objeto pode ser encontrado na estrutura de gerenciamento da informação (SMI).

3.3.1 SNMP e o gerenciamento da infra-estrutura

O uso de um *framework* básico, baseado em SNMP, pode reduzir o tempo de implementação dos sistemas de gerenciamento infra-estrutura de telecomunicações quando comparado com TMN e têm se mostrado como tecnologia candidata para a aplicação. Observa-se entretanto, que concepções adequadas das MIBs são fundamentais para o sucesso das implementações, pois este é um trabalho demorado e cheio de detalhes. Por mais que se façam estudos preliminares, sempre aparecem novos requisitos após a implementação.

A inclusão de novos requisitos, ou seja a ampliação das MIBs, não é uma tarefa trivial, pois é necessário manter a compatibilidade com o sistema anterior. Em sistemas muito amplos torna-se necessária a formação de grupos de trabalhos permanentes para conduzir a manutenção do sistema. Por outro lado, a estratégia de implementar MIBs extensas, visando aplicações que supostamente surgiriam no

futuro, parece não ser interessante, pois aumenta-se o tempo de implementação, sendo que a utilização dos objetos no futuro não é garantida[45].

Uma solução que utiliza SNMP foi implementada na Telia, a operadora de telecomunicações da Suécia, Noruega e Dinamarca. O sistema de gerenciamento de energia foi construído em UNIX, Open View e SNMP. Constatou-se que várias alterações foram necessárias na MIB, porque não foi possível prever todas as necessidades no início do projeto. A própria operadora afirma que teria sido melhor projetar uma MIB mais simples e com menos sinais, para então ampliá-la de acordo com as necessidades futuras. Quando o sistema foi posto em operação surgiram várias necessidades novas.

Fica claro que uma desvantagem óbvia na utilização do protocolo SNMP é que a sua utilização de forma adaptativa pode comprometer a interoperabilidade. Ou seja, cada fabricante adiciona seu próprio código, modificando as MIBs em seu conteúdo e estrutura. Isso significa que mesmo em uma aplicação utilizando dois recursos de diferentes fabricantes, ambos com SNMP compatíveis, não existe a garantia que eles possam ser gerenciados a partir de um mesmo gerenciador[33].

3.4 TECNOLOGIAS DE SISTEMAS DISTRIBUÍDOS

A computação distribuída pode ser definida como duas ou mais partes de software compartilhando informações entre si, podendo estas duas partes estarem rodando em uma mesma máquina ou em máquinas diferentes conectadas em rede[8]. Um sistema de computação distribuída possui um processo cliente e um processo servidor sendo executados em computadores que comunicam entre si utilizando algum protocolo de rede. Em sistemas distribuídos, os clientes pedem para os servidores executarem algumas funções, caracterizando assim o modelo cliente-servidor. A exemplo do modelo gerente-agente, o modelo cliente-servidor pode ser utilizado para o gerenciamento de sistemas de telecomunicações[29].

Estima-se que **aplicações distribuídas** irão interagir com os recursos dos sistemas de telecomunicações, seja ele legado ou recentemente introduzido, sendo que a maioria dos recursos estarão dispersos através das redes de serviços e

recursos cooperantes[38]. Vários consórcios de empresas e grupos que estabelecem normas estão desenvolvendo a integração de processamento distribuído e o gerenciamento das telecomunicações, como por exemplo o ITU – *International Telecommunication Union*, o IETF – *Internet Engineering Task Force*, o Open Group, a OMG, o NMF – *Network Management Forum* e o TINA C – *Telecommunication Information Networking Architecture Consortium*.

Uma das características primárias de sistemas distribuídos é a abertura e a escalabilidade, lembrando que estes são requisitos fundamentais para o gerenciamento de infra-estrutura de telecomunicações. Portanto, identifica-se que características inerentes de sistemas distribuídos de computação facilitam a implementação de sistemas de gerenciamento de infra-estrutura. Este gerenciamento exige que vários dados trafeguem de um computador para outro.

Apesar da transferência de dados ser uma operação computacional básica, é importante analisar as várias formas de realizar esta operação. O modelo gerente-agente apresentado anteriormente proporciona basicamente um modelo de comunicação síncrona. Tecnologias de sistemas distribuídos baseados no modelo cliente-servidor proporcionam algumas facilidades extras, como por exemplo a comunicação assíncrona. A comunicação síncrona é quando uma parte do software envia uma mensagem para uma outra parte do software e então espera pela resposta. A comunicação assíncrona é quando uma parte do software envia uma mensagem para uma outra parte de software e continua a operar, esperando que a resposta retorne em algum instante.

Entre os métodos de comunicação entre processos (IPC – *Inter Process Communication*) estão as tecnologias de sockets, RPC – *Remote Procedure Call* e RMI – *Remote Method Invocation*. Sistemas de gerenciamento baseados nestas tecnologias são possíveis e muitas implementações atuais são baseadas em algumas destas tecnologias, como por exemplo o sistema SIS[27] que baseia-se nos serviços de comunicação RPC.

Arquiteturas de sistemas distribuídos como DCE, D-COM e CORBA permitem aos implementadores trabalharem em níveis de abstração mais elevados

e oferecem diversos serviços pré-estabelecidos, liberando o implementador para concentrar-se na aplicação propriamente dita. Uma das tecnologias de sistemas distribuídos que apresentam maior grau de abertura atualmente, sendo orientado à objetos, é CORBA. Todas as especificações CORBA podem ser encontradas na Internet no endereço: <http://www.omg.org/schedule/>.

A seguir são descritas resumidamente as tecnologias para implementação de sistemas distribuídos:

3.4.1 Sockets

Uma das formas utilizadas para transferir mensagens entre diferentes computadores é por intermédio de sockets. Os sockets permitem a transferência de mensagens sobre uma rede TCP/IP, também chamados de datagramas. Podem ser utilizados dois tipos de sockets: sockets TCP (Transmission Control Protocol) ou sockets UDP (User Datagram Protocol). Normalmente os sockets TCP são mais utilizados dos que os UDP.

A transferência de mensagens através de sockets é mais rápida e consome menos memória em comparação com tecnologias como CORBA. Entretanto, é quase sempre mais trabalhoso programar com sockets do que com linguagens de mais alto nível. Se na infra-estrutura de telecomunicações as estruturas de dados fossem mais simples, talvez o uso de sockets seria uma boa escolha. Sockets se aplicariam bem no gerenciamento de falhas, onde os sinais são sinais rápidos e apresentam poucos detalhes.

3.4.2 Chamada de procedimentos remotos: RPC

Uma das técnicas mais difundidas em sistemas distribuídos é o emprego do RPC - *Remote Procedure Call*, sendo que existem alguns esforços no sentido de estabelecimento de mapeamentos entre RPC/IDL e CMIP/GDMO, visando a utilização de RPC como opção de gerenciamento distribuído de telecomunicações[10].

Como sockets, o RPC tem como objetivo a transferência de dados entre computadores. Como o próprio nome sugere, RPC trabalha com a chamada de procedimentos remotos, um mecanismo de programas não orientados a objetos[39].

Como exemplo de aplicação prática, Nogueira, Andrade e Menezes[27] descreveram o sistema SIS – Sistema Integrado de Supervisão como um sistema distribuído de gerenciamento, cujo maior objetivo é a integração à sua plataforma, de sistemas de supervisão e gerência com alto grau de heterogeneidade funcional e tecnológica. Uma das interfaces do SIS baseia-se nos serviços RPC – Remote Procedure Call.

As principais características do RPC são:

- ♦ Procedimentos são de natureza síncrona, significando que a aplicação que fez a requisição deve esperar até que o servidor possa realizar a tarefa.
- ♦ As aplicações que utilizam RPC normalmente preocupam-se com alguns aspectos de transporte em rede e são responsáveis em localizar o servidor que possa satisfazer a requisição.
- ♦ RPC geralmente exige que o servidor esteja ativo quando a requisição é iniciada, delimitando o local onde a requisição pode ser servida.
- ♦ DCE-RPC (Distributed Computer Environment - RPC) permite a especificação de múltiplos servidores, mas é o programador quem deve descobrir cada servidor e escolher um entre eles.
- ♦ DCE RPC não possui um mecanismo intermediário para manipular encontros automáticos e realizar o roteamento para os servidores.

Tanto o paradigma de sockets e quanto o de chamada de procedimentos remotos possibilitam a transferência de dados. Entretanto, o RPC é de implementação mais simples do que sockets, pois abstrai muitos dos detalhes de comunicação. Paradigmas que tem como precursor o RPC e que permitem trabalhar em um nível de abstração mais elevado ainda são o RMI – Remote Method Invocation e CORBA[9].

3.4.3 RMI – Remote Method Invocation

RMI melhora os mecanismos RPC utilizando os recursos de serialização do JAVA[18]. Assim, qualquer objeto que implemente a interface *Serializable* pode ser enviado pelo RMI como um argumento de um método ou retornar um valor.

O RMI também fornece um registro de serviços que roda em um servidor. Este registro de serviços mantém um banco de dados com o nome dos objetos remotos. Quando um cliente deseja utilizar um objeto remoto, ele procura pelo nome a referência deste objeto. Uma vez que a referência seja obtida, pode-se utilizar o objeto como se ele fosse local. RMI utiliza os conceitos de Stub e Esqueleto que serão abordados mais adiante. Embora RMI atualmente seja uma tecnologia aplicável somente baseado em JAVA, já existem trabalhos de cooperação entre a Sun e a IBM para permitir que RMI e CORBA operem em conjunto.

3.4.4 D-COM

D-COM é uma evolução da tecnologia de componentes Microsoft Active X. D-COM permite a criação de objetos que possam ser acessados remotamente por programas Visual Basic, C e C++[40]. É possível escrever programas em Visual J++ e Microsoft IDE para acessar objetos DCOM. Apesar de haverem esforços para aumentar a portabilidade da arquitetura D-COM para UNIX, atualmente D-COM opera somente em plataformas Microsoft.

3.4.5 CORBA

Talvez a proposta mais promissora para a computação distribuída de objetos seja a da OMG - Object Management Group, um consórcio apoiado em mais de 800 empresas membros. A OMG é uma organização sem fins lucrativos fundada em 1989 que se dedica em divulgar os padrões de orientação a objetos. A OMG tem definido um conjunto de especificações chamado de Object Management Architecture (OMA) para a computação distribuída. Em dezembro de 1990 foi publicado um documento pela OMG denominado de *Object Management*

Architecture Guide. A plataforma para as aplicações é o Common Object Request Broker Architecture - CORBA e seus serviços adicionais[3].

A versão 1.1 de CORBA, publicada em dezembro de 1991, descreve como desenvolver uma aplicação. A versão 1.1 foi desenvolvida em conjunto com a Digital Equipment Corporation, Hewlett-Packard Company, HyperDesk Corporation, NCR Corporation, Object Design, Inc, e SunSoft, Inc., sendo revisado e aceito por todo o OMG.

Embora as tecnologias apresentadas anteriormente possam ser utilizadas no gerenciamento distribuído da infra-estrutura de telecomunicações, optou-se aqui pela abordagem mais aprofundada de CORBA devido as seguintes razões:

- ♦ CORBA combina os conceitos de reutilização de software, através das técnicas de orientação a objetos, com conceitos computação distribuída. Cada parte não somente adiciona as suas qualidades como também complementa as qualidades do outro[28].
- ♦ CORBA permite que as aplicações acessem e compartilhem os objetos entre si, tornando estes objetos comuns a todas as aplicações que implementem CORBA.
- ♦ Existem muitos esforços para permitir a integração entre CORBA, SNMP e CMIP[6][7][15][16]. Muitas empresas (IBM, AT&T, Telefonica de Espanha, SMILE, BNR, CiTR, HP, etc) e organizações (OMG, X/open, The Open Group e NMF) estão trabalhando em padrões que permitam um gerenciamento interdomínio, ou JIDM – *Joint Inter Domain Management*[30]. As atividades compreendem um completo mapeamento CORBA-CMIP, CMIP-CORBA, CORBA-SNMP e SNMP-CORBA. Isso amplia a expectativa da aplicação de CORBA no gerenciamento da infra-estrutura. A figura 15[30] resume este panorama.

Um equipamento de telecomunicações pode ser gerenciado por um agente CORBA que se comunica com um gerente CMIP por intermédio de um mapeamento CORBA-CMIP. Em seu artigo, Integrando CORBA e Ambientes TMN, Kong e Chen abordam três modelos para a integração dos ambientes e salienta que, “enquanto que o gerenciamento de rede OSI permanece a principal tecnologia para a

construção de sistemas e aplicações de gerenciamento de baseado em TMN, CORBA é uma tecnologia útil para adicionar valor ao gerenciamento TMN, pois oferece as facilidades para a construção de aplicações distribuídas de gerenciamento como escalabilidade, confiabilidade e melhor desempenho. Portanto as várias tecnologias podem se complementar na implementação de sistemas de gerenciamento de infra-estrutura de telecomunicações.

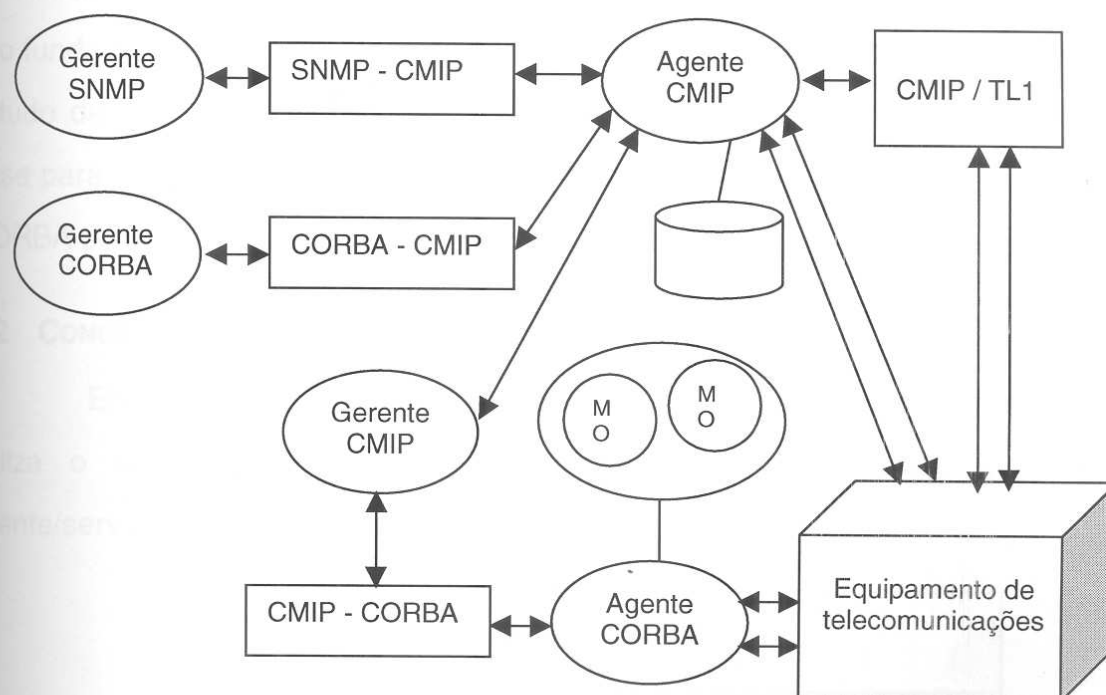


Figura 15 - Integração de CORBA com TMN e SNMP

4 A ARQUITETURA CORBA E SERVIÇOS APLICÁVEIS AO GERENCIAMENTO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES

4.1 INTRODUÇÃO

Este capítulo apresenta a arquitetura CORBA e identifica os serviços definidos no COSS – *Common Object Service Specification*, aplicáveis em sistemas de gerenciamento da infra-estrutura de telecomunicações. Veremos que modelo CORBA fornece muitos benefícios que facilitam a integração de aplicações de gerenciamento em um modelo distribuído. As informações apresentadas a seguir são fundamentais para a compreensão da proposta estabelecida no capítulo 5 e do estudo de caso desenvolvido no capítulo 6. A intenção deste capítulo é servir de base para os próximos capítulos, sendo que as pessoas que dominam a tecnologia CORBA podem passar direto para o capítulo 5.

4.2 CONCEITOS DE CLIENTE, SERVIDOR E MIDDLEWARE

Enquanto TMN e SNMP utilizam o conceito de agente/gerente, CORBA utiliza o conceito de cliente/servidor. A figura 16 exemplifica a interação cliente/servidor através de um middleware.

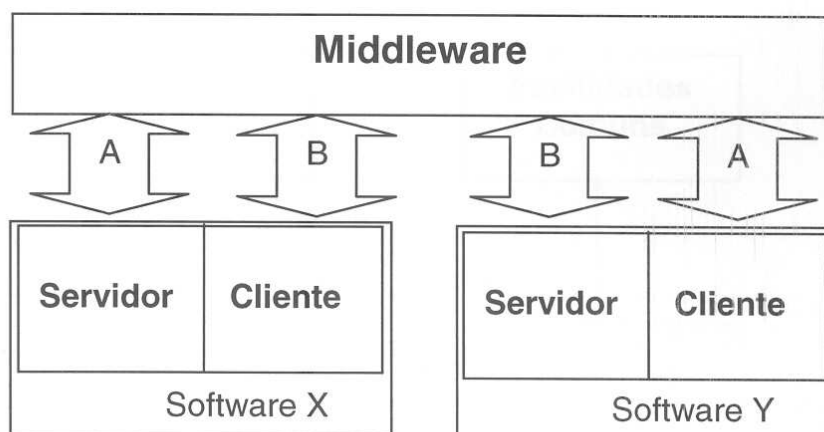


Figura 16 - Interação entre cliente e servidor por intermédio de middleware

Uma determinada porção de software X pode em uma interação A assumir o papel do servidor perante um outro pedaço de software Y que atua como cliente. Nesta interação o cliente Y invoca métodos que são executados pelo servidor X. Em uma outra interação B os papéis podem se inverter, ou seja, a porção X opera como cliente e o software Y como servidor.

O Middleware é um software que reside entre a aplicação e as operações internas do sistema da aplicação, isolando-a dos detalhes de baixo nível e das complexidades que suportam os sistemas[30]. Assim sendo, o desenvolvedor se preocupa apenas com a aplicação propriamente dita, ao invés de codificar para o sistema operacional ou interfaces de baixo nível. CORBA utiliza o conceito de middleware, pois isola a aplicação dos detalhes da comunicação.

4.3 A ARQUITETURA CORBA – OMA

A arquitetura CORBA foi publicada em 1990 no *Object Management Group Guide* como sendo orientada à objetos. Esta arquitetura foi denominada de OMA - *Object Management Architecture (OMA)*, que é a base do CORBA. As quatro partes que fazem parte do OMA estão apresentadas na figura 17.

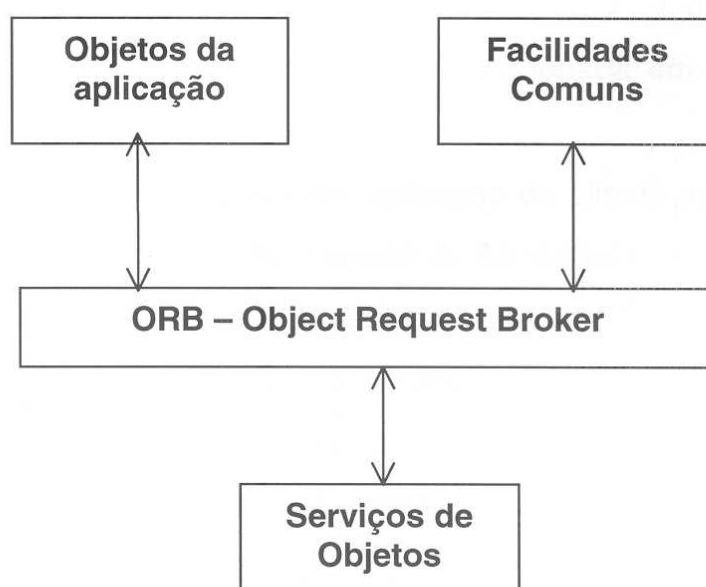


Figura 17 - OMA - *Object Management Architecture*

Em CORBA o sistema intermediário, ou middleware, que realiza o mapeamento dos requisitos de serviços do cliente para uma implementação de servidor é chamado de Broker – *Basic Object Request Broker*. A especificação CORBA define a arquitetura de um ORB, cujo papel é habilitar e regular a interoperabilidade entre objetos e aplicações[3]. O ORB é o componente a partir do qual CORBA derivou o seu nome, pois é o centro de comunicação para todos os objetos do sistema distribuído e foi o primeiro componente especificado pela OMG a ser implementado.

O Broker ou ORB possibilita um acoplamento fraco entre clientes e servidores, gerenciando a interação entre objetos. Este gerenciamento inclui todas as responsabilidades de um sistema distribuído, desde a localização e referência dos objetos, até as ações de *marshaling*² dos parâmetros requisitados e resultados. Assim sendo, apenas ele precisa saber a localização e as capacidades dos clientes e servidores CORBA na rede, sendo que múltiplos servidores podem trabalhar com um único cliente ou um único servidor pode operar com múltiplos clientes.

O ORB utiliza informações da requisição para determinar a melhor implementação que pode satisfazê-la. Esta informação inclui a operação que o cliente está requisitando, o tipo de objeto sobre o qual a operação está sendo executada, e qualquer informação adicional armazenada em objetos de contexto para a requisição.

A figura 18 mostra como uma aplicação do cliente invoca um método de uma implementação A.1 alocado no servidor A. As implementações são obviamente sistemas compostos por objetos possuidores de interfaces que possibilitam a comunicação através do ORB. A definição destas interfaces serão abordadas mais adiante no final deste capítulo. Note que uma outra implementação A.1 está disponível em um outro servidor B. Neste caso, o ORB procura realizar o balanceamento de carga dos servidores, ou seja, intercalar chamadas consecutivas entre os servidores A e B.

² *Marshaling* refere-se ao processo de tradução e transferência de parâmetros e resultados entre máquinas, processos e espaços de endereços.

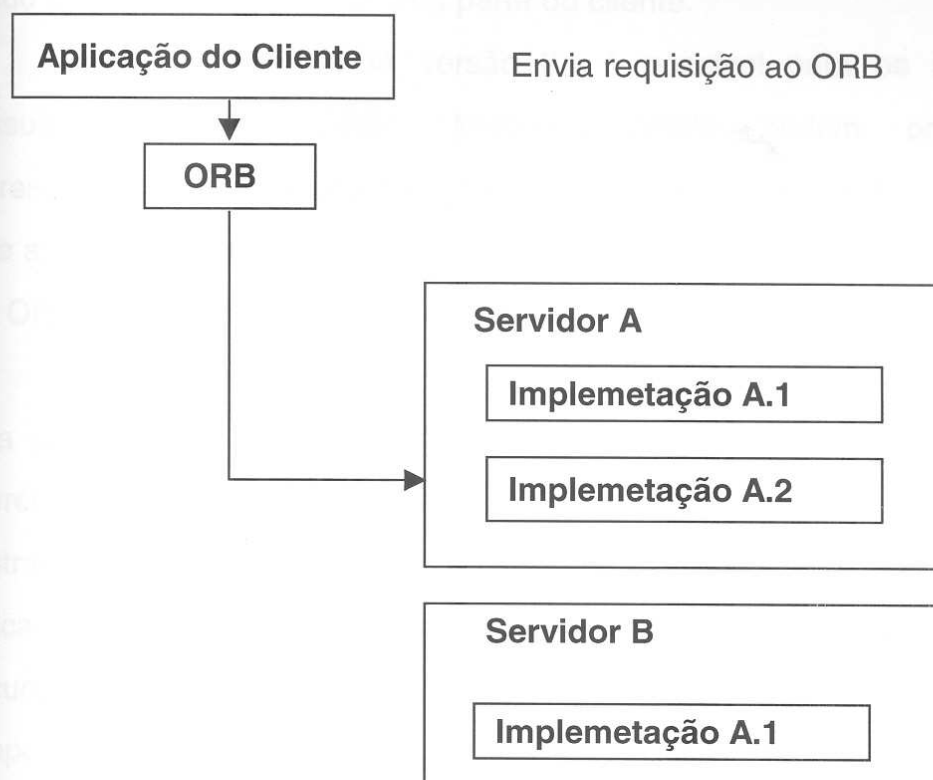


Figura 18 - Seleção de implementação pelo ORB

Além de selecionar a implementação para realização da requisição, o ORB também valida cada requisição e seus argumentos, e pode fornecer informação de autenticação e autorização de informação. Por exemplo, o ORB verifica se o objeto especificado na requisição é válido para a operação especificada. O ORB também procura no repositório de implementação um objeto que possa servir a requisição do cliente em invocações dinâmicas.

Apesar do ORB ser um componente singular, ele possui algumas funções específicas para o lado do cliente e outras funções específicas para o lado do servidor. No lado do cliente o ORB manipula pedidos de invocação e a seleção de servidores e métodos. Quando uma aplicação envia um pedido para o ORB, visando uma operação em um objeto, o ORB valida os argumentos a partir da interface e

despacha a requisição para o servidor, inicializando-o se necessário. O ORB é ligado à aplicação e constitui uma parte do cliente.

A partir de CORBA versão 2 é possível para os sistemas CORBA possuírem diversos ORBs. Múltiplos ORBs podem possuir diferentes representações para referências de objetos e diferentes estilos de invocação. Se uma aplicação do cliente possuir acesso a duas referências de objetos gerenciados por ORBs diferentes, os ORBs devem distinguir as referências dos objetos[43].

As facilidades comuns não fazem parte da especificação CORBA, mas de uma parte complementar ao OMA, adicionando funcionalidades. As facilidades diferem dos serviços CORBA por especificarem funcionalidades em níveis de abstração mais elevados. As facilidades comuns fornecem um conjunto de aplicações de uso geral para diferentes aplicações, como gerenciamento de documentos, acesso a base de dados, impressão de arquivos, ou sincronização de tempo em um ambiente distribuído.

Os serviços CORBA também não fazem parte do CORBA, mas eles são uma parte complementar da arquitetura OMA que o OMG define. Os serviços de objetos fornecem um conjunto de serviços em tempo de execução que auxilia na criação de aplicações robustas orientadas à objetos. O uso dos serviços de objetos pode facilitar a criação das aplicações. Os serviços de objetos permitem ao desenvolvedor chamarem funções previamente implementadas ao invés de escreverem e chamarem funções particulares de serviços. Todas as empresas, que desejarem implementar serviços de objetos, devem estar de acordo com as especificações da OMG[13]. Isso permite trocar de fornecedor de ORBs sem ter que alterar a implementação.

Os serviços estão especificados no COSS[28], ou *Common Object Services Specification*. São definidos alguns serviços como: serviço de nomes, para localização de objetos; serviço de eventos, para transmitir eventos entre objetos; serviço de ciclo de vida, para controlar a criação dos objetos; serviço de persistência, para armazenar os objetos. No final deste capítulo são identificados os serviços que podem facilitar a implementação de sistemas de gerenciamento de

infra-estrutura de telecomunicações e no anexo A alguns outros serviços especificados no COSS que podem ser úteis para implementações comerciais.

Objetos da aplicação fornecem um conjunto de objetos que realizam tarefas específicas para os usuários CORBA. Estas aplicações são essencialmente orientadas à objetos. Pode-se construir aplicações utilizando objetos mais simples, alguns para aplicações específicas e outras a partir das facilidades comuns. São os objetos da aplicação que implementam o sistema de gerenciamento de infra-estrutura.

4.4 COMPONENTES E INTERFACES DA ARQUITETURA CORBA

Em muitos sistemas distribuídos os clientes e os servidores são fortemente acoplados, sendo que cada parte possui muita informação sobre a tarefa do outro. Inicialmente, isso facilita a implementação, porém a necessidade de alterações futuras exigirá a modificação das partes. A infra-estrutura de telecomunicações é um sistema que tende a mudanças contínuas, visando a ampliação, modernização e otimização dos recursos.

Na arquitetura CORBA, o cliente e o servidor estão formalmente separados, sendo que o cliente CORBA sabe apenas como invocar um método e o servidor CORBA sabe apenas como cumprir a tarefa. Isso significa que pode-se modificar a forma como o servidor executa a tarefa sem afetar a forma como o cliente pede ao servidor para realizar a tarefa. Portanto, essa característica de CORBA promove a flexibilidade tão almejada para a implementação de sistemas de gerenciamento da infra-estrutura de telecomunicações.

A figura 19 procura resumir a arquitetura de um sistema CORBA, mostrando a interconexão dos seus componentes[28]. Embora esta figura ilustre sistemas de computadores que separam o cliente e o servidor, todos os componentes poderiam estar em uma mesma máquina.

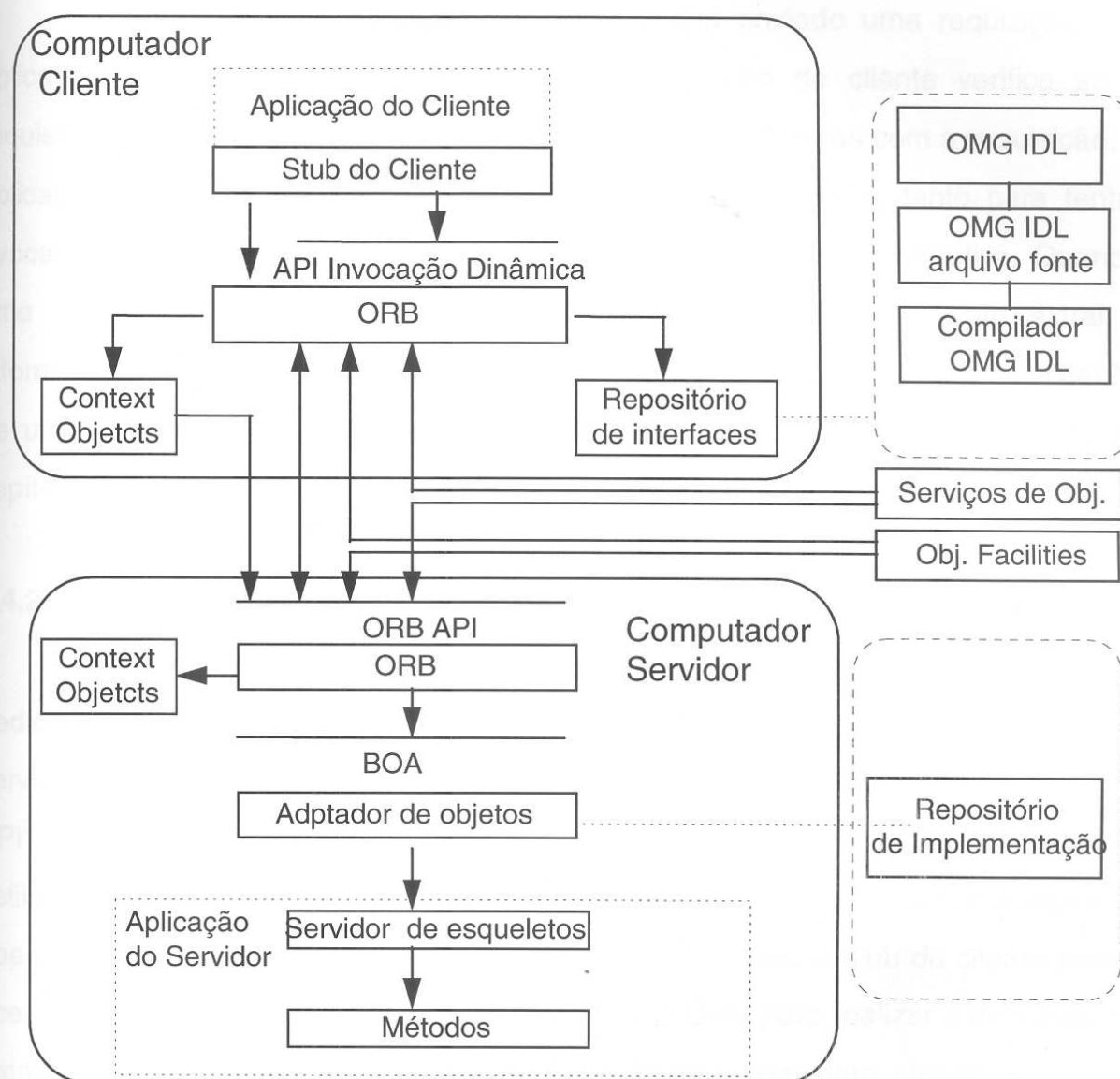


Figura 19 – Componentes e interfaces CORBA

A seguir são descritos os principais componentes que constituem um sistema CORBA:

4.4.1 Aplicação do cliente

A aplicação do cliente utiliza o sistema CORBA para evocar operações sobre algum servidor. Ela envia um pedido para um servidor através de evocação do estilo stub, dinâmica, ou combinação dos dois tipos. O pedido resulta em operações sobre objetos através de interfaces específicas. Cada tipo de evocação possui diferentes características e são utilizadas de forma diferenciada, embora o servidor não possa dizer se ele foi evocado através de stubs ou dinamicamente.

Uma vez que a aplicação do cliente tenha enviado uma requisição e a aplicação do servidor tenha respondido, a aplicação do cliente verifica se a requisição foi completada com sucesso. Se existirem problemas com a requisição, a aplicação do cliente pode recuperar a informação de exceção, tanto para tentar evocar o método novamente, quanto para realizar alguma ação corretiva. Quando uma requisição é completada com sucesso, a aplicação do cliente extrai a informação a partir dos argumentos da requisição. O estilo utilizado para a evocação de um método é irrelevante para o servidor. O estudo de caso desenvolvido no capítulo 6 desenvolve os dois tipos de evocações.

4.4.2 Evocação estilo stub

Este tipo de evocação é uma das duas formas possíveis para enviar um pedido. Uma aplicação do cliente pode enviar requisições para uma aplicação do servidor através do stub do cliente, gerado a partir de um compilador OMG IDL. Esta API³ especifica um determinado tipo de operação sobre um objeto. Na evocação do estilo stub a aplicação do cliente liga-se aos stubs do cliente que representam as operações definidas para uma ou mais interfaces. Ou seja, o stub do cliente possui acesso às referências de objetos e interage com o ORB para realizar a evocação de uma requisição. A evocação do tipo stub se parece com uma chamada de uma função, sendo de mais fácil utilização do que a evocação dinâmica. CORBA suporta apenas comunicação síncrona para a evocação de estilo stub.

Os stubs representam mapeamentos entre a linguagem do cliente e a implementação do ORB, permitindo o uso dos recursos por qualquer cliente que possua o mapeamento. Stubs são mecanismos proprietários otimizados para realizar chamadas ao ORB. ORBs de diferentes fornecedores podem ter stubs de cliente diferentes. Os stubs de cliente devem ser tratados como um componente que é compilado e ligado à aplicação.

³ API - Application Programming Interface - rotinas usadas para requisitar ações de baixo nível ao sistema operacional, muito usado no desenvolvimento de aplicações.

4.4.3 Evocação dinâmica

A evocação dinâmica é uma das duas formas de enviar uma requisição. A evocação dinâmica exige que a aplicação do cliente utilize operações para acessar o repositório de interfaces e construa uma requisição em tempo de execução. Para a evocação dinâmica, CORBA suporta tanto comunicação síncrona quanto comunicação assíncrona do tipo *deferred*[42].

Um programa para evocação dinâmica possui as seguintes rotinas:

- ◆ Rotinas de requisição que adicionam um argumento à requisição, chamam o ORB para evocar o método apropriado, ou apagam uma requisição.
- ◆ Rotinas assíncronas do tipo *deferred* enviam mais de uma requisição em paralelo, determinam se uma requisição está completada e verifica a próxima requisição.
- ◆ Lista rotinas, como as utilizadas para alocar uma lista e limpá-la para uso inicial, liberando a estrutura de lista e memória associada.

4.4.4 Objeto de contexto

A interface de contexto contém uma lista de propriedades de um cliente. Esta lista de propriedades é propagada para o servidor quando um cliente faz uma requisição. Os objetos de contexto possuem informações nos clientes, ambientes, ou em um pedido que não é passado como argumento formal de uma operação.

CORBA transmite informações de objetos de contexto do cliente para o servidor, e potencialmente para outros servidores, para suportar um conjunto de operações distribuídas. Desenvolvedores de aplicações podem utilizar informações de objeto de contexto para vários propósitos, incluindo a escolha do método, servidor, perfil de usuário, grupo de usuário e preferências gerais. A especificação CORBA não define o conteúdo de um contexto, sendo que o uso destas propriedades são definidas pelo usuário e implementador.

Os objetos de contexto podem ser temporários ou persistentes e podem ter um escopo associado. Assim, os desenvolvedores podem utilizar este escopo para

agrupar diferentes níveis de atributos em uma dada plataforma, fornecer os valores iniciais e sobrepor os comportamentos destes atributos.

O ORB pode incluir no objeto de contexto suas preferências de requisição para as aplicações do cliente. Como o objeto de contexto é passado como um argumento implícito, o ORB pode utilizar a informação no objeto de contexto para selecionar uma implementação, influenciando o comportamento do método, a localização ou política de ativação dos servidores.

O ORB irá procurar por um objeto de contexto apenas se um dos seguintes itens for verdadeiro:

- ♦ Existir uma cláusula de contexto no OMG IDL definindo a operação.
- ♦ A aplicação do cliente estiver utilizando a interface de evocação dinâmica, a qual irá requerer o uso de objeto de contexto, mesmo se o objeto de contexto for nulo.

CORBA especifica que a informação do objeto de contexto é opcional, de forma que o ORB não depende da informação fornecida. Entretanto, requisitos podem conter um argumento que referencia os objetos de contexto.

4.4.5 Repositórios de interface

O repositório de interfaces armazena módulos de informação sobre interfaces, como as informações codificadas na interface IDL, constantes, *typedefs*, e assim por diante. O repositório de interfaces inclui descrições de operações que são válidas para um dado objeto e os argumentos que são válidos para as operação. Ou seja, o repositório de interface contém descrições de interface para aplicações e dados sobre os objetos que existem na rede.

A função primária do repositório de interfaces é fornecer informação para requisições evocadas dinamicamente. A aplicação do cliente utiliza o repositório de interfaces durante a evocação dinâmica para obter a assinatura da operação que estiver sendo requisitada, incluindo qualquer tipo definido pelo usuário. A assinatura é um termo de programação geral para os argumentos associados, com uma rotina ou operação. Em CORBA, a assinatura para uma operação inclui os argumentos que são parte de uma operação específica, seus tipos, direção, e tipos retornados.

O repositório de interfaces contém o mesmo tipo de informação de evocação que uma implementação CORBA deve armazenar nos seus stubs de cliente para uma invocação do estilo stub. A aplicação do cliente acessa o repositório de interfaces como parte do processo de evocação dinâmica, utilizando as operações que CORBA define para estes propósitos.

Os desenvolvedores de aplicações CORBA devem gerenciar a instalação e a distribuição das definições de interface. Pelo armazenamento das definições no repositório de interfaces, os desenvolvedores podem atualizar estas ao invés de cada arquivo OMG IDL utilizado. Isso evita a recompilação de todo o sistema cada vez que alterações sejam necessárias.

Caso a definição da interface possua definições herdadas ou referenciadas (como interfaces, operações, ou definições de tipos de dados), os desenvolvedores podem disponibilizar estes arquivos herdados durante a geração de esqueletos ou stubs de clientes, pelo armazenamento dos mesmos no repositório de interfaces. Isso torna desnecessário a distribuição de arquivos OMG IDL. Outra possibilidade é o vendedor do sistema CORBA fornecer um browser que permita aos desenvolvedores pesquisarem o repositório de interfaces. Isso possibilita que o desenvolvedor encontre a definição de interfaces para reuso no seu próprio arquivo IDL, ou para utilizá-las na escrita das aplicações de empreguem a definição de uma interface para um servidor. Essa abordagem pode facilitar a implementação de sistemas de gerenciamento de infra-estrutura abertos, pois as interfaces ficam registradas e disponíveis aos desenvolvedores.

Apesar do repositório de interfaces ser uma parte integral do sistema CORBA, não é especificado como a leitura da informação deve ser implementada a partir do repositório de interfaces e como construir ou carregar o repositório. CORBA delega estes detalhes aos implementadores dos sistemas CORBA. Portanto, repositórios de interface de diferentes fornecedores CORBA serão diferentes em:

- ♦ Características de controle de versão, concorrência, e verificação de integridade.

- ♦ Flexibilidade e distribuição de capacidades, como repositório único e centralizado para todas as aplicações e usuários do sistema CORBA, ou um repositório de interface para cada computador e usuário, ou alguma combinação de ambos.
- ♦ Ferramentas e características que cada fornecedor oferece para o uso do repositório de interfaces deve ser importante para a seleção do sistema CORBA.

Assim sendo, pelo uso do repositório de interfaces é possível para um cliente localizar um objeto que não era conhecido durante o tempo de compilação, questionar os detalhes de sua interface, e então formular um pedido para ser enviado através do ORB. Isso oferece um nível de abertura muito interessante para os implementadores de sistemas de gerenciamento de infra-estrutura, pois as interfaces são acessadas e visualizadas através do repositório de interfaces.

4.4.6 Repositórios de implementação

O repositório de implementação é um local de armazenamento para as definições de implementação, como os dados sobre quais informações estão instaladas em um dado sistema. CORBA utiliza o repositório de implementações para associar referências de objetos com implementações, assim como ele usa o repositório de interface para associar referências de objetos com suas interfaces.

O repositório de implementação contém informações que permitem ao ORB localizar e ativar objetos que atendem às requisições dinâmicas. Adicionalmente às informações de implementação, este repositório contém dados para depuração, administração, versão de software, dados administrativos e assim por diante. A especificação CORBA não define a implementação dos repositórios. Portanto diferentes vendedores podem apresentar soluções diferenciadas destes recursos. A maioria das informações em um repositório de implementação é específico dos vendedores de sistemas CORBA ou ambientes operacionais. Desenvolvedores ou administradores podem utilizar operações sobre repositórios de implementações para instalarem implementações e políticas de controle relacionadas com ativação e execução de implementações.

A especificação CORBA não fala muito sobre o repositório de implementações. Entretanto, os vendedores CORBA devem determinar quais operações são utilizadas para acessarem os repositórios, como construí-los, e quaisquer capacidades adicionais, como por exemplo, informações para depuração das aplicações ou para administração do servidor.

4.4.7 Aplicação do Servidor

A aplicação do servidor utiliza o lado do servidor do ORB e contém uma ou mais implementações de objetos, incluindo os seus respectivos métodos. O ORB possui algumas funções específicas para o lado servidor. O ORB é o componente CORBA primário no lado do servidor que recebe as requisições de métodos enviadas, realiza o *unmarshaling* dos argumentos, ajusta do estado do contexto de acordo com a necessidade, evoca o despachante de métodos, realiza o *marshaling* dos argumentos de saída, e completa a evocação.

Quando um cliente envia uma requisição para o ORB, este seleciona uma implementação para satisfazer a requisição e utiliza o BOA para direcionar a requisição para a implementação. O BOA usa o esqueleto do servidor para chamar os métodos que tenham a implementação.

A aplicação do servidor precisa conter códigos para executarem os seguintes passos:

- ◆ inicializar o servidor
- ◆ manipular requisições de serviços
- ◆ desligar a aplicação do servidor

4.4.8 Basic Object Adapter - BOA

O adaptador de objetos é o mecanismo primário para o gerenciamento de referências de objetos e implementações no servidor, podendo realizar as seguintes funções:

- ◆ Providenciar o *binding* entre uma interface de objeto e um objeto servidor de implementação;

- ♦ Gerar e interpretar referências de objetos e mapeá-las para os objetos;
- ♦ Ativar e desativar objetos e implementações;
- ♦ Registrar as implementações;
- ♦ Invocar implicitamente métodos de implementação através dos esqueletos.

CORBA projetou o *Basic Object Adapter* (BOA) para ser o adaptador de objetos mais utilizado. Todos os fornecedores de sistemas Corba devem fornecer um BOA como parte do seu sistema, embora alguns fornecedores possam providenciar adicionalmente adaptadores de objetos especiais, como objetos adaptadores de bibliotecas ou adaptadores de base de dados orientado à objetos.

Na maioria dos sistemas CORBA o ORB passa argumentos de requisições e controle ao BOA, o qual manipula o processo de seleção e ativação de uma implementação, e retorna os resultados para o ORB. O BOA conecta o ORB aos métodos da implementação pelo uso do esqueleto do servidor⁴. O esqueleto do servidor contém rotinas que mapeiam chamadas do BOA para os métodos em uma implementação particular, que suporte as operações requisitadas.

Além de ser o ponto de conexão entre o ORB e o esqueleto do servidor, o BOA também é um objeto que possui operações definidas sobre ele e que os desenvolvedores de aplicação podem utilizar. O BOA pode ativar e desativar uma implementação, dependendo de como o fornecedor do sistema CORBA implementou a manipulação de desativação. A ativação prepara um objeto para executar operações, como copiar dados armazenados nas memórias das implementações para permitir a execução de métodos sobre os dados armazenados. A desativação libera uma implementação, de forma que ela não possa mais aceitar requisições. O BOA também fornece um grupo de operações que permitem à aplicação do servidor a gerenciar objetos, referências de objetos, e implementações sobre o servidor.

Embora CORBA recomende o conjunto mínimo de operações e capacidades que o BOA deva providenciar, ela não descreve como implementá-las. CORBA não especifica um mecanismo para o *binding* do programa de aplicação ao

⁴ Estes são os esqueletos do servidor que foram gerados a partir das definições de interface.

ORB ou ao BOA. A interface de aplicação entre broker e o adaptador de objetos, assim como entre o BOA e os esqueletos é específico dos implementadores de ORB.

4.4.9 Esqueleto do servidor

O esqueleto do servidor fornece a conexão entre o BOA e os métodos que atendem cada operação sobre um objeto. Os esqueletos possuem informações necessárias para mapear uma operação sobre os objetos, pois detém o código necessário para envio de requisições para um método específico de um objeto.

Quando o ORB recebe uma requisição, ele contacta imediatamente o BOA através de um mecanismo específico do fornecedor do sistema CORBA. Uma vez que o BOA tenha selecionado a implementação apropriada, ele notifica o servidor pela chamada de rotinas contidas no servidor de esqueletos. O esqueleto liga o BOA à implementação e ao método apropriado para a requisição. A maioria da funcionalidade fornecida pelo ORB para a implementação de um objeto é obtida pelos esqueletos IDL e pelo adaptador de objetos BOA.

4.5 CORBA E OS CONCEITOS DE ORIENTAÇÃO À OBJETOS

A integração das aplicações de gerenciamento de infra-estrutura em um ambiente CORBA deve ser baseada em um modelo orientado a objetos. Este modelo fornece base para a análise, projeto e implementação de software que seja extensível e reutilizável[36]. A possibilidade de reutilização e extensão de software é uma característica que favorece o desenvolvimento de sistemas de gerenciamento de infra-estrutura de telecomunicações, pois pode reduzir os tempos de implementação e de adaptação de novos recursos. Portanto, uma implementação com CORBA possibilita a evolução para novas gerações de software de gerenciamento de infra-estrutura e possibilita a extensão de modelos antigos de acordo com as mudanças dos requisitos.

O projeto orientado a objetos concentra-se na abstração em objetos das partes de um sistema, incentivando o desenvolvedor a definir cada objeto do

sistema como um componente, sem que o resto do sistema necessite saber como os componentes realizam suas tarefas.

O modelo de objetos é um conceito da computação orientada a objetos. De forma ampla, um modelo de objetos é uma forma conceitual de pensar em um problema e em suas soluções possíveis. A base para o modelo de objetos é o próprio conceito de objetos, que são entidades que possuem comportamentos e atributos próprios. Os atributos permitem obter ou alterar os dados de um objeto. O comportamento de um objeto é definido pelas operações que podem ser executadas sobre ele. Portanto, cada objeto pode ser manipulado através das operações que possui. Por exemplo, na representação de um retificador como um objeto, este pode possuir uma operação de ligar ou desligar. O nível de tensão em sua saída pode ser um atributo do objeto retificador[2].

Como cada objeto pode realizar apenas certas operações fica claro, tanto para o projetista do sistema quanto para o implementador, o que cada objeto pode realizar. Uma grande vantagem do software orientado à objetos, é que este tende a ser mais fácil de documentar e apresenta mais modularidade em relação à programação procedural.

Como CORBA utiliza um modelo de objetos, todas as aplicações do sistema são vistos como coleções de objetos e operações associadas. A localização da aplicação na rede e o sistema operacional sobre o qual a aplicação roda é transparente ao cliente e ao servidor. Assim sendo, as chamadas de métodos⁵ remotos são transparentes ao usuário.

Abstração é a habilidade de agrupar objetos e focar as características comuns do grupo. Os desenvolvedores utilizam a abstração para decompor os sistemas em grupos de objetos, que podem ser abstraídos ainda mais. Isso possibilita centrar no que pode ser feito com os objetos sem se preocupar com os detalhes de implementação.

⁵ Um método é o código contido em uma implementação que atende a requisição do cliente para uma operação em um objeto específico.

O encapsulamento esconde os detalhes de implementação de um objeto dos seus usuários, ou seja, quando o conceito de encapsulamento é utilizado no contexto de uma aplicação, a forma como um objeto funciona é escondida do resto do sistema. Objetos podem requisitar serviços enviando mensagens para outros objetos que fornecem o serviço. O encapsulamento permite modificar a implementação de um objeto sem afetar os seus clientes, facilitando a implementação.

A herança promove o reuso de software, permitindo que as capacidades e comportamentos de um objeto sejam passados para outro objeto. Quando um objeto herda comportamento de uma interface simples, isso é denominado de herança simples. Quando um objeto herda comportamentos de mais de uma interface, denomina-se herança múltipla. Nem todos os sistemas orientados a objetos possuem herança múltipla. No estudo de caso do capítulo 6 utiliza-se a linguagem JAVA para implementar o sistema distribuído. Veremos que JAVA não permite herança múltipla[17].

Polimorfismo é a capacidade de substituir objetos com interfaces caçadas por outra em tempo de execução. Isso é, pode-se definir uma descrição comum para objetos que compartilham o mesmo comportamento e propriedades. CORBA chama esta descrição de interface. Com polimorfismo, instâncias de objetos que possuem descrição em comum podem responder a mesma requisição utilizando ações diferentes. Polimorfismo possibilita projetar aplicações que sejam facilmente extensíveis. Nem todos os sistemas de objetos requerem polimorfismo, porém CORBA requer.

4.6 MODELO DE COMUNICAÇÃO CORBA

A combinação da computação distribuída e o modelo de objetos em CORBA melhora tanto a computação distribuída quanto a computação orientada a objetos. A incorporação do modelo de objetos em CORBA fornece uma forma de abstrair as complexidades inerentes do ambiente cliente/servidor. A computação orientada a objetos complementa os benefícios da computação distribuída porque

ele esconde as funções de comunicação entre os objetos, como a descoberta da disponibilidade, localização, e os mecanismos de acesso ao servidor.

A figura 20 apresenta o modelo de comunicação CORBA, onde cliente evoca métodos utilizando o stub do cliente e o esqueleto do servidor em uma implementação de objeto. CORBA adiciona referências particulares para objetos no ambiente computacional. Através destas referências os objetos são identificados no sistema.

Corba suporta tanto estilos de comunicação síncrona quanto assíncrona. A definição CORBA para assíncrono é na verdade um modelo de *pooling* onde o cliente pergunta se a operação foi completada. CORBA também define requisições do tipo *one-way*, onde a aplicação não necessita esperar que a requisição seja completada e nenhum argumento é retornado.

Esta capacidade de CORBA de suportar diversos estilos de comunicação possibilita que aplicações de gerenciamento de infra-estrutura tornem-se de fácil implementação, pois os tratamentos de concorrência, pooling e envio de mensagens são transparentes aos implementadores.

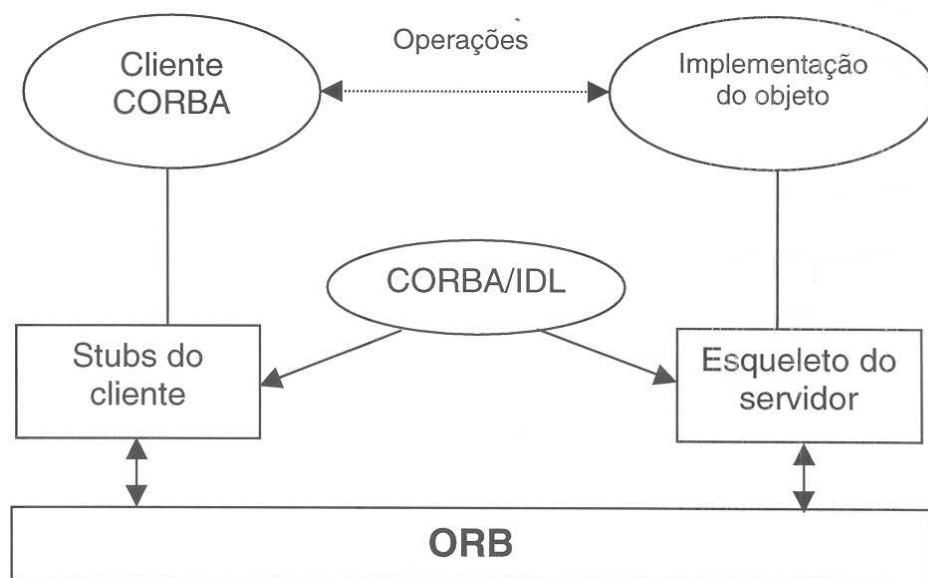


Figura 20 - Modelo de comunicação CORBA

4.6.1 OMG IDL

As interfaces para os servidores podem ser especificadas em uma linguagem simbólica e abstrata. A representação de interfaces nestes moldes é anterior à especificação CORBA e aos sistemas orientados à objetos. Este conceito é utilizado em implementações do tipo RPC – Remote Procedure Call.

O propósito de um arquivo IDL é permitir a definição de interfaces independentes de linguagem. Isso é obtido pelo mapeamento entre a sintaxe IDL – *Interface Definition Language* e qualquer linguagem⁶ que for utilizada para implementar os clientes e servidores. Assim sendo, os clientes e servidores não necessitam ser implementados utilizando a mesma linguagem.

Os desenvolvedores de aplicações CORBA utilizam o OMG IDL para definir a estrutura de dados e as características de interface para objetos. Cada operação definida em uma interface consiste no nome da operação, argumentos, tipos de dados, e retorno de valores. O OMG IDL não é uma linguagem de programação e não pode ser utilizada para gerar objetos ou códigos executáveis.

A linguagem CORBA IDL obedece às mesmas regras léxicas que C++ e inclui várias palavras chave específicas para o uso em sistemas distribuídos. Veremos no capítulo 6 que implementar uma interface IDL é similar a escrever declarações de classe em C++.

Embora não seja requisitado na especificação CORBA, o compilador OMG IDL é uma das partes mais importantes deste sistema. Ele gera o stub do Cliente para a invocação do tipo stub e gera o esqueleto do servidor. Um compilador OMG IDL está incluso na maioria dos sistemas CORBA comercializados.

Pode-se utilizar o arquivo OMG IDL para definir tipos de objetos pela definição de suas interfaces. O corpo de uma interface declara as constantes exportadas, declaração de tipos, exceções, atributos, e operações. Definições de operações incluem nome da operação, tipo de dado retornado, tipo de todos os

⁶ Na verdade é necessário que o fornecedor do sistema CORBA disponibilize o mapeamento para a linguagem em questão.

argumentos, exceções retornáveis e informação contextual que poderiam afetar os dados despachados.

4.7 SERVIÇOS DE OBJETOS CORBA APLICÁVEIS NO GERENCIAMENTO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES.

A seguir são identificados os serviços que têm aplicação direta em sistemas de gerenciamento de infra-estrutura de telecomunicações. Os demais serviços não abordados estão descritos no anexo A e possuem uma aplicação secundária, como controle de licença, propriedade, entre outros.

Salienta-se que muitos dos serviços especificados no *COSS- Common Object Service Specification* ainda estão em fase de implementação. Portanto esta abordagem limita-se a uma avaliação preliminar.

4.7.1 Serviço de nomes

É através do serviço de nomes que os clientes encontram os objetos capazes de oferecerem os métodos desejados. A localização da aplicação é facilitada pela independência da sintaxe e pelo fornecimento de um nome do tipo atributo. O serviço de nomes fornece a habilidade de ligar o nome de um objeto à um contexto de nomes. Este serviço é fundamental para a implementação de sistemas de gerenciamento de infra-estrutura, pois é através dele que os objetos que representam equipamentos gerenciados são encontrados na rede. Boa parte das funções de gerenciamento utilizam este serviço, sendo que o serviço de nomes colabora com a escalabilidade do sistema de gerenciamento de infra-estrutura.

4.7.2 Serviço de eventos

O serviço de eventos tem uma aplicação direta no gerenciamento de alarmes e falhas da infra-estrutura de telecomunicações, pois fornece capacidades básicas que podem ser configuradas de uma maneira flexível e poderosa. O serviço de eventos utiliza o conceito de fornecedor/consumidor. O fornecedor pode ser um objeto que envia eventos, enquanto que o consumidor é um objeto que recebe

eventos. Como os fornecedores podem gerar eventos sem saber a identidade dos consumidores e os consumidores podem receber eventos sem saber a identidade dos fornecedores, é possível a criação de um canal de eventos para o tratamento exclusivo de falhas sem se preocupar com a escalabilidade. Como é desejável informar juntamente com a anúncio da falha qual objeto gerenciado falhou, pode-se estender o canal de eventos básicos para suportar interação de tipos, que possua a informação de qual objeto envia o evento.

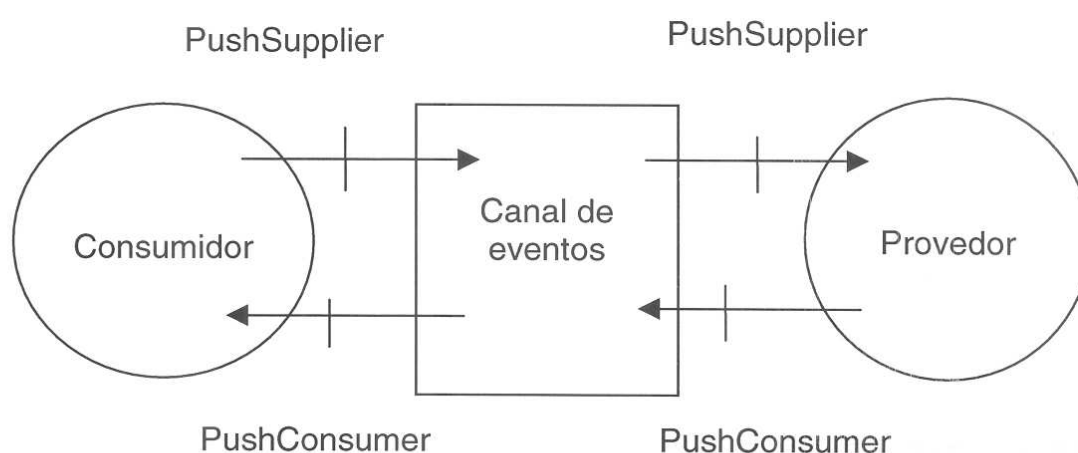


Figura 21 - Canal de eventos estilo *push*

A figura 21 apresenta uma das configurações possíveis do serviço de eventos que apresenta funcionalidades similares ao modelo gerente/agente de TMN e SNMP.

Como são suportados eventos assíncronos na entrada e saídas das notificações, a confiabilidade da entrega de eventos de falha está garantida, podendo existir múltiplos consumidores e fornecedores.

O serviço de eventos elimina a necessidade de um servidor centralizado ou a dependência de qualquer serviço global. As interfaces de serviços de eventos permitem implementações que fornecem diferentes qualidades de serviços para

satisfazer diferentes requisitos de aplicações. Podem existir múltiplos consumidores e fornecedores de eventos.

4.7.3 Serviço de persistência de objetos

O serviço de persistência de objetos é especialmente aplicável no gerenciamento da infra-estrutura, pois fornece um conjunto de interfaces comuns para os mecanismos utilizados na retenção e gerenciamento dos estados persistentes dos objetos. O objeto possui a responsabilidade de gerenciamento do seu estado, mas pode utilizar ou delegar ao Serviço de persistência de Objetos para este trabalho.

A característica principal do Serviço de Persistência de Objetos é sua abertura. Isso significa que é possível haver uma variedade de clientes diferentes e implementações de serviços de persistência de objetos, podendo estes trabalharem em conjunto.

Exemplificando a importância da aplicação do serviço de persistência da infra-estrutura suponhamos a seguinte situação: um objeto que contenha um contador de surtos elétricos e que esteja programado para enviar alarmes cada vez que a contagem atingir um determinado valor. Caso este objeto não seja persistente, toda vez que o *host* que contenha este objeto reinicializar a contagem seria anulada.

4.7.4 Serviço de controle de concorrência

O serviço de controle de concorrência habilita múltiplos clientes a coordenarem seus processos a recursos compartilhados. A coordenação de acesso a recursos significa que, quando múltiplos cliente acessam de forma concorrente um único recurso, as ações conflitantes entre os clientes são reconciliadas de forma a manter o recurso em um estado consistente. Este serviço pode ser aplicado no gerenciamento de bancos de dados destinados a armazenamento de informações oriundas de monitoramentos para fins estatísticos, de todos os equipamentos da infra-estrutura.

O uso concorrente de um recurso é regulado com locks, onde cada lock é associado com um único recurso e um único cliente. A coordenação é atingida prevenindo que múltiplos clientes executem locks para um mesmo recurso, caso as suas atividades possam conflitar. Assim, um cliente deve obter um lock apropriado antes de acessar um recurso compartilhado.

Os serviços de controle de concorrência definem vários modos de locks que correspondem a diferentes categorias de acesso. Essa variedade de modos de locks fornece uma resolução de conflitos flexível. Por exemplo, fornecendo diferentes modos de leitura e escrita, possibilita que um recurso suporte múltiplos clientes concorrentes em uma transação apenas de leitura.

4.7.5 Serviço de tempo

Conforme abordado no capítulo 2, subsistemas podem possuir relógio de tempo real, sendo que estes precisam ser sincronizados. O serviço de tempo pode auxiliar na construção de mecanismos de sincronização, pois possibilita a obtenção do tempo atual com um erro estimado associado, além de acertar a ordem nas quais eventos ocorrem. O serviço de tempo consiste em dois serviços, definindo assim duas interfaces: *Universal Time Objects (UTOs)* e *Time Interval Objects (TIOs)*, sendo representado pela interface *TimeService*.

4.7.6 Serviço de segurança

Gerenciamento de segurança é outro requisito para gerenciamento da infra-estrutura de telecomunicações que é suportado pelos serviços CORBA. A funcionalidade de segurança definida pela especificação CORBA compreende:

- ◆ Identificação e autorização dos usuários (usuários humanos e objetos que precisam operar sob seus direitos) para verificar se eles são realmente quem eles anunciam.
- ◆ Autorização e controle de acesso – decidir se um usuário pode acessar um objeto, normalmente utilizando a identidade e/ou outro atributo de privilégio do

usuário (como papel, grupos, clareza de segurança), e se os atributos de controle do objeto alvo podem acessá-lo.

- ♦ A auditoria de segurança torna possível identificar os usuários corretamente, mesmo após uma cadeia de chamadas através de vários objetos.
- ♦ A segurança de comunicação entre objetos, que geralmente é realizada sobre as camadas mais baixas de comunicação.
- ♦ *Non-repudiation* fornece evidências irrefutáveis de ações como prova de origem dos dados no recipiente ou prova das datas de recebimento do emitente para proteção contra tentativas consecutivas de falsamente negar o recebimento ou envio de dados.

5 ARQUITETURA PARA GERENCIAMENTO DA INFRA-ESTRUTURA DE TELECOMUNICAÇÕES BASEADA EM CORBA

5.1 INTRODUÇÃO

Conforme apresentado nos capítulos anteriores, o ambiente de gerenciamento de infra-estrutura de telecomunicações compõe-se de informações gerenciais distribuídas, pois os dados necessários para o gerenciamento são provenientes de muitas origens. Isso dificulta o gerenciamento integrado da infra-estrutura de telecomunicações, sendo que sistemas de gerenciamento de diferentes fabricantes necessitam de conversões especiais para possibilitarem o compartilhamento de informações.

A arquitetura proposta neste capítulo visa oferecer uma base para a implementação de sistemas de gerenciamento que suportem características de distribuição e heterogeneidade dos equipamentos.

A arquitetura para gerenciamento da infra-estrutura de telecomunicações proposta neste capítulo é baseada em CORBA e define uma estratégia de alto nível para a implementação de sistemas integrados de gerenciamento, permitindo que os implementadores de tais sistemas trabalhem níveis mais elevados de abstração. Adotou-se a mesma abordagem da TMN, sendo que a arquitetura é definida sob três perspectivas, ou seja: arquitetura física, funcional e de informação. A arquitetura física engloba os componentes físicos que compõe o sistema de gerenciamento, como computadores, redes e interfaces. A arquitetura funcional é apresentada em camadas constituídas por blocos funcionais capazes de dar suporte às funções de gerenciamento. E finalmente, a arquitetura de informação, totalmente baseada em objetos, oferece um *framework* básico para a implementação de sistemas de gerenciamento de infra-estrutura.

5.2 ARQUITETURA FÍSICA

A arquitetura física proposta para o gerenciamento da infra-estrutura de telecomunicações, baseada em CORBA, consiste por uma série de computadores

ligados em rede, preferencialmente próximos aos subsistemas de infra-estrutura e dotados de algum tipo de interface com os equipamentos que constituem os subsistemas, seus controladores ou mesmo com sistemas de gerenciamento legados. Opcionalmente, o hardware no qual roda a aplicação de gerenciamento pode estar integrado ao subsistema de infra-estrutura.

A figura 22 ilustra a arquitetura física, onde não existe uma máquina específica que venha a realizar as funções de gerenciamento como em TMN e SNMP. Na arquitetura proposta o gerenciador pode ser um sistema que roda em qualquer computador da rede. Portanto, o computador A pode gerenciar tanto os recursos de infra-estrutura representados no computador A quanto no computador B, assim como o computador B pode gerenciar os recursos alocados no computador A e no próprio computador B. Os subsistemas S1, S2, ... Sn, podem ser do grupo I ou do grupo II, desde que possuam algum tipo de interface para aquisição de dados.

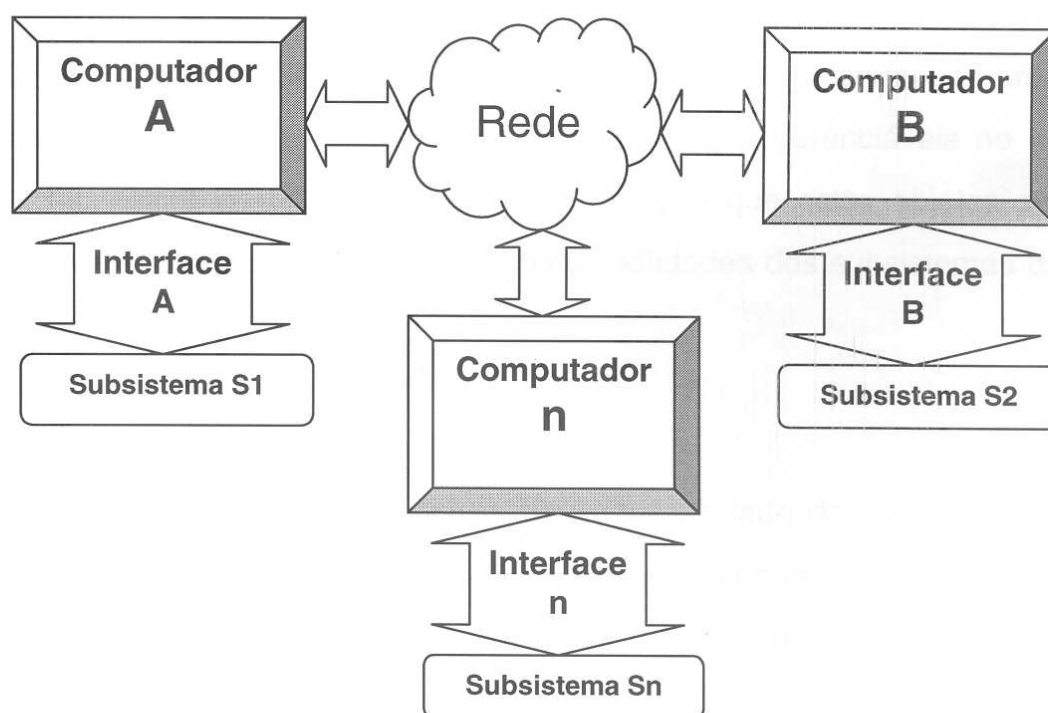


Figura 22 - Arquitetura física genérica

As interfaces A, B, ... n são normalmente proprietárias, de baixo nível e fogem do escopo deste trabalho. Entretanto, salienta-se que mecanismos devem ser construídos internamente aos objetos gerenciáveis, que sejam capazes de acessar estas interfaces, objetivando a comunicação de dados. Estas interfaces podem ainda ser mapeamentos SNMP-CORBA ou CMIP-CORBA[16] que possibilitem o gerenciamento de sistemas legados.

A exemplo das normas Telebrás, não é definida uma tecnologia de rede de dados específica para a arquitetura física. Porém, a rede que suporta os meios de comunicação para o sistema de gerenciamento distribuído deve oferecer velocidade, desempenho, disponibilidade e robustez compatíveis com a aplicação. A rede pode ser local - LAN ou mesmo uma rede mais ampla, como uma WAN – *Wide Area Network* ou a própria Internet[26]. Portanto, o mecanismo de transporte de dados, entre os computadores que exercem as funções de gerenciamento distribuído, pode ser baseado em qualquer tecnologia que seja integrável aos sistemas CORBA comerciais.

Embora o gerenciamento seja feito sobre os equipamentos que constituem os subsistemas, estes não são considerados como parte integrante da arquitetura física, sendo eles representados em termos de objetos gerenciáveis no sistema. Veremos na descrição da arquitetura de informação que estes objetos possuem atributos e métodos que representam as funcionalidades dos subsistemas da infraestrutura de telecomunicações.

5.3 ARQUITETURA FUNCIONAL

A proposta de uma arquitetura funcional, no lado do subsistema, para o gerenciamento informatizado da infra-estrutura é apresentada na figura 23. O sistema é dividido em camadas, sendo que a camada mais inferior é o próprio sistema operacional dos computadores servidores.

Uma das vantagens desta arquitetura é a possibilidade da operação em múltiplas plataformas, não sendo necessário prender-se a um único sistema operacional. Como pode-se observar, a arquitetura possibilita a coexistência de

diversos sistemas operacionais. Ou seja, uma aplicação de gerenciamento que roda em UNIX pode trocar informações gerenciais com um outro sistema que esteja rodando sobre Windows NT. Essa abertura favorece a integração dos sistemas legados e diminui o tempo de implementação dos sistemas de gerenciamento de infra-estrutura.

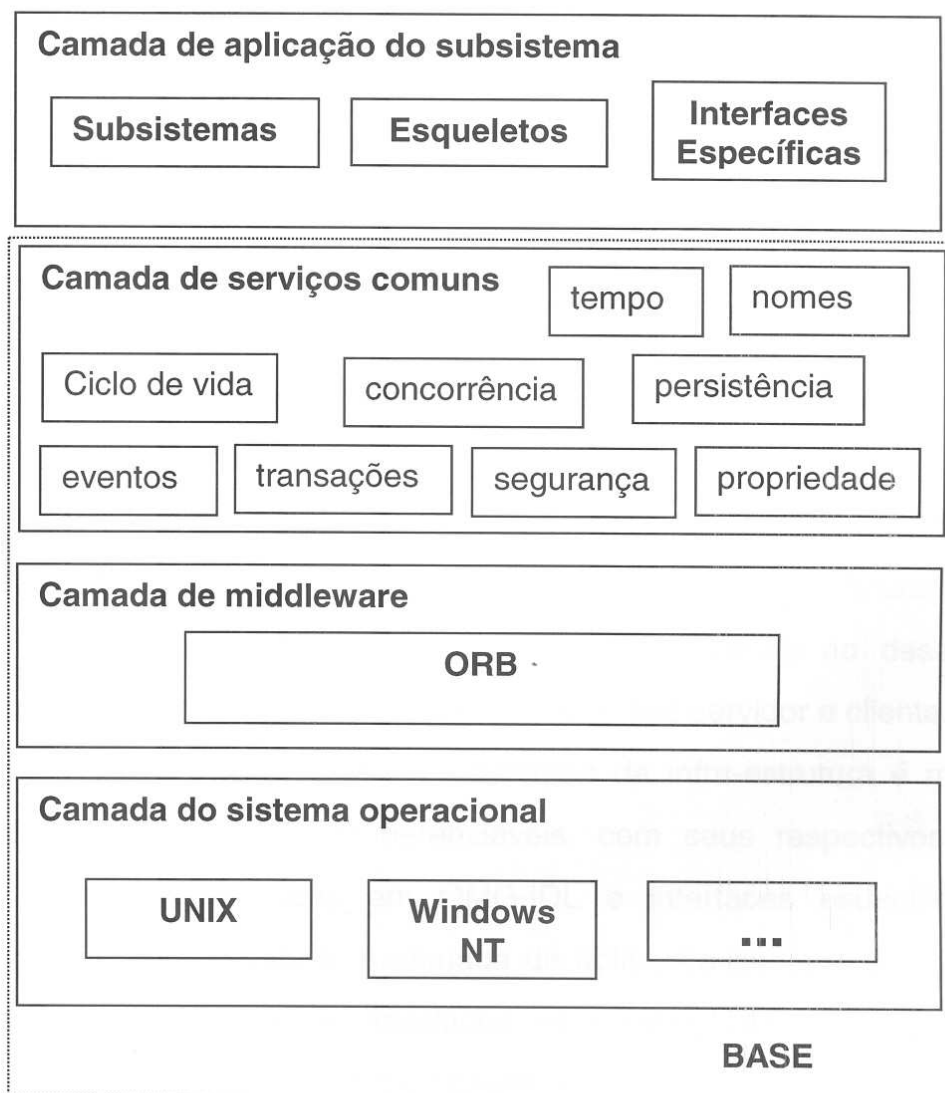


Figura 23 - Arquitetura para gerenciamento baseada em CORBA - lado do subsistema

A camada de middleware é implementada por intermédio de CORBA. Embora várias tecnologias pudessem ser empregadas na camada de middleware, a

arquitetura CORBA oferece atualmente maior abrangência do que as outras alternativas descritas no capítulo anterior. Esta camada, responsável pela comunicação entre os sistemas gerenciadores e gerenciados, direciona as mensagens de alarmes ou localiza os subsistemas e seus equipamentos na rede de forma transparente aos usuários e implementadores do sistema de gerenciamento.

A camada de serviços comuns está baseada no COSS – *Common Object Service Specification*, [28], e consiste em um conjunto de interfaces e descrições de comportamento de serviços comuns CORBA. A utilização e aplicação destes serviços no gerenciamento da infra-estrutura já foi abordado no capítulo anterior. Nesta proposta de arquitetura sugere-se que as funcionalidades de gerenciamento de segurança e de alarmes sejam executados por esta camada, pois os serviços de eventos e de segurança, especificados no COSS, podem suportar estas funcionalidades. Outros serviços, como serviços de licença e propriedades teriam aplicação em implementações comerciais da arquitetura proposta.

As três camadas mais inferiores constituem a base da arquitetura funcional para gerenciamento da infra-estrutura. Tanto o software gerenciador quanto o gerenciado se apoiam sobre esta mesma base, cabendo ao desenvolvedor a implementação da camada de aplicação dos sistemas servidor e cliente.

A aplicação no lado dos subsistemas da infra-estrutura é modelada em termos de classes de objetos gerenciáveis, com seus respectivos atributos e métodos, interfaces definidas em OMG-IDL e interfaces específicas com os subsistemas de infra-estrutura. A camada de aplicação tem a responsabilidade de coletar os dados a partir de interfaces específicas com os equipamentos ou dispositivos controladores da infra-estrutura, instanciar objetos que representem os recursos gerenciados e proporcionar a conexão com o ambiente CORBA.

Para possibilitar o gerenciamento dos subsistemas através do ORB, são utilizados os esqueletos do servidor, através dos quais o BOA realiza as chamadas para os métodos que suportam as operações necessárias. Conforme visto anteriormente, estes esqueletos são gerados a partir das interfaces OMG-IDL. Estas interfaces devem ser bem definidas e estarem disponíveis aos implementadores do

sistema. Caso a definição de novas interfaces seja necessário, estas devem ser publicadas e acessíveis aos implementadores, promovendo a abertura tão desejada no gerenciamento da infra-estrutura de telecomunicações.

A camada de aplicação do subsistema oculta as interfaces específicas entre o ele e os equipamentos de infra-estrutura dos sistemas gerenciadores, ou seja, são as interfaces específicas que definem as iterações e o fluxos de informações através das fronteiras da camada de aplicação do subsistema com os equipamentos de infra-estrutura. O ocultamento de informações internas permite que as interfaces IDL não necessitem ser modificadas quando alterações ocorrem nas interfaces específicas. Cada subsistema é implementado de modo independente sem afetar aos demais.

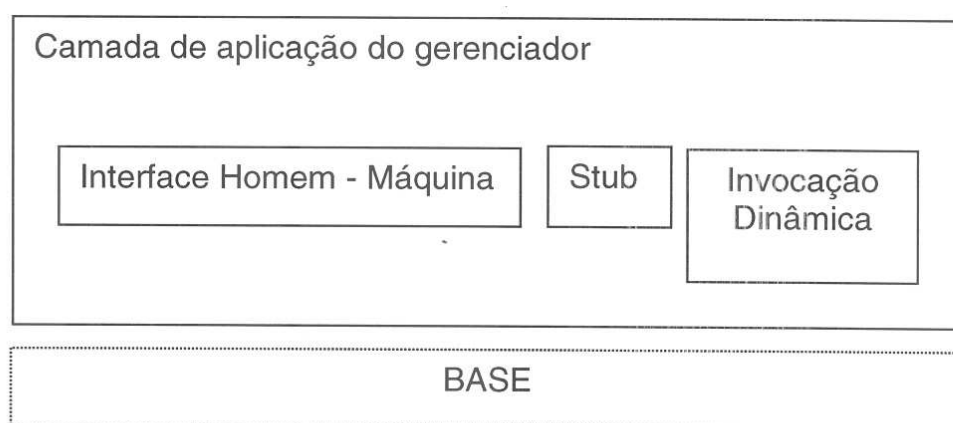


Figura 24 - Arquitetura para gerenciamento baseada em CORBA - lado do gerenciador

A camada de aplicação do gerenciador se apoia na mesma base do subsistema e possui os stubs dos clientes gerados a partir da compilação dos arquivos OMG-IDL. A figura 24 ilustra a camada de aplicação do gerenciador. De forma alternativa é possível efetuar ações de gerenciamento através de métodos de evocação dinâmica previstos no CORBA. Para o sistema cliente utilizar este mecanismo é necessário que ele possua as referências dos objetos distribuídos que

representam os elementos gerenciados, sendo conhecidos ou obtidos a partir do repositório de interfaces. Isso pode ser feito automaticamente pelo serviço de nomes.

O relacionamento entre o sistema gerenciador e os subsistemas gerenciados segue o paradigma cliente/servidor. O cliente é um sistema gerenciador que evoca no servidor métodos para gerenciamento e fornece os resultados ao usuário através de uma interface homem/máquina, ou armazena-os em um banco de dados para análise futura. A figura 25 ilustra este relacionamento, procurando destacar a conexão entre a camada do servidor e a camada do cliente.

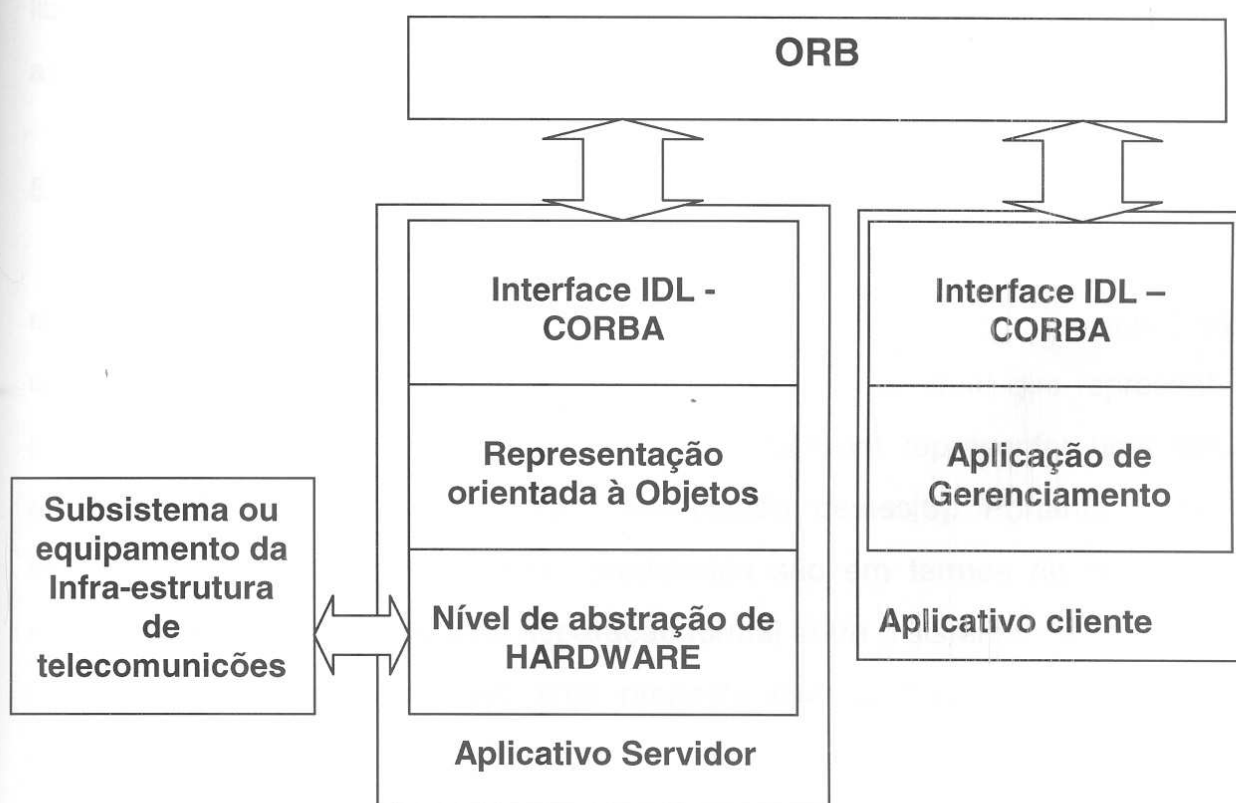


Figura 25 - Relacionamento funcional cliente/servidor

O aplicativo do servidor possui duas funções básicas que são: abstrair as particularidades de hardware a partir dos dados através de interfaces específicas com os subsistemas de infra-estrutura, e proporcionar interfaceamento com o sistema de gerenciamento baseado em CORBA. A representação dos recursos em

objetos gerenciáveis não exige necessariamente um mapeamento um para um, ou seja, um grupo de recursos pode ser representado por apenas um objeto caso seja mais conveniente.

O aplicativo do cliente também deve possuir interfaces CORBA definidas através de arquivos OMG- IDL. Embora não esteja explícito na figura 25, o aplicativo do cliente implementa uma interface para o usuário do sistema, assim como implementa mecanismos de acesso a banco de dados, visando armazenar os dados de gerenciamento pertinentes.

Nota-se que esta arquitetura não se limita a apenas alguns serviços comuns como especificado em TMN e SNMP (get, set, ...). O implementador tem a liberdade de definir novos tipos de serviços de acordo com a necessidade da aplicação.

5.4 ARQUITETURA DE INFORMAÇÃO

A arquitetura de informação proposta para o gerenciamento da infraestrutura de telecomunicações é totalmente baseado no princípio de orientação à objetos CORBA. Um objeto gerenciado é uma visão conceitual que representa as propriedades físicas de um recurso, podendo também representar uma relação entre os recursos ou, ainda, uma combinação destes[5]. Portanto, todas as funcionalidades de gerenciamento modeladas são em termos de evocação de métodos remotos, existindo uma separação formal entre o sistema cliente e sistema servidor. Apresenta-se a seguir uma proposta para a arquitetura do servidor e posteriormente para o cliente.

A arquitetura apresentada baseia-se nos subsistemas de infra-estrutura pertencentes ao grupo I. Optou-se pela abordagem do grupo I devido a maior quantidade de informações obtidas durante a revisão das normas, entretanto a arquitetura pode ser estendida para os subsistemas pertencentes ao grupo II. Utiliza-se para a apresentação da arquitetura de informação a notação definida pela UML – Unified Modeling Language[29][12].

A figura 26 apresenta um diagrama de classes geral que é a base da proposta para a arquitetura de informação do sistema de gerenciamento de infraestrutura baseado em CORBA. Como pode-se observar, a classe Equipamento herda da classe Skeleton os comportamentos e atributos que possibilitarão o gerenciamento distribuído dos objetos. O uso de herança é a forma mais simples de implementar o servidor, porque os objetos servidores e as referências dos objetos se comportam da mesma maneira[42]. Entretanto, o uso de herança para o modelo do servidor possui uma desvantagem: Em linguagens como JAVA, que suportam apenas herança simples[17], o servidor não pode utilizar este mecanismo para outros propósitos, como por exemplo para implementar a interface do usuário ou outras interfaces específicas. Como alternativa ao uso de herança múltipla, pode-se utilizar o mecanismo de delegação[4], acarretando a necessidade de código adicional.

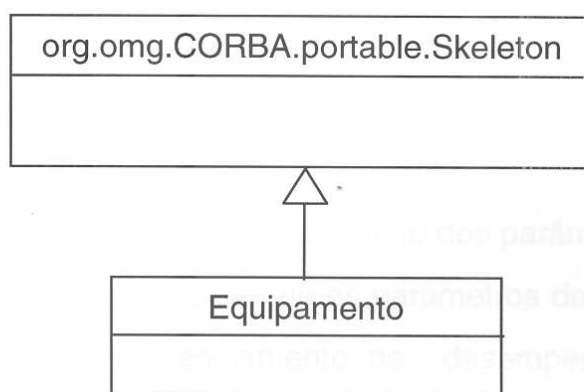


Figura 26 - Diagrama de classe geral

A classe Skeleton é gerada a partir da compilação do arquivo OMG-IDL, onde as interfaces são especificadas independentemente da linguagem. A classe Equipamento representa qualquer equipamento da infra-estrutura que precise ser gerenciado. Esta classe, por herdar de Skeleton, oferece acesso dos seus métodos que estejam definidos no arquivo IDL. Opcionalmente, os clientes podem evocar métodos pertencentes à classe Equipamento de forma dinâmica, sem necessitarem

do stub do cliente. Ambos os mecanismos são implementados no estudo de caso desenvolvido no próximo capítulo.

Para uma definição mais detalhada da arquitetura de informação é necessário identificar os atributos e métodos necessários para o gerenciamento dos subsistemas. No capítulo 2, sessão 2.3.7, foram definidos os níveis de gerenciamento e as funcionalidades que sistemas de gerenciamento de infraestrutura devem apresentar, identificando-se claramente que o gerenciamento da infra-estrutura de telecomunicações pode ser realizado no segundo nível de gerenciamento, ou seja, de gerência de equipamentos. Neste nível as funções de gerência aplicáveis à infra-estrutura de telecomunicações são: gerência de alarmes, configuração, desempenho, e segurança. Antes de implementar estas funcionalidades é necessário identificar qual a natureza dos parâmetros gerenciados, visando a definição de uma arquitetura abrangente. Para isso, os parâmetros referentes aos subsistemas são classificados como: parâmetros de identificação, de estado funcional, de medição e de alarme. As ações de controle também foram identificadas e classificadas, pois definem alguns dos métodos necessários.

A estratégia utilizada para a classificação dos parâmetros gerenciáveis está apresentada na figura 27. Observa-se que os parâmetros denominados de medição são utilizados tanto para o gerenciamento de desempenho dos equipamentos quanto para o gerenciamento de configuração. Ou seja, quando é realizada uma ação de controle, normalmente é verificado se o processo foi corretamente realizado pela leitura dos parâmetros de medição. Caso a verificação indique desvios, alarmes podem ser gerados.

Os parâmetros que representam os estados funcionais dos equipamentos podem ser alterados a partir de operações de controle ou por causas externas. Estes parâmetros também podem ser manipulados pelo gerenciamento de configuração. Porém, caso o estado funcional se altere devido à outros fatores inesperados, notificações de alarmes devem ser efetuadas. Além da alteração de parâmetros de estado funcional, o gerenciamento de configuração necessita ter

acesso à leitura destes a qualquer instante, visando justamente verificar a configuração atual.

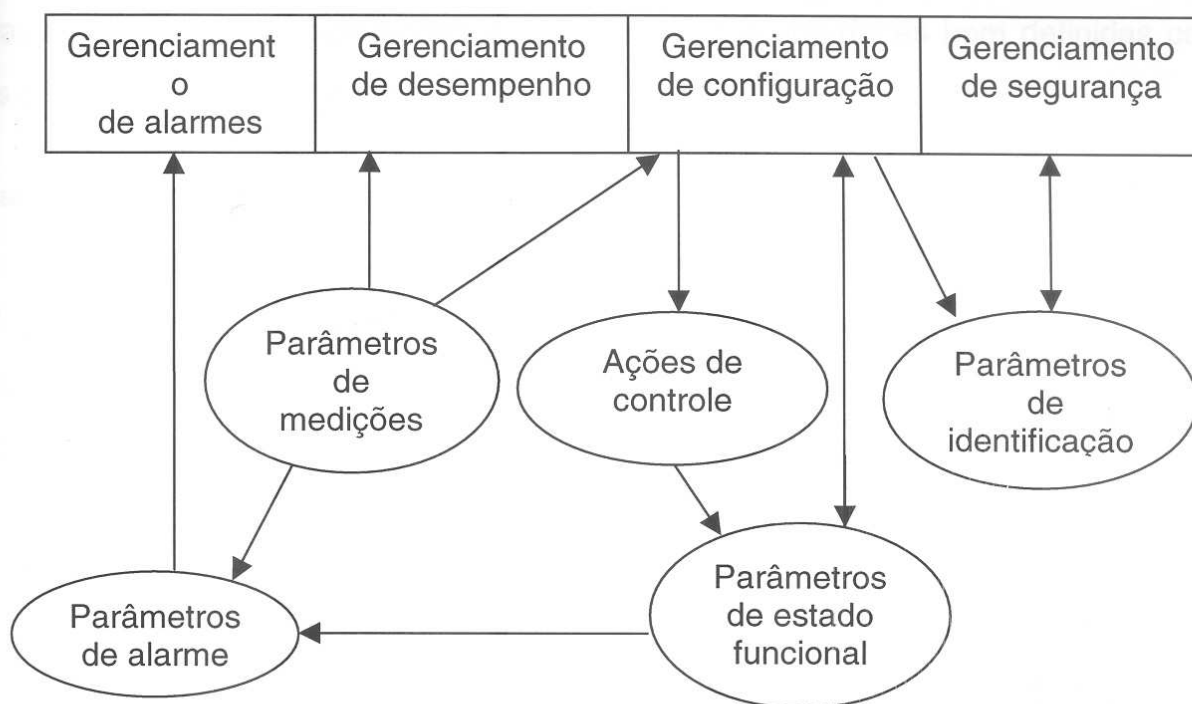


Figura 27 - Funcionalidades de gerenciamento e parâmetros gerenciados

Os parâmetros gerenciáveis e as ações de controle são identificados de acordo com estratégia apontada na figura 27. A seguir, estes parâmetros são classificados em tabelas, de acordo com a sua utilização, relacionando-os com os seguintes subsistemas:

S1 – Subsistema de energia CC

S2 – Subsistema de energia CA

S3 – No-break

S4 – Subestação de Energia

S5 – Subsistema de iluminação

S6 – Subsistemas hidráulicos

A opção da divisão da infra-estrutura de telecomunicações em subsistemas é muito conveniente, pois permite simplificar a modelagem em objetos do sistema de gerenciamento integrado. Para a arquitetura de informação um subsistema não é apenas um objeto nem uma função, mas sim um pacote de classes, associações, operações e eventos que têm interfaces bem definidas com os sistemas de gerenciamento.

Parâmetros de Identificação

Apesar de não terem sido abordados no capítulo 2, os parâmetros de identificação são necessários para dar identidade aos equipamentos que compõe a infra-estrutura de telecomunicações, além de serem fundamentais para o gerenciamento de segurança. A tabela 2 apresenta alguns dos parâmetros de identificação.

Parâmetro gerenciado \ Subsistema	Subsistema					
	S1	S2	S3	S4	S5	S6
Nome do fabricante	X	X	X	X	X	X
Tipo de equipamento	X	X	X	X	X	X
Localização	X	X	X	X	X	X
Número de série	X	X	X	X	X	X

Tabela 1 – Parâmetros de identificação

Verifica-se que todos os equipamentos pertencentes aos subsistemas devem possuir parâmetros que os identifiquem.

Ações de controle

Parâmetro gerenciado \ Subsistema	Subsistema					
	S1	S2	S3	S4	S5	S6
Desinibir USCA		X				
Desligar	X	X			X	X
Dimerizar					X	
Inibir USCA		X				
Ligar	X	X			X	X

Tabela 2 - Parâmetros de controle

Na tabela 3 são classificadas algumas ações de controle que visam alterar o estado de parâmetros de equipamentos da infra-estrutura de telecomunicações. Verifica-se que as ações de ligar e desligar são necessárias para a maioria dos subsistemas.

Parâmetros que representam estados funcionais

Os parâmetros gerenciados listados na tabela 4 representam basicamente o estado de equipamentos da infra-estrutura que não caracterizam falhas. Estes parâmetros são normalmente representados como sendo booleanos.

Subsistema	S1	S2	S3	S4	S5	S6
Parâmetro gerenciado						
Bateria desconectada	X					
Bateria em carga			X			
Bateria em descarga	X					
Bomba de combustível operando		X				
Bomba pré – lub operando		X				
Bomba pré aquecedor operando		X				
Chave manual/automático						X
Disjuntor de entrada CA			X			
Disjuntor de saída CA			X			
Disjuntor em trip				X		
Estado circuito de iluminação e tomadas					X	
Estado da rede			X			
Estado disjuntor de alta tensão				X		
Estado disjuntor de baixa tensão				X		
Estado disjuntor de by-pass			X			
Estado dos capacitores				X		
Estado módulo de rede		X				
Estado módulo GMG		X				
Estado pré-aquecimento		X				
Estado seccionadora de entrada				X		
Estado seccionadora de saída				X		
Estado seccionadora do trafo				X		
GMG em operação		X				
GMG em carga		X				
GMG telecomandado		X				
Inversor em carga			X			
Manual/automático			X			
Manutenção	X	X	X			
Rede em carga		X	X			
Reposição	X	X	X			
USCA inibida		X				

Tabela 3 - Parâmetros de estado funcional

Os parâmetros de estados funcionais mais genéricos são os estados de manutenção e reposição. Os demais parâmetros tendem a ser específicos para cada equipamento ou subsistema.

Parâmetros de medições

Os parâmetros de medição são fundamentais para o gerenciamento da infra-estrutura, sendo basicamente grandezas numéricas dimensionais. São obtidos fisicamente através da leitura de elementos sensores próximos aos equipamentos. A tabela 6 apresenta os principais parâmetros de medição do grupo I.

Subsistema	S1	S2	S3	S4	S5	S6
Parâmetro gerenciado						
Consumo médio instantâneo				X		
Corrente bateria	X		X			
Corrente consumidor	X					
Corrente de saída CC			X			
Corrente de terra				X		
Corrente trifásica de consumidor			X	X		
Demanda instantânea				X		
Fator de potência				X		
Frequência de consumidor			X	X		
Luminosidade					X	
Nível da caixa						X
Nível óleo isolante				X		
Pressão dos gases estágio 1				X		
Temperatura do mancal do alternador		X				
Temperatura do motor		X				
Temperatura elemento	X					
Temperatura entrada do alternador		X				
Temperatura estágio 1				X		
Temperatura estágio 2				X		
Tensão bateria	X					
Tensão consumidor	X					
Tensão de saída CC			X			
Tensão elemento	X					
Tensão trifásica de consumidor			X	X		

Tabela 4 - Parâmetros de medição

Parâmetros que representam alarmes

Apesar dos alarmes poderem ser classificados de acordo com o seu impacto no funcionamento da infra-estrutura, a tabela 5 não destingue se os alarmes são do tipo principal, secundário, alerta, alerta capturado ou apenas

gravação. O objetivo desta tabela é simplesmente identificar os parâmetros que representam alarmes para os subsistemas do grupo I. Entretanto, sistemas comerciais devem possibilitar esta distinção.

Pode-se notar que alguns dos alarmes se repetem para os subsistemas abordados, como falha CA, defeito e fusível interrompido.

Subsistema	S1	S2	S3	S4	S5	S6
Parâmetro gerenciado						
CA/rede anormal	X	X	X	X		
Defeito	X		X			
Defeito pressão/sobre velocidade		X				
Disjuntor anormal		X				
Falha de frequência		X				
Falha de tensão		X				
Falha na partida		X				
Falta de fase						X
Final de carga eminente		X				
Flutuação anormal	X					
Fusível interrompido	X	X	X			
GMG anormal		X				
Nível alto tanque diário		X				
Nível baixo tanque diário		X				
Retificador anormal	X	X				
Sobrecarga		X				
Sobretensão		X				
Tensão cons. Baixa	X					
USCA anormal		X				

Tabela 5 - Parâmetros de alarme

5.4.1 Modelo geral para os subsistemas

A forma escolhida neste trabalho para a definição da arquitetura de informação para os subsistemas é a modelagem da infra-estrutura de telecomunicações em termos de objetos gerenciáveis. O modelo apresentado na figura 28 apoia-se na classificação dos parâmetros gerenciados realizada na sessão anterior e nos diagramas apresentados no capítulo 2.

Os objetos gerenciáveis do grupo I são facilmente identificados: Retificador, Controlador, USCA, Rede, GMG, Chave de transferência, Inversor, Bateria, Transformador, Disjuntor, Chave seccionadora, Quadro de distribuição, Banco de capacitores, Quadro de subestação, Sensor de luminosidade, Lâmpadas, Caixa de água (ou poço de águas fluviais) e Bombas.

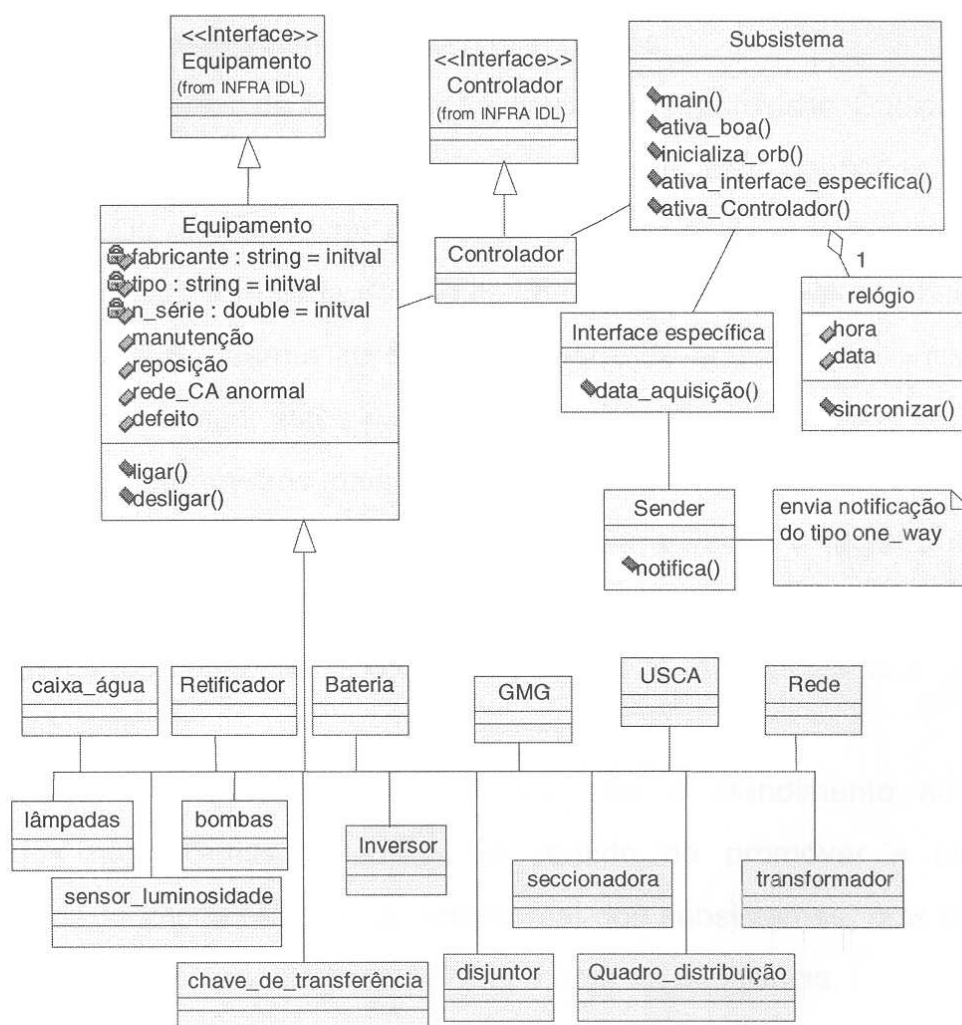


Figura 28 – Diagrama de classes para sistemas gerenciados

O modelo apresentado não inclui todos os elementos passíveis de gerenciamento existentes em uma infra-estrutura de telecomunicações, porém outros equipamentos podem ser facilmente incorporados ao modelo.

O objeto Subsistema é responsável por inicializar o ORB e o adaptador de objetos - BOA no lado do servidor, registrar o objeto Controlador no BOA, além de instanciar os objetos do tipo controlador. Salienta-se que a classe Subsistema não retorna nenhum argumento quando instanciada.

Para um gerenciamento verdadeiramente distribuído, optou-se por definir para cada objeto do tipo "Equipamento" uma interface OMG-IDL através do qual as funcionalidades de gerenciamento serão executadas.

Os parâmetros de identificação estão na superclasse Equipamento, pois como observa-se na tabela 2, estes parâmetros são genéricos a todos os equipamentos. Os métodos ligar e desligar também estão nesta superclasse, pois tendem a existir para a maioria dos equipamentos. Como pode-se observar alguns parâmetros, como de alarme de defeito, estados de manutenção e reposição são genéricos. Devido a isso, estes parâmetros também foram incluídos na superclasse equipamento. Os parâmetros mais específicos foram alocados nas classes filhas.

A classe Controlador é responsável pela ativação e registro no BOA dos objetos Equipamento. É através do objeto Controlador que os sistemas Gerenciadores (ou neste caso, os clientes) obtêm as referências para os objetos do tipo Equipamento.

A especificação da classe relógio visa o atendimento aos requisitos encontrados nas normas Telebrás, no sentido de promover a sincronização obrigatória de todos os relógios de tempo real dos subsistemas, pois os eventos e alarmes devem estar associados com data e hora da ocorrência.

Embora os alarmes estejam associados aos equipamentos, optou-se aqui pela definição de uma classe exclusiva para notificação dos alarmes objetivando a inclusão de estratégias filtragem. A classe Sender pode notificar alarmes através do serviço de canal de eventos ou opcionalmente enviar as notificações através da evocação de métodos do tipo `one_way`, ou seja, que não retornam argumentos. No modelo proposto optou-se pela notificação de alarmes utilizando métodos `one_way` pela sua simplicidade de uso. Portanto, no caso de ocorrência de alarmes, o objeto interface específica evoca um pedido de registro em outro servidor chamado aqui de

Registrador. A figura 29 ilustra este mecanismo, sendo que o servidor do objeto Registrador pode estar alocado em qualquer máquina da rede, podendo ou não ser incorporado a um sistema gerenciador.

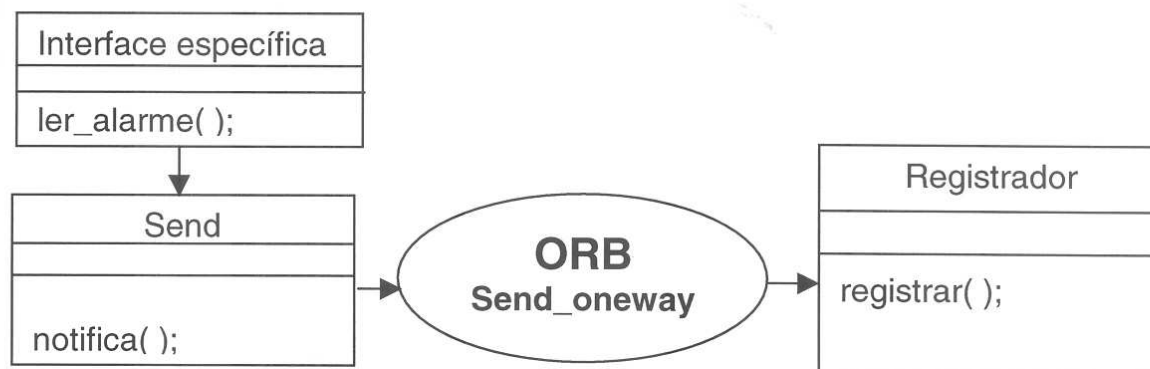


Figura 29 – Mecanismo para notificação de falhas

A classe **interface específica** é responsável por realizar a comunicação com os recursos físicos e atualizar os atributos internos dos objetos gerenciáveis, criados a partir da classe Controlador. Se o atributo representar um alarme, esta interface atualiza o parâmetro do objeto gerenciado e notifica a ocorrência para um servidor específico, o qual irá registrar o alarme no sistema gerenciador.

5.4.2 Arquitetura de informação do gerenciador

A figura 30 apresenta um diagrama de classes que representa a arquitetura de informação do lado do gerenciador. Como pode-se observar, a classe Gerenciador possui o método `bind( )`, o qual permite obter a referência de um objeto remoto a ser gerenciado. O mesmo programa gerenciador que realiza as funcionalidades de gerenciamento para os objetos remotos pode gerenciar objetos locais utilizando ou não os mecanismos CORBA.

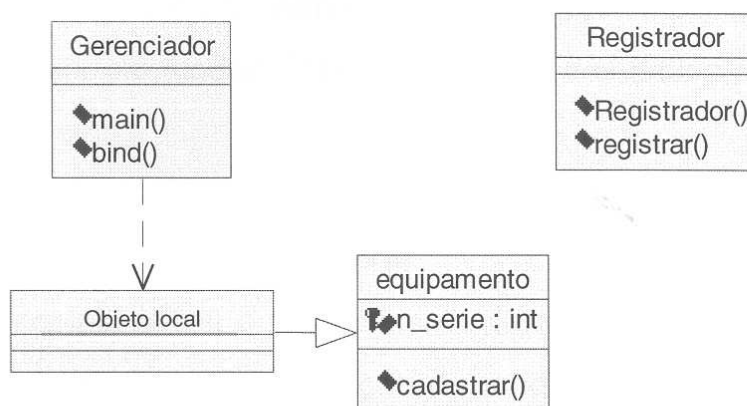


Figura 30 - Diagrama de classes do sistema gerenciador

A objeto Registrador possui o método registrar(), que recebe as notificações de alarme oriundos do objeto Sender, alocado no servidor. Apesar do objeto estar no lado do cliente Gerenciador, ele funciona como um servidor de registros de alarmes dos subsistemas. Como pode-se observar ele opera independentemente do objeto Gerenciador, podendo inclusive rodar em um outra máquina. Obviamente, para o gerenciamento da infra-estrutura, é fundamental que os registros efetuados pelo objeto Registrador sejam acessíveis pelo objeto Gerenciador. Portanto, todos os registros de alarme são gravados em banco de dados, sendo que o gerenciador pode a qualquer momento obter os dados. A figura 31 ilustra esta conexão.

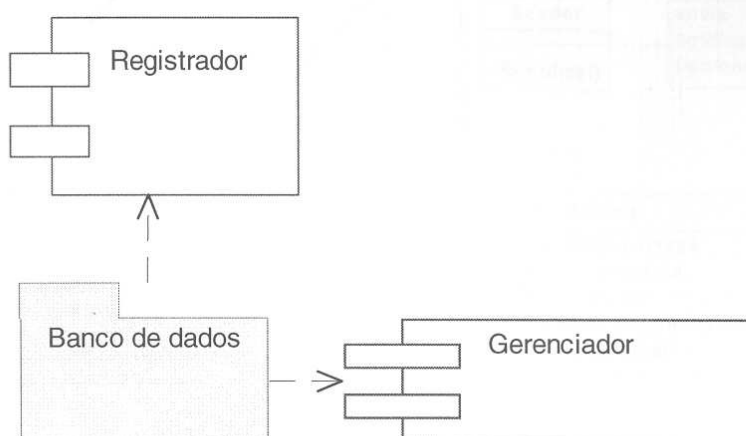


Figura 31 – Modelo para acesso aos parâmetros de alarmes

A seguir veremos como o modelo genérico proposto pode ser particularizado para um modelo específico a partir dos dados apurados nas tabelas anteriores:

5.4.3 Especialização do modelo geral

Para complementar o entendimento desta proposta, apresenta-se a seguir uma metodologia para a especialização do modelo geral. Embora a metodologia apresentada possa ser aplicada a qualquer um dos subsistemas, escolheu-se o subsistema de energia CC, ou S1, como exemplo. Os equipamentos que fazem parte deste subsistema são retificador, bateria e um painel, dedicado à apresentação local de parâmetros deste subsistema.

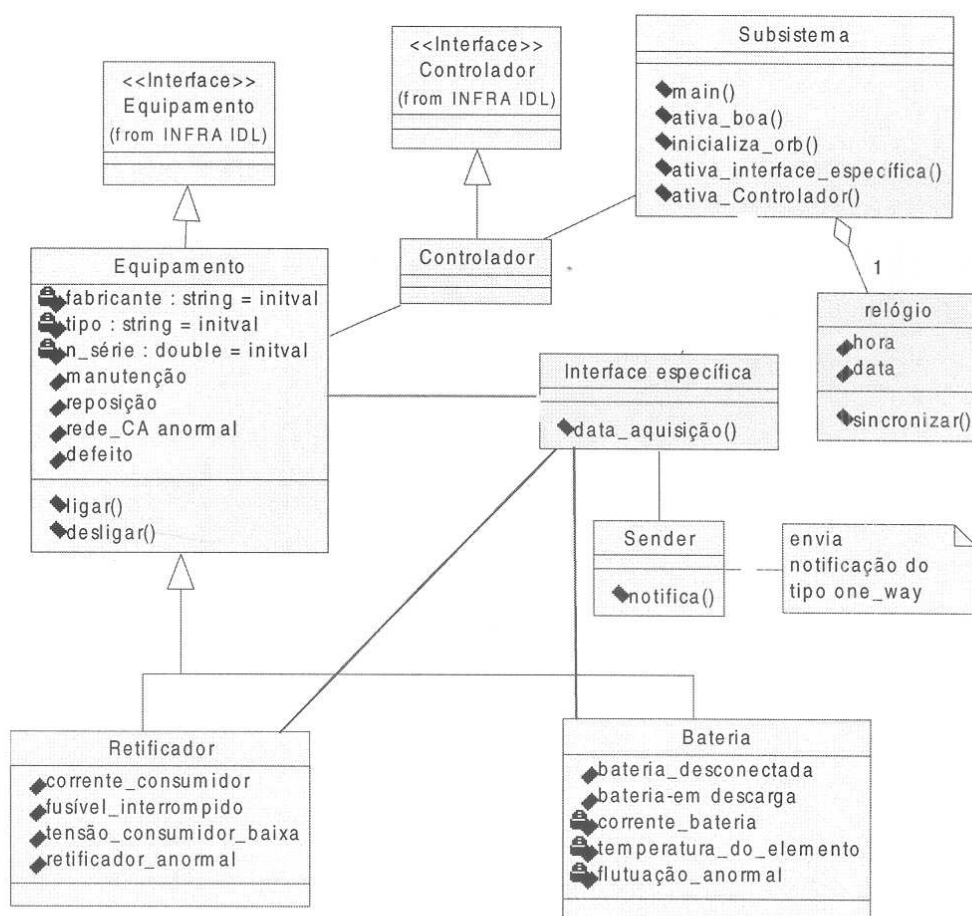


Figura 32 - Diagrama de classes do subsistema de energia CC

A figura 32 apresenta o diagrama de classes completo para o subsistema de energia CC . Os métodos e atributos são extraídos das tabelas anteriores. Nota-se que o modelo para o subsistema de energia CC é totalmente baseado no modelo genérico apresentado anteriormente.

Portanto, outros modelos podem ser especializados a partir do modelo da figura 28 para os demais subsistemas da infra-estrutura de telecomunicações, bastando utilizar as tabelas de classificação de parâmetros para identificar os métodos e atributos. A escolha de incluí-los em uma classe superior ou em uma classe filha pode ser baseada no fato dos parâmetros e métodos serem gerais ou específicos aos subsistemas e equipamentos da infra-estrutura.

6 ESTUDO DE CASO

6.1 INTRODUÇÃO

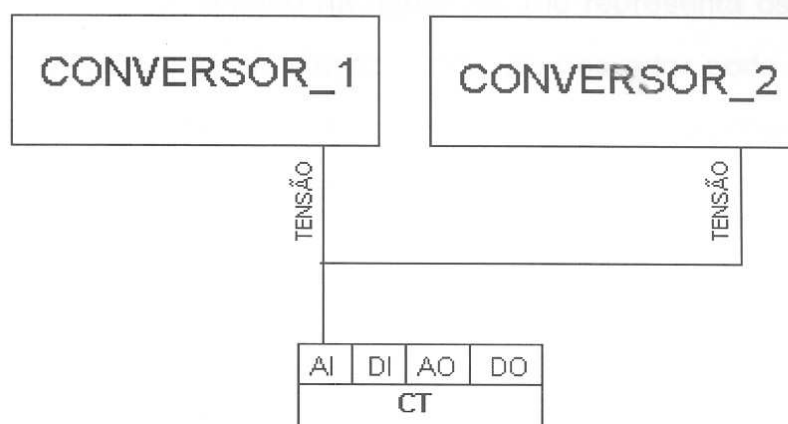
O objetivo deste capítulo é validar a arquitetura definida no capítulo anterior através da definição, implementação e validação de uma aplicação de CORBA no gerenciamento da infra-estrutura de telecomunicações. Portanto, é desenvolvido um sistema de software distribuído, baseado em CORBA, capaz de simular funções de gerenciamento de equipamentos da infra-estrutura de telecomunicações sobre uma rede de computadores. Como são diversos os elementos que compõe a infra-estrutura de telecomunicações, foge do escopo desta simulação a abordagem de todos os componentes gerenciáveis. Procura-se portanto validar os mecanismos mais significativos da arquitetura proposta no capítulo anterior. Este capítulo inicia com a especificação da aplicação na seção 6.2. Na seqüência é apresentada a concepção do sistema simulado na seção 6.3 e a definição das interfaces IDL na seção 6.4. No final são apresentados os resultados relevantes.

6.2 DEFINIÇÃO DOS SUBSISTEMAS A SEREM GERENCIADOS

O primeiro passo é a definição de um estudo de caso que seja simples e que avalie a aplicação de arquitetura proposta no gerenciamento da infra-estrutura. Neste estudo de caso são estipulados dois subsistemas a serem gerenciados, um que será instalado em uma máquina remota e um outro que está conectado à mesma máquina, onde roda o sistema gerente.

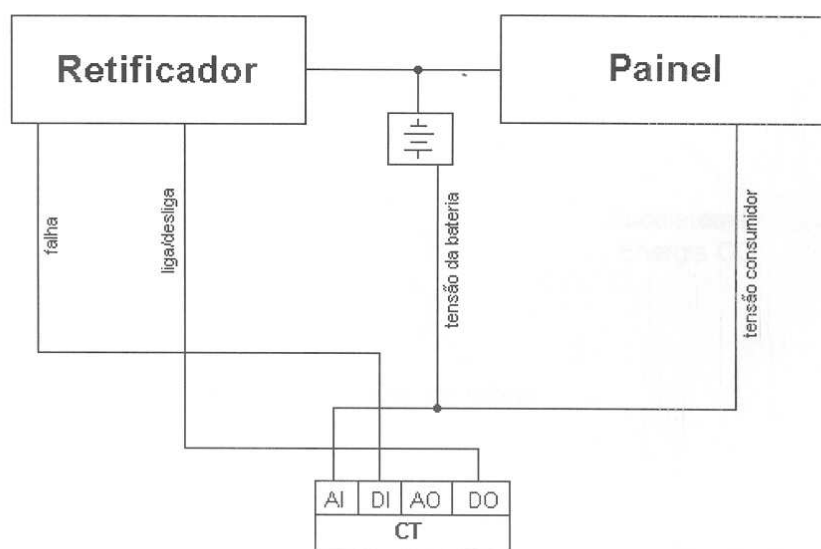
O subsistema remoto é similar ao que foi modelado na seção 5.4.3, ou seja, o subsistema de energia CC com algumas simplificações, visando resumir a quantidade de parâmetros gerenciados. As figuras 33 e 34 apresentam os subsistemas a serem gerenciados e seus parâmetros. O equipamento retificador possui apenas os parâmetros defeito e tensão do consumidor e possibilita as ações de ligar e desligar. O equipamento bateria possui o parâmetro tensão da bateria. Para estes equipamentos estão associadas funcionalidades de gerenciamento de configuração, alarme e desempenho. O subsistema local é composto por dois

conversores DC/DC , sendo que é possível ler apenas parâmetros de tensão destes equipamentos, ligá-los ou desligá-los.



AI - entrada sinais analógicos
 DI - entrada sinais digitais
 AO - saída sinais analógicos
 DO - saída sinais digitais

Figura 33 - Subsistema local



AI - entrada sinais analógicos
 DI - entrada sinais digitais
 AO - saída sinais analógicos
 DO - saída sinais digitais

Figura 34 – Subsistema remoto

6.2.1 Casos de uso do sistema de gerenciamento

Utilizando a metodologia de casos de uso da UML[29], foi elaborado um diagrama que está apresentado na figura 35. Ele representa os casos de uso que serão implementados pela simulação, onde um operador pode alterar os atributos dos objetos gerenciáveis relativo ao subsistema remoto de energia CC, assim como os parâmetros gerenciáveis dos equipamentos do subsistema local.

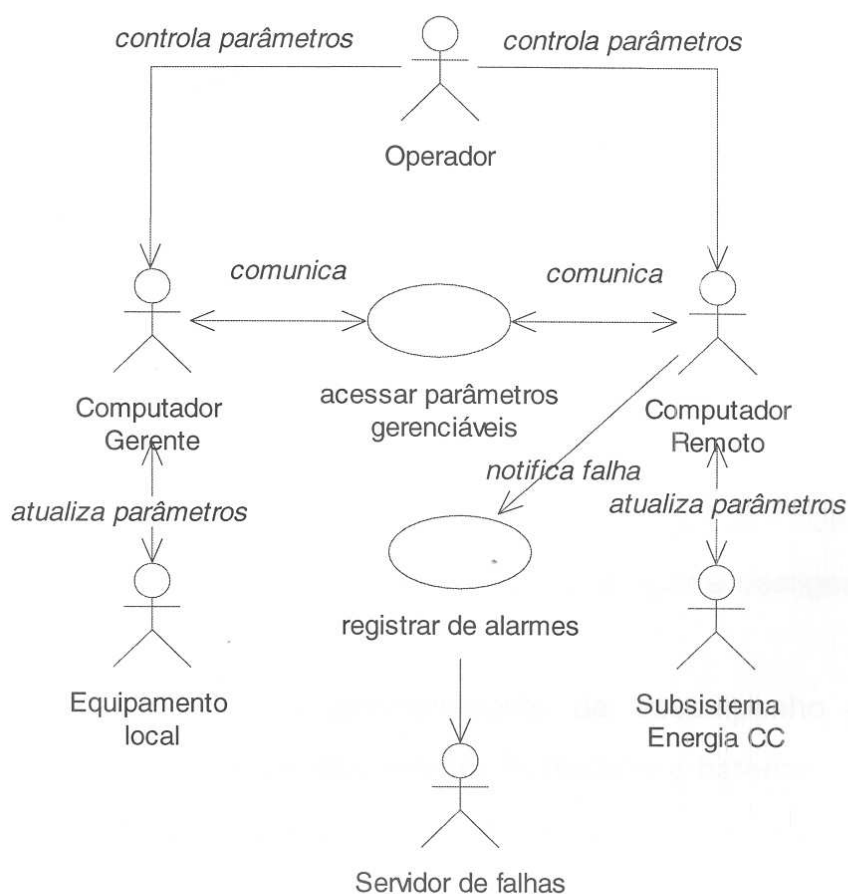


Figura 35 – Diagrama de caso de uso simulado

Os atributos alterados no subsistema, seja ele remoto ou local, são visíveis pelo computador gerente através do acesso e apresentação na tela dos parâmetros

gerenciáveis. Isso permite avaliar a transparência do sistema com relação à evocação de métodos locais e remotos.

Quando um evento de falha ocorrer no subsistema de Energia CC, o computador remoto notifica esta falha para um servidor de falha que registra a mesma. O servidor de falhas pode estar em qualquer máquina da rede e apenas registra a ocorrência da falha no retificador, pois este é o único equipamento que gera este tipo de alarme.

6.3 CONCEPÇÃO DA SIMULAÇÃO DO SISTEMA DE GERENCIAMENTO

Um visão geral do sistema a ser simulado é apresentada na figura 36, onde o componente Gerenciador é um software cliente que roda em uma máquina da rede, como por exemplo no computador 1, e pode realizar as seguintes tarefas:

- ◆ Realizar o bind com um objeto remoto através do ORB.
- ◆ Ler dados de entrada a partir do teclado e alterar os parâmetros de gerenciamento dos objetos que representam os equipamentos dos subsistemas gerenciados.
- ◆ Efetuar funcionalidades de controle e configuração dos equipamentos conectados a ele, como por exemplo, funções de ligar e desligar os conversores CONV_1 e CONV_2 .
- ◆ Efetuar funcionalidades de gerenciamento de desempenho e configuração, medindo parâmetros de objetos remotos (retificador e bateria).

O componente Subsistema é um software que inicializa o servidor, podendo rodar em qualquer máquina da rede, como por exemplo no computador 2, realizando as seguintes tarefas:

- ◆ Inicializar ORB
- ◆ Inicializar o BOA
- ◆ Ler dados de entrada a partir do teclado e passar para os objetos que representam os equipamentos do subsistema que estaria conectado ao computador 2, ativando e recuperando falhas, alterando parâmetros, etc.

- ♦ Instanciar os objetos e registrá-los no BOA . dos computadores, visando
- ♦ Atender às requisições oriundas dos clientes.

O componente Sup_fal é um servidor que tem como função registrar os alarmes notificados pelos equipamentos remotos, podendo rodar em qualquer máquina de rede e realizando as seguintes tarefas:

- ♦ Inicializar ORB
- ♦ Inicializa o BOA
- ♦ Atender às requisições de registro de falhas.

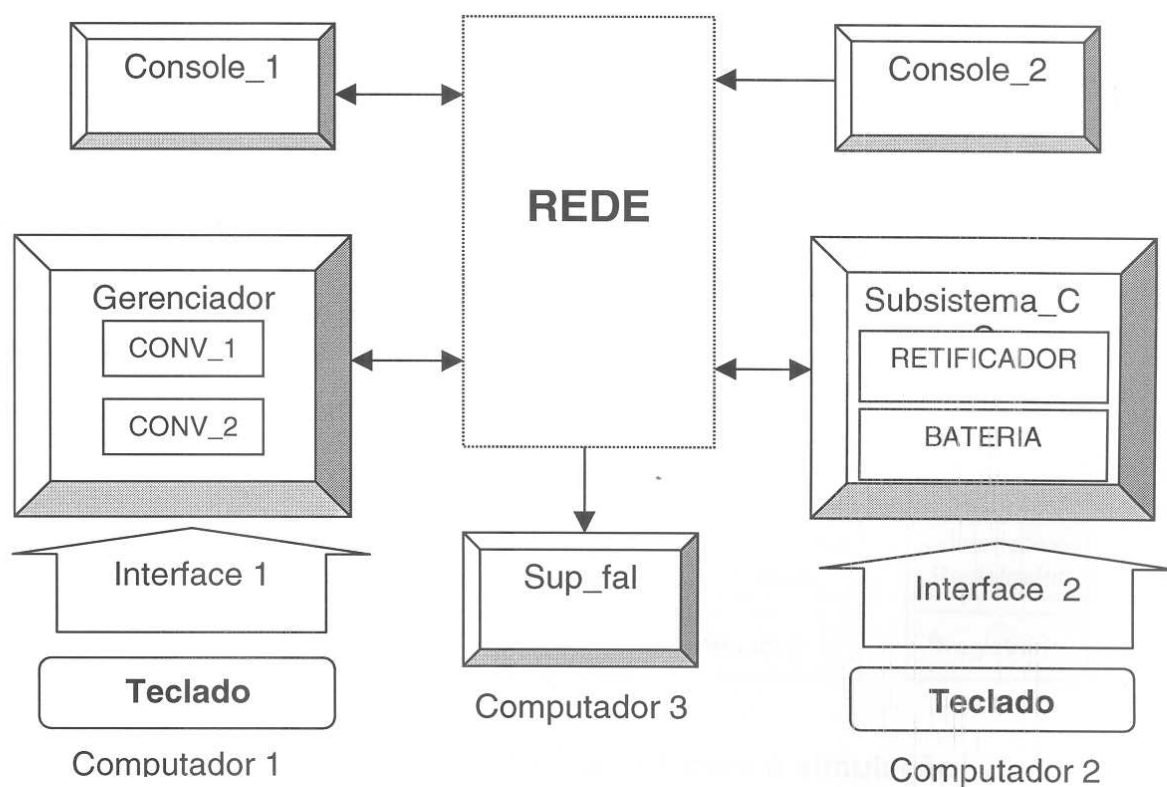


Figura 36 - Visão geral do sistema simulado

Por simplificação, este servidor não registra a identidade do equipamento que notificou a falha. Entretanto, uma implementação que faça isso é trivial. Bastaria passar um argumento de identificação com a notificação. Na simulação o registro é feito apenas na tela do computador 3, fugindo do escopo deste trabalho implementar a conexão com um banco de dados para armazenamento das falhas. Para isso seria necessário ainda a especificação e implementação de um módulo

capaz de sincronizar os relógios de tempo real dos computadores, visando associar as falhas com data e hora de ocorrência.

Foram implementados e testados dois programas que poderiam vir a serem úteis para um sistema real:

O primeiro, denominado de Console_1, implementa um método de invocação dinâmica e faz uso do repositório de interfaces. Este programa acessa os parâmetros gerenciáveis dos objetos remotos de forma dinâmica, utilizando o repositório de interfaces.

O segundo, denominada de Console_2, simula a geração de alarmes, sendo um módulo que pode ser instalado em qualquer ponto da rede e utilizada por um outro subsistema.

6.3.1 Projeto das interfaces

Nesta seção é desenvolvido um modelo para o sistema a ser simulado. O primeiro passo é a definição das interfaces IDL que serão necessárias para o gerenciamento do sistema remoto.

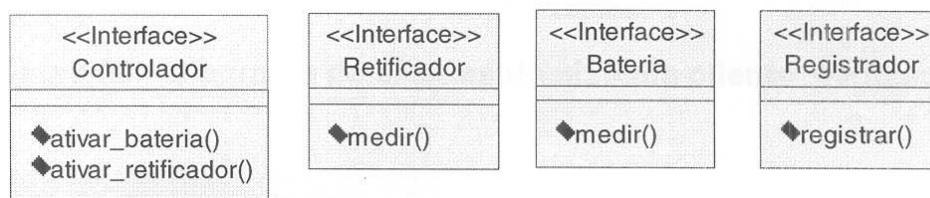


Figura 37 - Interfaces IDL para a simulação

A interface Controlador especifica dois métodos, ativar_retificador e ativar_bateria, que possibilitam o registro dos objetos no adaptador de objetos. Tanto a interface Bateria quanto Retificador especificam o método medir(), o qual é evocado remotamente pelo sistema Gerenciador. A última interface necessária para a simulação é a Registrador, que é utilizada para o gerenciamento de alarmes.

6.3.2 Projeto do Sistema Gerenciador

O modelo para o sistema gerenciador é apresentado na figura 38, sendo composto por um módulo cliente e um servidor. O módulo cliente possui um método para realizar a ligação ou *bind* com os objetos remotos retificador e bateria, assim com instancia objetos do tipo Conversor. O sistema foi modelado desta forma para verificar a transparência do sistema em evocar métodos locais fora do ORB e métodos remotos através de evocação por stub do cliente.

O módulo servidor é implementado pela classe Registrador e possui o método registrar(). Observa-se que este o Registrador não possui nenhuma ligação ou associação com a classe Gerenciador.

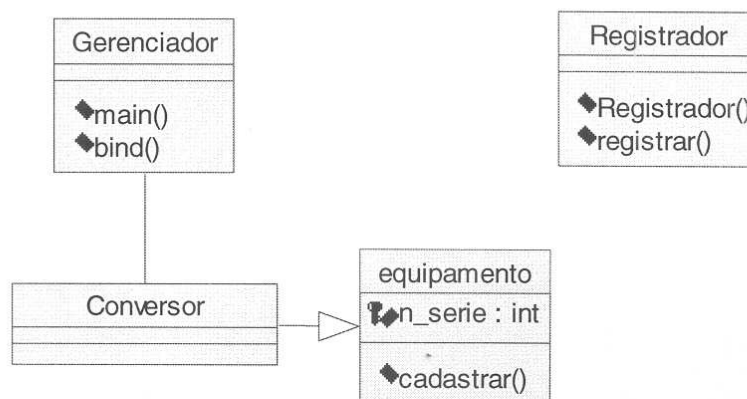


Figura 38 - Diagrama de classes do sistema cliente simulado

6.3.3 Projeto do Sistema Gerenciado

O modelo do sistema servidor é apresentado na figura 39 e segue a proposta do capítulo anterior, seção 5.4.2. Observa-se claramente que se trata do mesmo diagrama para o subsistema de energia CC, porém com menos parâmetros gerenciáveis. Essa simplificação não compromete a validação do modelo, pois o objetivo é a validação dos mecanismos para o gerenciamento da infra-estrutura de telecomunicações.

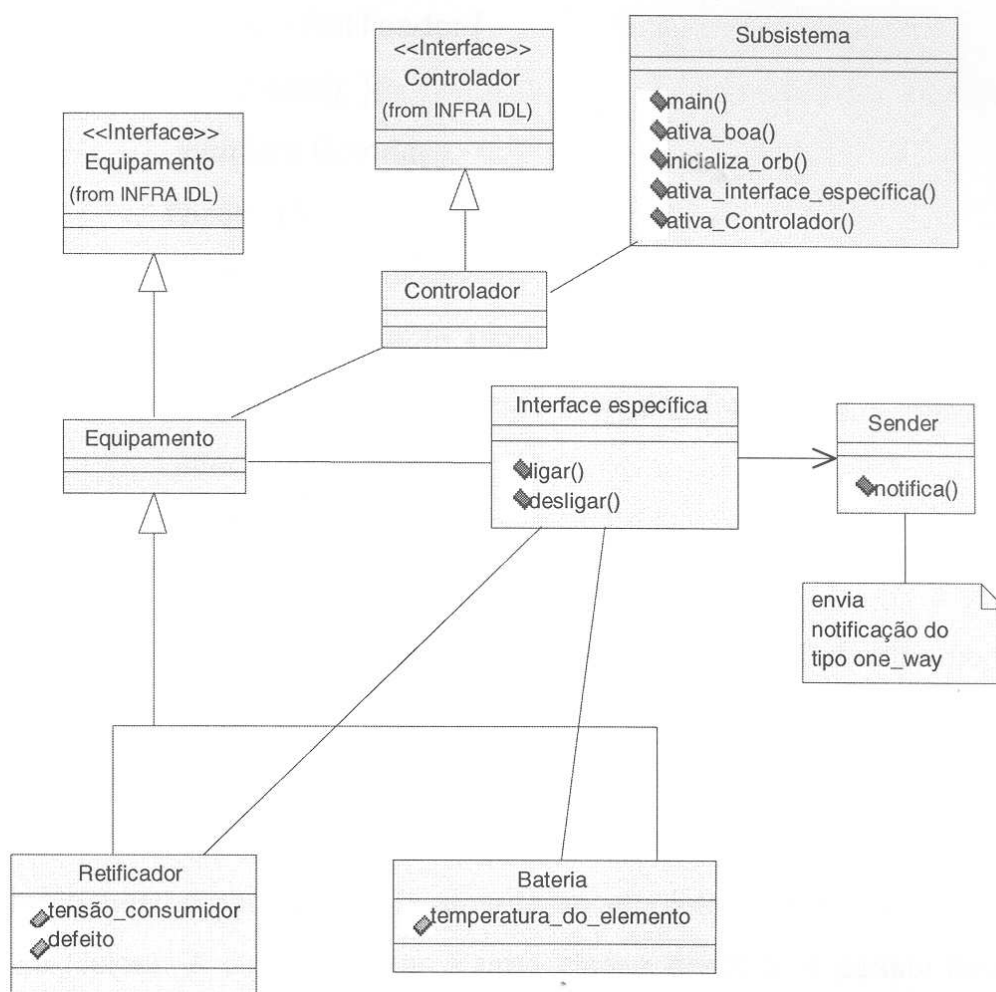


Figura 39 - Diagrama de classes do sistema servidor

6.4 IMPLEMENTAÇÃO DAS INTERFACES IDL – *INTERFACE DEFINITION LANGUAGE*

A definição de interfaces CORBA, codificada em IDL, estipula as características e comportamentos dos objetos gerenciados. A IDL é utilizado para identificar as interfaces (objetos e operações associadas) que os clientes utilizarão para realizarem requisições de tarefas e as interfaces que os servidores possuirão para executarem as tarefas. As definições IDL também tornam possível o mapeamento de objetos CORBA para linguagens de programação particulares como, C, C++, e JAVA.

A seguir está a definição IDL utilizada para a simulação:

```

module Infra {
    interface Retificador {
        float medir(); };
        interface Bateria{
            float medir();};
        interface Controlador {
            Retificador ativar(in string nome_do_gerente);
            Bateria ativar_2(in string nome_do_gerente); };
        interface Registrador {
            void registrar(); };
};

```

Como pode-se observar, as quatro interfaces IDL definidas no modelo da figura 39 estão descritas acima. Este arquivo IDL foi compilado, gerando os stubs do sistema Gerenciador e os esqueletos para os objetos gerenciados.

O compilador utilizado foi o `idl2java`, fornecido juntamente com o Visibroker, sendo que o mesmo gera automaticamente as seguintes classes[41]:

- ◆ Classe Helper: A classe Helper é uma classe abstrata e possui métodos para operações nos objetos associados. Ela contém o método `bind()`, que faz a ligação com os objetos remotos.
- ◆ Classe Holder: Como a linguagem JAVA permite somente que os parâmetros sejam passados por valor e não por referência, as classes Holder são utilizadas para passar parâmetros de entrada e entrada/saída associados com operações de requisições[42].
- ◆ Classe Stub: A classe Stub fornece a implementação para a interface que o cliente chama pelo nome.
- ◆ Classe ImplBase: Esta classe é a classe Skeleton do servidor a partir da qual uma implementação herda as propriedades.
- ◆ Classe Tie: Essa classe herda da classe ImplBase e delega cada chamada para a implementação do objeto respectivo.

- ♦ Classe Operations: Essa classe define todos os métodos que devem ser implementados pelo objeto de implementação.
- ♦ Classe Example: Essa classe fornece códigos que devem ser complementados para implementação de um objeto.

A figura 40 apresenta o diagrama destas classes para uma das interfaces, sendo que ele se repete para as demais.

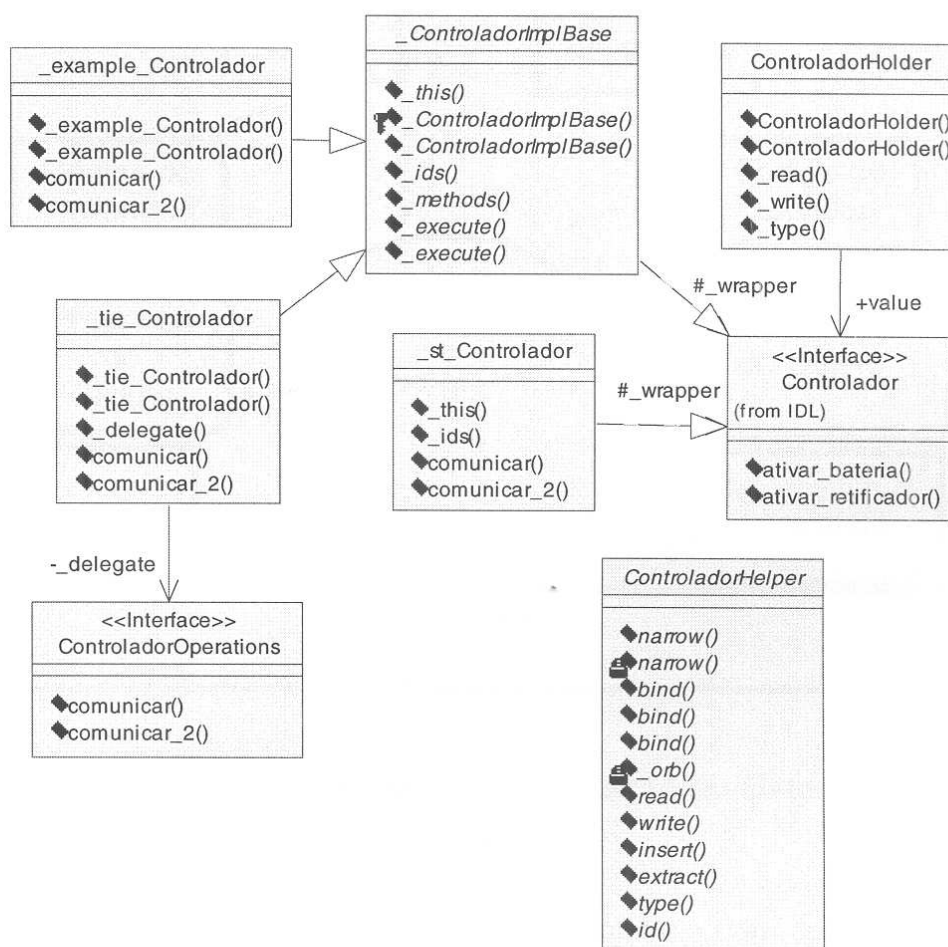


Figura 40 - Diagrama das classes geradas a partir do compilador IDL

6.4.1 Implementação do sistema

A figura 41 ilustra a seqüência seguida para a implementação do sistema simulado.

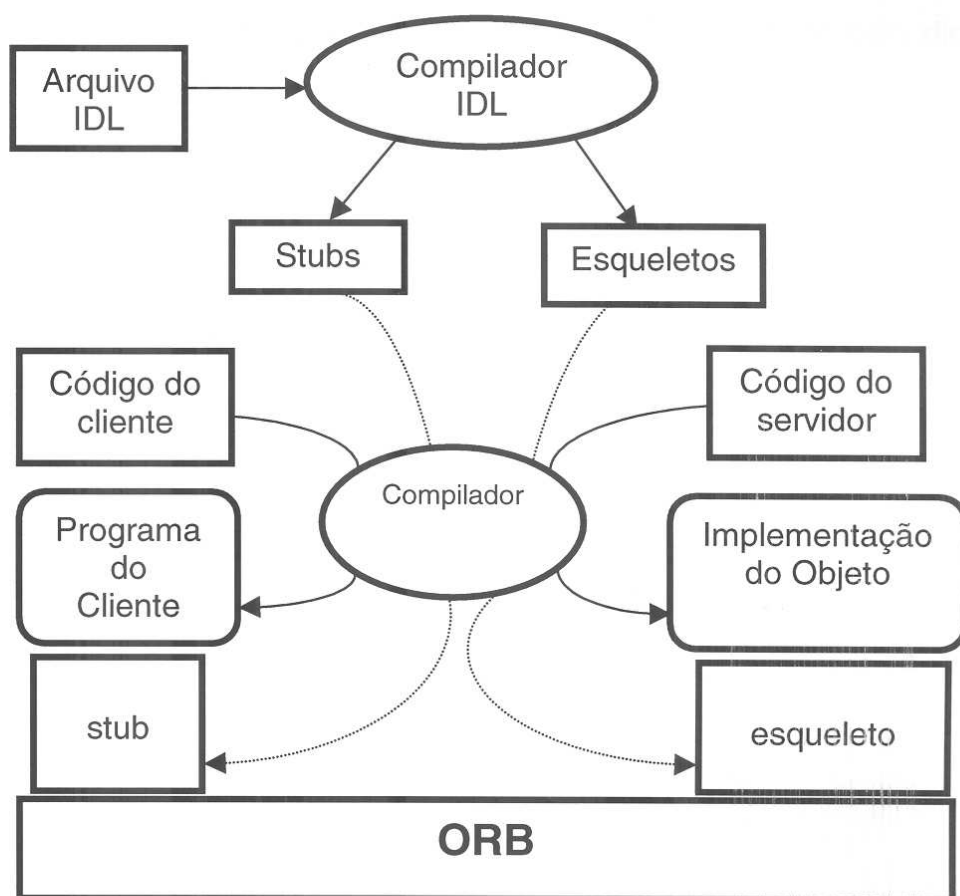


Figura 41 – Metodologia para implementação do sistema simulado

Os seguintes arquivos de códigos, disponíveis com comentários no anexo B, foram implementados e compilados:

- ♦ Gerenciador.java: implementa o programa cliente
- ♦ Subsistema_1.java: implementa o programa servidor
- ♦ Bateria.java: implementa a classe Bateria
- ♦ Retificador.Java: implementa a classe Retificador
- ♦ ControladorImpl.Java: implementa a classe Controlador

- ♦ Sender.Java: Implementa a classe Sender para notificação de alarmes no estilo one_way.
- ♦ Sup_Fal.Java: implementa a classe Sup_Fal
- ♦ Registrador
- ♦ Console_1.Java: implementa o programa cliente para evocação dinâmica aos objetos gerenciáveis
- ♦ Console_2.Java: implementa o programa cliente para envio de falhas por evocação dinâmica.
- ♦ Equipamento.Java: superclasse para a classe conversor
- ♦ Classe.conversor: implementa a classe Conversor

A linguagem de programação utilizada para desenvolvimento dos clientes e dos servidores foi JAVA e o ORB que serviu de base para a implementação foi o Visibroker, da Inprise.

Foram utilizadas ferramentas auxiliares como o ROSE, da Rational, para a modelagem do sistema, o ambiente Visual Studio para a edição do código e o JDK 1.1.6, da Sun, como ferramenta de compilação.

6.4.2 Limitações da simulação

Um software para gerenciamento integrado da infra-estrutura de telecomunicações é muito complexo e necessita interagir com vários subsistemas e sistemas de gerenciamento. A aplicação desenvolvida utiliza um pequeno número de transações se comparada com todas as possibilidades possíveis para sistemas reais. Em um sistema real, a base de dados com os históricos de gerenciamento estaria localizada em máquinas específicas e atualizada permanentemente. Na simulação esta funcionalidade não é contemplada, dando-se ênfase aos mecanismos de invocação em CORBA e na implementação das funcionalidades de gerenciamento de configuração, desempenho e alarme.

O nível de gerenciamento 2 requisita ainda o gerenciamento de segurança, porém esta facilidade não foi incluída na simulação. Apesar de CORBA especificar o serviço de segurança, o produto CORBA comercial utilizado no desenvolvimento do

sistema, não disponibiliza este serviço no seu pacote de avaliação distribuído pela Internet.

Conforme apresentado no capítulo 2, as interfaces entre os subsistemas e os computadores tendem a ser específicas, ficando fora do escopo desta simulação. Portanto, considera-se para efeitos de simulação, que os dados que seriam provenientes destas interfaces, sejam obtidos pelo teclado das máquinas que hospedam os sistemas cliente e servidor.

Embora exista uma variedade de transportes de rede que poderiam ser considerados, a rede disponibilizada para a simulação implementou o protocolo de comunicação TCP/IP e meios de comunicação Ethernet conectada do tipo LAN 10 base T. As informações de gerenciamento trafegaram por esta rede de computadores, propiciando o gerenciamento integrado de acordo com as premissas definidas no início.

Apesar de não ter sido implementado uma interface gráfica, o sistema implementado permite que os parâmetros dos objetos gerenciados seja alterado por comandos via teclado.

E finalmente, as exceções que venham a ocorrer não são tratadas pela aplicação desenvolvida, por não se tratar de um sistema comercial.

6.5 RESULTADOS PRINCIPAIS DA SIMULAÇÃO

O sistema foi testado em uma rede local e os resultados puderam ser comprovados. Apesar do sistema ter sido desenvolvido em Windows 95, ele foi testado conforme mostra a figura 36, sobre uma rede Windows NT envolvendo 3 computadores, onde as funcionalidades de gerenciamento propostas foram verificadas:

Observou-se durante o desenvolvimento da aplicação, que o sistema de gerenciamento de infra-estrutura simulado é modular, pois possibilita alterações sem perder as funcionalidades implementadas, não sendo necessário recompilar todo o código quando novas funcionalidades são adicionadas.

Outro requisito verificado é a escalabilidade do sistema. Pois, foi possível adicionar novos recursos de software, mesmo com o sistema em operação.

Os aspectos de distribuição foram verificados, sendo que o sistema apresentou transparência às chamadas remotas. Ou seja, foi imperceptível ao operador identificar se as evocações de métodos eram remotas ou locais.

O sistema permitiu ainda a simulação das funcionalidades de gerenciamento, como gerenciamento de alarmes e configuração, proporcionando sempre uma visão integrada dos subsistemas gerenciados.

Verificou-se o funcionamento das formas de invocação dinâmica e por *binding*, constatando-se a transparência do servidor na execução dos métodos e respostas.

Portanto, o sistema simulado atendeu aos principais aspectos de gerenciamento levantados no segundo capítulo.

7 CONCLUSÃO

No segundo capítulo a infra-estrutura de telecomunicações foi apresentada e dividida em subsistemas. Foram levantados ainda os requisitos para gerenciamento da infra-estrutura de telecomunicações, considerando tanto os requisitos mais genéricos quanto as especificações oriundas de práticas Telebrás. Verificou-se ainda que, apesar de existir no mercado vários sistemas de gerenciamento comerciais, a grande maioria não está preparada para um gerenciamento verdadeiramente integrado. Esse capítulo auxiliou na avaliação das tecnologias atualmente empregada e foi fundamental para a elaboração da proposta de arquitetura baseada em CORBA.

O terceiro capítulo apresentou as principais tecnologias que são empregadas no gerenciamento de infra-estrutura de telecomunicações. Verificou-se que o protocolo CMIP agrega um número muito maior de detalhes quando comparado com SNMP, sendo compatível com o modelo OSI e especificado pela TMN para gerenciamento de redes. Entretanto, este protocolo ainda não é muito empregado no gerenciamento da infra-estrutura de telecomunicações devido à sua complexidade e elevado custo de implementação. Apesar da implementação de sistemas SNMP serem de mais simples, eles possuem menos recursos que o CMIP e rodam exclusivamente em redes TCP-IP. Portanto, tanto uma tecnologia quanto a outra não são totalmente apropriadas para o desenvolvimento de sistemas de gerenciamento de infra-estrutura.

Foram pesquisadas ainda algumas tecnologias de sistemas distribuídos e verificou-se que as funcionalidades de gerenciamento podem ser implementadas utilizando CORBA com vantagens significativas em termos de simplificação na implementação dos sistemas, sendo que existem pesquisas para a integração de CORBA com tecnologias consagradas como TMN e SNMP. Esse fato apoia a elaboração de uma arquitetura de gerenciamento de infra-estrutura de telecomunicações baseada em CORBA.

No quarto capítulo foram apresentados os principais conceitos sobre os quais CORBA se apoia, assim como os componentes e interfaces definidos pelo

OMG. Verificou-se que CORBA possui uma arquitetura bem definida e que oferece os mecanismos de comunicação necessários para o gerenciamento da infra-estrutura de telecomunicações. Observou-se ainda que boa parte dos requisitos apontados no segundo capítulo são características intrínsecas de CORBA, pois esta arquitetura se propõe a ser aberta, escalonável, distribuída e modular. Além disso, constatou-se que muitos dos serviços CORBA, definidos no COSS, possuem aplicação direta na implementação de sistemas de gerenciamento de infra-estrutura.

No capítulo 5 foi apresentada uma proposta de arquitetura para gerenciamento integrado de infra-estrutura de telecomunicações e baseada em CORBA. A proposta foi dividida em três visões: arquitetura física, funcional e de informação.

Foi desenvolvido um modelo para a arquitetura de informação, incluindo os subsistemas do grupo I e uma forma para a especialização deste modelo. Como exemplificação o modelo genérico serviu de base para um modelo específico do subsistema de corrente contínua. As tabelas apresentadas, por possuírem os parâmetros de gerenciamento classificados de acordo com as funcionalidades de gerenciamento, auxiliam na elaboração de modelos específicos para outros subsistemas.

Para comprovar a funcionalidade da arquitetura proposta foi desenvolvido, no sexto capítulo, um estudo de caso baseado no modelo específico do subsistema de corrente contínua. Verificou-se que as vantagens da computação orientada à objetos, agregadas por CORBA, favorecem o desenvolvimento de sistemas de gerenciamento de infra-estrutura, como herança e encapsulamento. Foram simuladas funcionalidades de gerenciamento em uma rede local, onde os objetos gerenciados foram distribuídos em máquinas distintas, comprovando-se que a arquitetura proposta oferece uma forma homogênea de visualização e implementação de sistemas de gerenciamento de infra-estrutura em um ambiente integrado, levando-se em consideração as funcionalidades de gerenciamento estabelecidas no capítulo dois. Observou-se ainda, que os requisitos de abertura,

modularidade, escalabilidade e distribuição são plenamente atendidos pela arquitetura proposta neste trabalho.

Portanto, os objetivos do trabalho, definidos na seção 1.1 foram plenamente atingidos, pois foram pesquisados os elementos que compõe a infra-estrutura de telecomunicações, seus parâmetros gerenciáveis e identificados os requisitos importantes para o gerenciamento da infra-estrutura de telecomunicações. Foi realizada ainda uma investigação preliminar sobre os serviços CORBA que podem ser aplicadas diretamente em sistemas de gerenciamento da infra-estrutura de telecomunicações e proposta uma arquitetura apropriada, baseada em CORBA, para o gerenciamento da infra-estrutura de telecomunicações. E finalmente a arquitetura proposta foi simulado em uma rede de computadores, utilizando objetos distribuídos.

7.1 SUGESTÕES PARA TRABALHOS FUTUROS

Com o propósito de aprimorar os resultados obtidos por este trabalho, são propostas, a seguir, algumas sugestões que poderão servir para estudo em outros trabalhos:

- a) Pesquisar novas arquiteturas para o gerenciamento integrado da infra-estrutura de telecomunicações.
- b) Pesquisar as interfaces específicas entre o sistema de gerenciamento e os subsistemas e definir uma arquitetura para integração.
- c) Pesquisar soluções para a interface homem-máquina para sistemas de gerenciamento de infra-estrutura e seus aspectos ergonômicos.
- d) Realizar um estudo mais profundo sobre a aplicação dos serviços comuns, definidos no COSS, no gerenciamento da infra-estrutura de telecomunicações.
- e) Desenvolvimento de um método para inicialização automática dos objetos gerenciados.

Anexo A – Outros Serviços CORBA [28]

A1. Serviço de Ciclo de vida

Os serviços de ciclo de vida definem convenções para a criar, apagar, copiar e mover objetos. Como o ambiente baseado em CORBA suporta objetos distribuídos, os serviços de ciclo de vida definem serviços e convenções que permitem aos clientes executarem operações sobre objetos em diferentes localidades.

O modelo de cliente para criação é definido em termos de fábricas de objetos. Uma fábrica é um objeto que cria outro objeto. Fábricas não são objetos especiais. Da mesma forma que qualquer objeto, fábricas são interfaces OMG-IDL bem definidas e implementações em alguma linguagem de programação.

A2. Serviço de transação

O serviço de transação de objetos suporta a interoperabilidade entre diferentes modelos de programação. Por exemplo, alguns usuários podem querer adicionar implementações de objetos às aplicações procedurais existentes e aumentar as implementações de objetos com código que utiliza o paradigma procedural. Para fazer isso em ambiente de transações, precisa-se que o código de objetos e procedural compartilhem uma única transação. Portanto, o serviço de transações podem facilitar a integração com sistemas legados da infra-estrutura.

A3. Serviço de perguntas

O objetivo do serviço de perguntas é permitir aos usuários e objetos invocarem perguntas em coleções de outros objetos. As perguntas são declarações com predicados e incluem a habilidade de especificar valores de atributos; de invocar operações arbitrária; e invocar outros serviços de objetos.

O serviço de perguntas permite indexação e mapea bem os mecanismos de pergunta utilizados nos sistemas de banco de dados e outros sistemas que armazenam e acessam grandes coleções de objetos, sendo baseado nos padrões de pergunta incluindo SQL-92, OQL-93 e OQL-93 básico.

A4. Serviço de licença

O serviço de licença fornece um mecanismo aos produtores de software para controlar o uso de sua propriedade intelectual. Os produtores podem implementar o serviço de licença de acordo com suas próprias necessidades, e das necessidades de seus clientes, porque o este serviço não impõe suas próprias políticas de negócio ou práticas.

Uma licença em um serviço de licenças possui três tipos de atributos que permitem aos produtores aplicarem controles flexíveis: tempo, mapeamento de valor, e consumidor. O tempo permite que as licenças tenham datas de início, duração e expiração. Mapeamento de valor permite que os produtores implementem um esquema de licenças de acordo com as unidades, alocação (através de uso concorrente de licenças). Os atributos do usuário permitem que uma licença seja reservada por designada para máquinas especiais.

A5. Serviço de propriedades

Fornece a habilidade de associar valores nomeados dinamicamente com objetos e fora do sistema estático do tipo IDL. O serviço de propriedades define operações para criar e manipular conjuntos de pares nomes/valores ou tuplas nome/valor/ modo. Os nomes são strings OMG IDL simples. Este serviço não se apoia em qualquer outro serviço de objetos.

A6. Serviço de relacionamento

Os serviços de relacionamento permitem que as entidades e os relacionamentos sejam representados explicitamente. As entidades são representadas como objetos CORBA. O serviço como dois novos tipos de objetos: relacionamentos e papéis.

O papel representa um objeto CORBA em um relacionamento. A interface de relacionamento pode ser estendida para adicionar atributos e operações específicos de relacionamento. Adicionalmente, relacionamentos de grau arbitrário

podem ser definidos. De forma similar, a interface Papel pode ser estendida para adicionar atributos e operações específicas de um papel.

Os serviços de ciclo de vida definem operações para copiar, mover e remover grafos de objetos relacionados, enquanto que os serviços de relacionamento permitem que os grafos de objetos relacionados sejam cruzados sem ativar os objetos relacionados.

A7. Serviço de exteriorização

Os serviços de exteriorização definem protocolos e convenções para exteriorização e internação de objetos. Exteriorizar um objeto é gravar o estado do objeto em um fluxo de dados (memória, arquivo de disco, sobre a rede, e assim por diante) e então ser internado em um novo objeto no mesmo ou em diferentes processos. O objeto externado pode existir por um tempo arbitrário, ser transportado para meios fora do ORB, e ser internado em um ORB diferente e desconectado.

Para efeitos de portabilidade, os clientes podem pedir que os dados externados sejam armazenados em um arquivo que tenha formato definido na especificação de exteriorização de serviços.

A8. Serviço de negociação de objetos.

O serviço de negociação de objetos fornece um serviço de *matchmaking* para os objetos. O fornecedor de serviços registra a disponibilidade do serviço invocando um operação de exportação no *trader* (negociador), passando como parâmetro informações sobre o serviço ofertado. As operações de exportação carregam uma referência de objeto que pode ser utilizada pelos clientes para invocarem operações nos serviços ofertados, uma descrição do tipo de serviço ofertado (por exemplo, os nomes das operações para as quais ele vai responder, com seus parâmetros e tipos dos resultados), informação dos atributos distinguidos dos serviços oferecidos.

A9. Serviço de coleção de objetos

Coleções são grupos de objetos que, como um grupo, suportam algumas operações e apresentam comportamento específico que são relacionados a natureza da coleção ao invés do tipo de objeto que eles contém. Exemplos de coleções são conjuntos, filas, pilhas, listas e árvores. O propósito do Serviço de Coleção de objetos é fornecer uma forma uniforme para criar e manipular a maioria das coleções genéricas.

ANEXO B – Códigos fonte da simulação

B1. Implementação do Subsistema_1

```
// Subsistema_1.java
import java.io.*;
public class Subsistema_1 {
    public static void main(String[] args) {
        // inicializa o ORB
        org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);
        // inicializa o adaptador de objetos
        org.omg.CORBA.BOA boa = orb.BOA_init();
        // cria o thread para ler teclado
        new Thread(new Interface()).start();
        System.out.println("lendo interface especifica");
        // declaracao de interfaces
        Infra.Controlador objeto_1;
        Infra.Controlador objeto_2;
        // instancia objetos
        objeto_1 = new ControladorImpl("retificador");
        objeto_2 = new ControladorImpl("bateria");
        // registra os objetos no BOA
        boa.obj_is_ready(objeto_1);
        boa.obj_is_ready(objeto_2);
        System.out.println(objeto_1 + " esta ativo.");
        System.out.println(objeto_2 + " esta ativo.");
        boa.impl_is_ready();
    }
}
```

// thread para ler interface específica

```
class Interface extends Subsistema_1 implements Runnable {
    Interface() {
        float z=0;
        RetificadorImpl.tensao = z;
    }

    public void run() {
        String msg = "";
        DataInputStream stdIn = new DataInputStream(System.in);
        while(true){
            System.out.println("Comandos:\n" +
                " medir -> mede os objetos do subsistema_1\n" +
                " des_ret -> desliga o retificador\n" +
                " lig_ret -> liga o retificador\n" +
                " falha -> retificador em estado de falha\n" +
```

```

" cons  -> retira retificador do estado de falha\n" +
" sair  -> sai do programa\n");
System.out.print ("Interface especifica ->");
try {msg = stdin.readLine();
}
catch (IOException e) {}

if(msg.equals("lig_ret") ) {
    if (RetificadorImpl.falha == false){
        RetificadorImpl.tensao = 54;
        System.out.println (" retificador ligado...");
    }

    if (RetificadorImpl.falha == true)
        System.out.println (" retificador em falha...");
}

if(msg.equals("cons") ) {
    RetificadorImpl.falha = false;
    RetificadorImpl.tensao = 54;
}

if(msg.equals("des_ret") ) {
    RetificadorImpl.tensao = 0;
    System.out.println ("retificador desligado...");
}

if(msg.equals("falha")) {
    if (RetificadorImpl.falha == true){
        System.out.println (" retificador anteriormente em
falha..");
    }

    if (RetificadorImpl.falha == false){
        RetificadorImpl.falha = true;
        RetificadorImpl.tensao = 0;
        Sender.notifica(null);
    }

}

if(msg.equals("medir") ) {
    System.out.println ("retificador = " +RetificadorImpl.tensao);
    System.out.println ("bateria = " +BateriaImpl.tensao);
}

if(msg.equals("sair"))
break;

```

```

    }
}

```

B2. Implementação do Controlador

```

// ControladorImpl.java
import java.util.*;
public class ControladorImpl extends Infra._ControladorImplBase {
    public ControladorImpl(String nome_do_gerente) {
        super(nome_do_gerente);
    }
    public Infra.Retificador ativar(String nome_do_gerente) {
        String _nome_do_gerente;
        _nome_do_gerente = nome_do_gerente;
        System.out.println ("conectado Retificador com " + _nome_do_gerente);
        Infra.Retificador objeto_1 = new RetificadorImpl();
        _boa().obj_is_ready(objeto_1);
        return objeto_1;
    }
    public Infra.Bateria ativar_2(String nome_do_gerente) {
        String _nome_do_gerente;
        _nome_do_gerente = nome_do_gerente;
        System.out.println ("conectado Bateria com " + _nome_do_gerente);
        Infra.Bateria objeto_2 = new BateriaImpl();
        _boa().obj_is_ready(objeto_2);
        return objeto_2;
    }
}

```

B3. Implementação de Retificador

```

// RetificadorImpl.java
import java.io.*;
public class RetificadorImpl extends Infra._RetificadorImplBase {
    public static float tensao;
    public static boolean falha;
    public RetificadorImpl() {
    }
    public float medir() {
        if (falha == false) {
            System.out.println ("retificador em operacao normal ... ");
        }
        if (falha == true) {
            System.out.println ("retificador em falha ... ");
            tensao = 0;
        }
        System.out.println ("tensao no retificador = " + tensao);
    }
}

```

```

        System.out.println ("alarme de falha = " + falha);
        return tensao;
    }

```

```

}

```

B4. Implementação de Bateria

```

// BateriaImpl.java
import java.io.*;
public class BateriaImpl extends Infra._BateriaImplBase {
    public static float tensao = 48;
    public BateriaImpl() {
    }
    public float medir() {
        System.out.println ("tensao na bateria = " + tensao);
        return tensao;
    }
}

```

B5. Implementação de Sender

```

// Sender.java
import java.io.*;
import org.omg.CORBA.Request.*;
public class Sender {
    public static void notifica(String[] args) {
        String registrador = "FAL";
        org.omg.CORBA.Object proxy, status;
        org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);
        {
            proxy = orb.bind("IDL:Infra/Registrador:1.0", registrador, null, null);
            org.omg.CORBA.Request request = proxy._request("registrar");
            request.set_return_type(orb.get_primitive_tc(org.omg.CORBA.TCKind.tk_objref));
            request.send_oneway();
        }
    }
}

```

B6. Implementação do Registrador de falhas

```

// RegistradorImpl.java
import java.util.*;

```



```

public class RegistradorImpl extends Infra._RegistradorImplBase {
    public RegistradorImpl(String elemento_da_infra) {
        super(elemento_da_infra);
    }
    public void registrar() {
        System.out.println ("falha registrada ");
    }
}

```

B7. Implementação de Console_1

```

// Console_1.java
// Este módulo notifica alarmes por invocação dinâmica
public class Console_1 {
    public static void main(String[] args) {
        if(args.length != 0) {
            System.out.println("Uso: vbj Interface_1\n");
            return;
        }
        String controlador = "retificador";
        String objeto_remoto = "retificador";
        org.omg.CORBA.Object proxy, status;
        org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);
        {
            proxy = orb.bind("IDL:Infra/Controlador:1.0", controlador, null, null);
            request.add_in_arg().insert_string(objeto_remoto);
            request.set_return_type(orb.get_primitive_tc(
                org.omg.CORBA.TCKind.tk_objref));
            request.invoke();
            status = request.result().value().extract_Object();
            {
                org.omg.CORBA.Request request = status._request("medir");
                request.set_return_type(orb.get_primitive_tc(
                    org.omg.CORBA.TCKind.tk_float));
                request.invoke();
                float tensao = request.result().value().extract_float();
                System.out.println ("A medida atraves de invocacao dinamica em " +
                    objeto_remoto + " = " + tensao);
            }
        }
    }
}

```

B7. Implementação de Console_2

```

// Esta é uma interface para envio de falhas por invocação dinâmica em estilo de
//comunicação one-way
public class Console_2 {

```

```

    public static void main(String[] args) {
        if(args.length != 0) {
            System.out.println("Uso: vbj Interface_2 \n");
            return;
        }
        String registrador = "FAL";
        org.omg.CORBA.Object proxy, status;
        org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);{
            System.out.println("notificando falha pela Interface_2\n");
            proxy = orb.bind("IDL:Infra/Registrador:1.0", registrador, null, null);
            org.omg.CORBA.Request request = proxy._request("registrar");
            request.set_return_type(orb.get_primitive_tc(org.omg.CORBA.TCKind.tk_objref));
            request.send_oneway(); }
        }
    }
}

```

B8. Implementação de Sup_fal

```

//Sup_fal.java
// Inicializa componente registrador de falhas
import java.io.*;
public class Sup_fal {
    public static void main(String[] args) {
        // inicializa o ORB
        org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);
        // inicializa o adaptador de objetos
        org.omg.CORBA.BOA boa = orb.BOA_init();
        Infra.Registrador objeto_fal;
        // instancia o objeto_fal a partir da classe RegistradorImpl
        objeto_fal = new RegistradorImpl("FAL");
        // registra o objeto_scm no ORB
        boa.obj_is_ready(objeto_fal);
        System.out.println(objeto_fal + " esta pronto.");
        boa.impl_is_ready();
    }
}

```

B9. Implementação do Gerenciador

```

// Arquivo fonte: Gerenciador.java
import java.io.*;
public class Gerenciador {
    public static void main(String args[]) {
        String nome_do_gerente = "Gerenciador_1";
        Infra.Retificador equipamento_1;
        Infra.Bateria equipamento_2;
        float medida_1;
        float medida_2;
    }
}

```

```

//inicializa o ORB
org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);
//localiza os objeto remotos
Infra.Controlador objeto_remoto_1 =
Infra.ControladorHelper.bind(orb, "retificador");
Infra.Controlador objeto_remoto_2 =
Infra.ControladorHelper.bind(orb, "bateria");
// construtor de objetos remotos
equipamento_1 = objeto_remoto_1.ativar(nome_do_gerente);
equipamento_2 = objeto_remoto_2.ativar_2(nome_do_gerente);
//construtor de objetos locais
conversor Conversor_Obj = new conversor("CONV_1", 48);
conversor Conversor_Obj2 = new conversor("CONV_2", 12);
String entrada = "";
DataInputStream stdIn = new DataInputStream(System.in);
while(true){
    System.out.println("Comandos:\n" +
        " ligar          -> liga conversores 1 e 2\n" +
        " ligar CONV_1      -> liga conversor 1 \n" +
        " ligar CONV_2      -> liga conversor 2\n" +
        " desligar CONV_1   -> desliga conversor 1\n" +
        " desligar CONV_2   -> desliga conversor 2\n" +
        " medir retificador -> mede retificador remoto\n" +
        " medir retificador -> mede retificador remoto\n" +
        " medir bateria     -> mede bateria remota\n" +
        " medir             -> mede todos os elementos\n" +
        " sair              -> sai do programa\n");
    System.out.print ("Gerenciador ->");
    try {entrada = stdIn.readLine();
    }
    catch (IOException e) {}

    if(entrada.equals("ligar") ) {
        int x = Conversor_Obj.ligar();
        Conversor_Obj.tensao = x;
        Conversor_Obj2.tensao = x;
        entrada= "medir";
    }
    if(entrada.equals("ligar CONV_1") )
    {
        Conversor_Obj.tensao=52;
        entrada= "medir";
    }
    if(entrada.equals("ligar CONV_2") )
    {
        Conversor_Obj2.tensao=14;
        entrada= "medir";
    }
    if(entrada.equals("desligar CONV_1"))

```

```

{
    Conversor_Obj.tensao=0;
    entrada= "medir";
}
if(entrada.equals("desligar CONV_2"))
{
    Conversor_Obj2.tensao=0;
    entrada= "medir";
}
if(entrada.equals("desligar"))
{
    Conversor_Obj.tensao=0;
    Conversor_Obj2.tensao=0;
    entrada= "medir";
}
// chamada para medir retificador
if(entrada.equals("medir retificador"))
{
    //retorna o estado do objeto retificador
    medida_1 = equipamento_1.medir();
    System.out.println("retificador = "+ medida_1);
}
// chamada para medir bateria
if(entrada.equals("medir bateria"))
{
    //retorna o estado do objeto bateria
    medida_2 = equipamento_2.medir();
    System.out.println("bateria = "+ medida_2);
}
if(entrada.equals("medir"))
{
    System.out.println("tensao em "+ Conversor_Obj.nome +" = "
    +Conversor_Obj.medir());
    System.out.println("tensao em "+ Conversor_Obj2.nome +" = "
    +Conversor_Obj2.medir());
    medida_1 = equipamento_1.medir();
    System.out.println("retificador = "+ medida_1);
    medida_2 = equipamento_2.medir();
    System.out.println("bateria = "+ medida_2);
}
if(entrada.equals("sair")) break;
}
}

```

B10. Implementação de equipamento

```
// equipamento.java
//
import java.io.*;
public class equipamento{
    int n_serie;
    public void cadastrar(){
    }
}
```

B11. Implementação de conversor

```
// conversor.java
public class conversor extends equipamento {
    String nome;
    int tensao;
    conversor(String n, int a) {
        nome = n ;
        tensao = a; }
    void conversor() {
        System.out.print("O nome do conversor e " + nome);
        System.out.println("Sua tensao de entrada e" + tensao );
    }
    int ligar(){
        int tensao2 = 50;
        return tensao2;
    }

    int medir()
    {
        return tensao;
    }
}
```

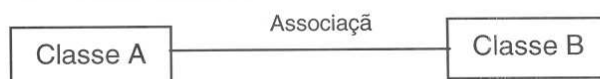
ANEXO C – Glossário UML

Agregação Uma forma especial de associação que especifica uma relação entre um todo e uma parte componente.



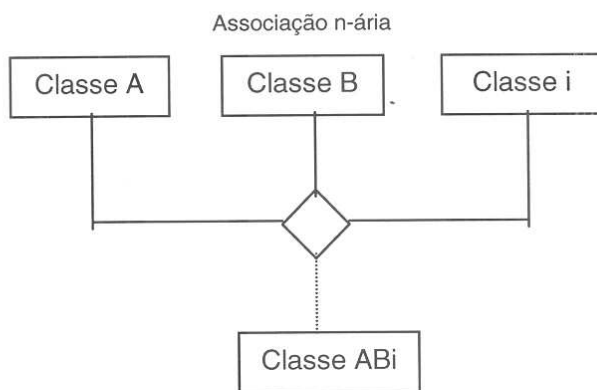
Agregação regular

Associação Uma relação que descreve um conjunto de vínculos entre elementos do modelo.



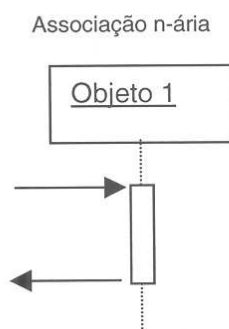
Associação n-ária

Uma associação entre três ou mais classes.

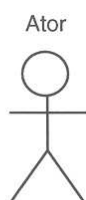


Ativação

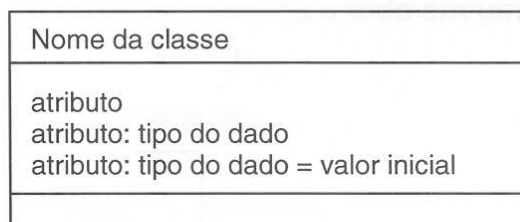
Execução de uma ação



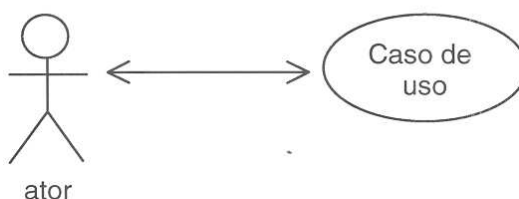
Ator Um estereótipo predefinido que denota um agente fora do sistema que interage em casos de uso



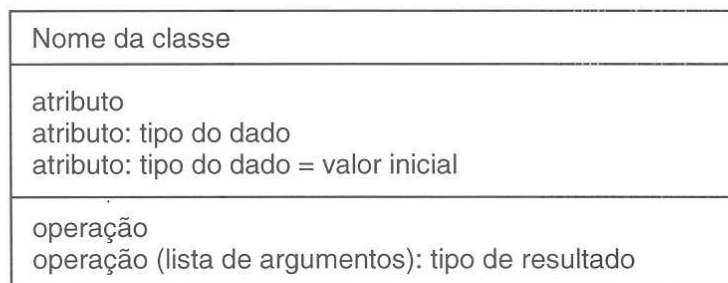
Atributo Uma propriedade nomeada de um tipo



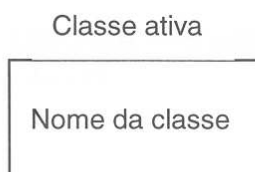
Caso de uso Uma descrição de ações de um sistema mediante o recebimento de um tipo de requisição de usuário.



Classe Uma descrição de um conjunto de objetos que compartilham os mesmos atributos, operações, relações e semântica.



Classe ativa Uma classe cujas instâncias são objetos ativos



Classe de agregação

Uma classe que representa o todo em um relacionamento de agregação (todo/parte).

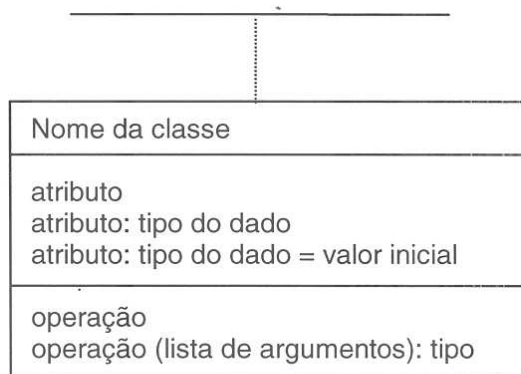
Classe de agregação



Classe de associação

Um elemento de modelagem que tem associação e propriedades de classe. Pode ser vista como uma associação que tem propriedades de classe ou uma classe que tem propriedades de associação.

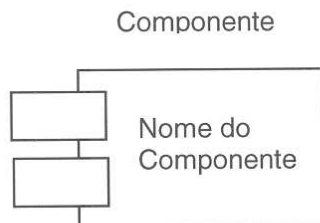
Classe de associação



Cliente Um tipo, uma classe ou um componente que requisita um serviço de outro tipo, classe ou componente.



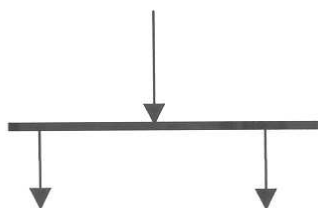
Componente Um módulo de software executável com identidade e uma interface bem definida.



Composição Uma forma de agregação com forte propriedade e vida coincidente da parte com o todo. Podem ser criadas partes com multiplicidade não fixa depois da própria combinação, mas uma vez criadas elas vivem e desaparecem com isso. Tais partes também podem ser removidas explicitamente antes do fim da combinação. A combinação pode ser recursiva e uma agregação composta é sinônimo de composição.

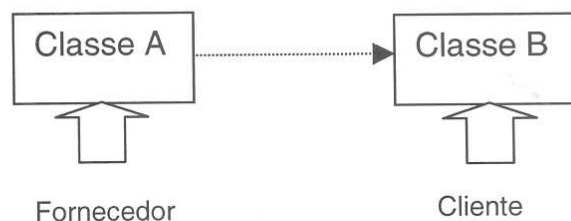


Concorrência Ocorrência de duas ou mais atividades durante o mesmo intervalo de tempo. A concorrência é alcançada executando-se duas ou mais linhas de procedimento simultaneamente.

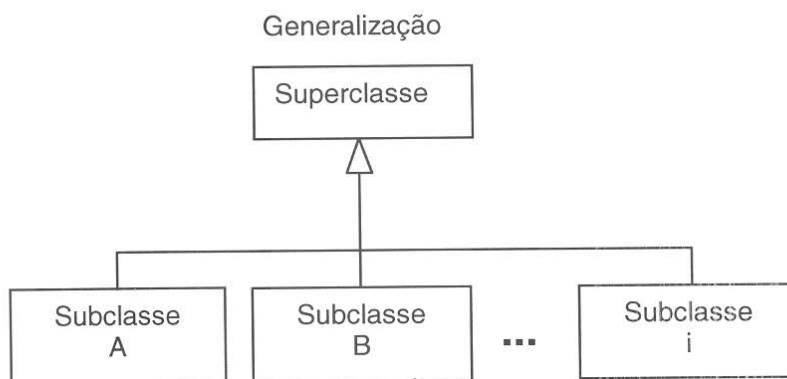


Dependência

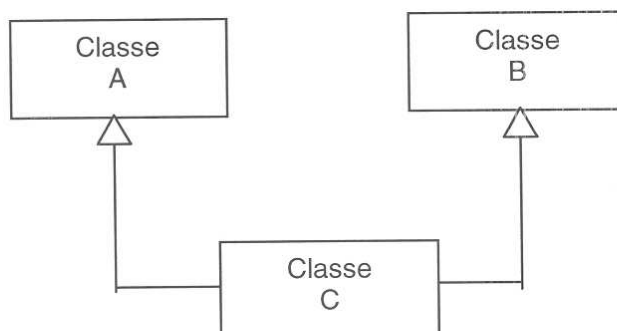
Um relacionamento entre dois elementos de modelagem no qual uma mudança em um elemento (elemento independente) afetará outro elemento (elemento dependente).

**Generalização**

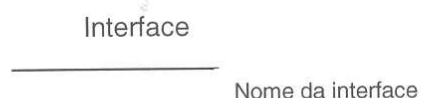
Relacionamento de taxinomia entre um elemento mais geral e um elemento mais específico. O elemento mais específico é completamente consistente com o elemento mais geral e contém informação adicional. Uma instância do elemento mais específico é usada onde o elemento mais geral for permitido.

**Herança múltipla**

Variação semântica de generalização na qual um tipo pode ter mais de um supertipo (ou uma subclasse ter mais de uma superclasse)

**Interface**

Descreve o comportamento externamente visível de um tipo – classe, componente, pacote, nó etc. – consistindo em um conjunto de operações e assinaturas com semântica dinâmica.

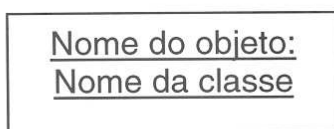


Multiplicidade Uma especificação do alcance de cardinalidades permissíveis que um conjunto pode assumir. Essencialmente uma multiplicidade é um subconjunto dos inteiros não negativos ou um intervalo infinito positivo.



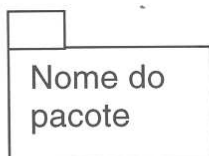
Objeto Coisa do mundo real com limite bem definido e identidade que encapsula atributos e operações, sendo uma instância de uma classe.

Objeto

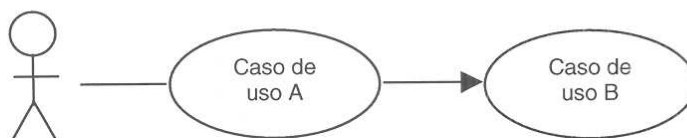


Pacote Um mecanismo de propósito geral para organizar elementos em grupos. Podem estar aninhados dentro de outros pacotes. Um sistema pode ser pensado como um único pacote de alto nível com tudo sobre si mesmo.

Pacote



Usos Um relacionamento de um caso de uso concreto para um caso de uso abstrato no qual o comportamento definido para o caso de uso concreto emprega o comportamento definido para o caso de uso abstrato.



Referências Bibliográficas

- [1] BERTHOLDI, Carlos Eduardo; PRECOMA, Acir. TMN para Gerenciamento de Redes SDH. Curitiba, 1997. Monografia (Especialização em Telecomunicações) – Pro-reitoria de pesquisa e extensão, Pontifícia Universidade Católica do Paraná.
- [2] BERTHOLDI, Carlos Eduardo; BETINI, Roberto Cesar; JAMHOUR, Edgard. SNMP Aplicado aos Sistemas de Energia de Telecomunicações. Artigo não publicado, Curitiba, mai.1997.
- [3] BETZ, Mark. OMG`s CORBA. Objects. Dr. Dobb`s, San Mateo, v19, n16, p.8–12, winter, 1995.
- [4] BLAIR, Gordon. Object Oriented Languages, Systems and Applications. UK:Longman Group.1991. p. 66-67.
- [5] BRISA, Sociedade brasileira para interconexão de sistemas abertos. Gerenciamento de redes. São Paulo: Makron Books, 1993.
- [6] CHEN, Graham; NEVILLE, Michael; KONG, Qinzhen. Distributed Network Management Using CORBA/TMN, 7th IFIP/IEEE International Workshop on distributed Systems Operation and Management (DSOM), 1996 .
- [7] CHEN, Graham. Building Integrated TMN Solutions Using Multiple Technologies-An Introduction. Network Management Forum's General Meeting, Long Beach, 1997.
- [8] COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. Distributed Systems: concepts and design. Padstow: Addison-Wesley, 1995.
- [9] DOEDELEIN, Osvaldo Pinali; GUAPO, Marco Antonio; PEREIRA, Wellington Alves. OD na Prática – Aplicação Distribuída com uso de JAVA e CORBA. Objetos Distribuídos`97, 1997. 126 transparências:p&b.
- [10] FORBICI, Eduardo Sudol. Proposta para construção de um módulo de mapeamento entre RPC/IDL e CMIP/GDMO. Curitiba, 1997. Dissertação (Mestrado em Ciências) – Centro Federal de Educação Tecnológica do Paraná.

- [11] FRANCISCO, Henrique Rolfsen. Orientação a Objetos e Computação Distribuída. Revista Developers`Magazine, São Paulo, n. 12, p. 26-28, ago. 1997.
- [12] FURLAN, José Davi. Modelagem de objetos através da UML. São Paulo: Makron Books, 1998.
- [13] GEMSTONE. Gemstone Role in CORBA Systems, white paper, www.gemstone.com, 1997.
- [14] GONSALVES, Iran Lima; MILO, Sérgio. Descrições Gerais de Supervisão para Alimentação da Rede Óptica de Assinantes. Revista TELEBRÁS, Brasília, p. 50-59, jul.96.
- [15] KONG, Qinzeng; BERRY, Andrew. A General Discovery System for Open Distributed Processing. Open Distributed Processing – Experiences with distributed environments, IFIP, Raymond and Armstrong, Chapman & Hall, 1995.
- [16] KONG, Qinzeng; CHEN, Graham. Integrating CORBA and TMN Environment. IEEE/IFIP Network Operations and Management Symposium, USA, 1996.
- [17] LEMAY, Laura; PERKINS, Charles L. Teach yourself java in 21 days. Indianapolis: SamsNet, 1996.
- [18] McCarty, Bill; Cassady, Luke. JAVA, Distributed Objects, Indianápolis:SAMS, 1999.
- [19] MARKLUND, Anders; LUNDIN, Ola, Internet Solutions in Energy Management, Anais do II Congresso Internacional de Infra-estrutura para Telecomunicações, p.133-141, Fortaleza, 1998.
- [20] MATHIENSEN, Gregory W. Monitoring and supervision of telecom infrastructure. Minicurso proferido no Cinintel 97, Fortaleza, 24 nov. 1997.
- [21] MATHIENSEN, Gregory W.: Monitoring and Control of Telecommunications Energy Systems, Anais do V Seminário de Infra-Estrutura e Energia do STB, p.189-205, Brasília, 1995.
- [22] MATHIENSEN, Gregory W.: The Impact of Technology Evolution on the Future of Telecommunications Energy Management, Anais do I Congresso

Internacional de Infra-estrutura para Telecomunicações, p.111-115, Fortaleza, 1997.

- [23] MILLER, Mark A . Managing internetworks with SNMP. New York: M&T Books, 1993.
- [24] MOTA, Marcelo D. Cardoso; GUIMARÃES, Martos R. Junior, Panorama da Infra-Estrutura para Telecomunicações e Perspectivas da Supervisão e Automação sob o Ponto de Vista de Padrões e Protocolos, Anais do II Congresso Internacional de Infra-estrutura para Telecomunicações, p. 121-129, Fortaleza, 1997.
- [25] MOTA, Marcelo D. Cardoso. Critérios de avaliação e Especificação de Sistemas de Automação e Gerência de Infra-Estrutura, Anais do II Congresso Internacional de Infra-estrutura para Telecomunicações, p. 121-132, Fortaleza, 1998.
- [26] MARKLUND, Anders; LUNDIN, Ola. Internet Solutions in Energy Management, Anais do II Congresso Internacional de Infra-estrutura para Telecomunicações, p. 133-141, Fortaleza, 1998.
- [27] NOGUEIRA, José Marcos Silva; ANDRADE, Márcio Migueletto de; MENEZES, Ângelo Martino. Integração de Interfaces Q3 ao SIS: Estratégia e Prática. XV Simpósio Brasileiro de Redes de Computadores, São Carlos, 1997.
- [28] OBJECT MANAGEMENT GROUP. Common Object Service Specification. 1997.
- [29] OBJECT MANAGEMENT GROUP. UML SUMMARY, version 1.1, 1997.
- [30] OTTE, Randy; PATRIK, Paul; ROY, Mark. Understanding CORBA – The Common Object Request Broker Architecture. New Jersey: Prentice-Hall, 1996.
- [31] PEREIRA, Armando Alves. Experiência da Telebrás no Desenvolvimento de Sistemas Distribuídos para Gerência de Falhas. Objetos Distribuídos'97, 1997. 36 transparências :p&b.
- [32] PERAZOLO, Marcelo. Interoperabilidade entre CORBA e Padrões de Gerência de Telecomunicações. Objetos Distribuídos'97, 1997. 129 transparências: p&b.
- [33] PERKINS, David; MCGINNIS, Evan. Understanding SNMP MIBs. New Jersey: Prentice-Hall, 1997.

- [34] PINTO, Pedro Bosco Mota. Especificações gerais para sistemas de automação de infra-estrutura predial, Anais do V Seminário de Infra-Estrutura e Energia do STB, p. 279-305, Brasília, 1995.
- [35] RAMALHO, Eduardo Antonio. Gerência Integrada de Redes e Serviços. Revista TELEBRÁS, Brasília, 7-11, jul.96.
- [36] RUMBAUGH, James; BLAHA, Michael; PREMERLANI, Willian; EDDY, Frederick; LORENSEN, Willian. Object oriented modeling and design . New Jersey: Prentice-Hall, 1994.
- [37] TELEBRÁS. Especificações Gerais para Sistemas de Supervisão e Automação de Infra-estrutura. 1995.
- [38] TRIMMEL, Peter. Is Web-based TMN on its way?. Telcom report, v. 20, p.30-31. Siemens Telecommunications. 1997.
- [39] VALDÉS, Ray. Introducing Interoperable Objects. Dr. Dobb's, San Mateo, v19, n16, p.4 -6, winter, 1995.
- [40] VALDÉS, Ray. Implementing Interoperable Objects. Dr. Dobb's, San Mateo, v19, n16, p.63, winter, 1995.
- [41] VISIGENIC Software, Inc. Programmer's guide: San Mateo, 1998.
- [42] VISIGENIC Software, Inc. Reference Manual: San Mateo, CA, 1998.
- [43] VISIGENIC Software, Inc. Distributed Object Computing in the internet Age, www.visigenic.com, 1997.
- [44] WALLBERG, Ulf; MONTENEGRO, Eduardo Pereira; OLIVEIRA, Alvaro Augusto Simões . Uma visão sobre gerenciamento em rede de equipamentos de energia, Anais do V Seminário de Infra-Estrutura e Energia do STB, p. 238-245, Brasília, 1995.
- [45] WICKSTRÖM, Robert; ENGLUND, Kerstin; THORÉN, Tony , Experiences of Energy Management Systems in Telecommunications, Anais do I Congresso Internacional de Infra-estrutura para Telecomunicações, p. 117-121, Fortaleza, 1997.