UNIVERSIDADE FEDERAL DE SANTA CATARINA

DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO E SISTEMAS

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

Modelo para Incorporar Conhecimento Baseado em Experiências à Arquitetura TMN

Tese submetida à Universidade Federal de Santa Catarina para a obtenção do título de Doutor em Engenharia de Produção

Alexandre Moraes Ramos

Florianópolis, Junho de 2000

Modelo para Incorporar Conhecimento
Baseado em Experiências àArquitetura TMN

Alexandre Moraes Ramos

Prof. Ricardo de Miranda Barcia, Ph.D.
Coordenador do Curso

BANCA EXAMINADORA

Prof. João Bosco da Mota Alves, Dr.
Orientador

Prof. Luiz Fernando Ramos Molinaro, Dr.
Examinador Externo

Prof. Marcello Thiry, Dr. Eng.
Examinador Externo

Prof. Alejandro Martins, Dr. Eng.
Examinador

Prof. Roberto Pacheco, Dr. Eng
Examinador

Prof. Elizabeth Specialski
Moderadora

*A minha **Ana***
com muito amor...

Aos meus pais Ary e Wanda
Pelo amor, incentivo e carinho.

AGRADECIMENTOS

Em primeiro lugar, a Deus por tudo.

A todas as pessoas, que direta ou indiretamente, contribuíram para que este trabalho fosse concretizado.

Ao meu orientador Prof. Dr. João Bosco da Mota Alves pela amizade, orientação e pelos valores humanos transmitidos.

À minha amiga Profª Drª. Elizabeth Specialski pela força constante e seu notório saber a me

À minha família: Wânia, André, Ângelo, Dulce, Tatiana, Ana Paula, Rafael, Mariana, Ana Cecília, Renato, Leonório, Anadir, Roberto, Cacaio, Valda

Aos amigos Christiane, Simone, Joyce, Angelita, Alessandro, Silvana, Gisele, Márcio, Fernanda, Marcello, Iara, Lília, Alessandra, Eduardo, Débora, Rivalino, Freitas, Marta e Vicente não apenas pelo apoio, mas também pelos momentos

Aos professores integrantes da banca examinadora pelas críticas construtivas e elogios recebidos.

À UNIVALI e à UFSC por terem me proporcionado as condições para a realização deste trabalho.

SUMÁRIO

LISTA DE FIGURA

LISTA DE TABELA

LISTA DE ABREVIATURAS

APNOMS - Asia-Pacific Network Operations Managament Systems

ASN.1 - Abstract Syntax Notation One

BA – Boletim Atividade

CBRF - Case Base Reasoning Function

CMIP - Common Management Information Protocol

CMIS - Common Management Information Service

CRM - Customer Relationship Management

DAF - Directory Access Function

DCF - Data Communication Function

DCN - Data Communication Network

DN - Distinguish Name

DSF - Directory System Function

ERP - Enterprise Resource Planning

FAQs - Frequently Asked Questions

GDMO - Guidelines for the Definition of Managed Objects

ICF - Information Conversion Function

IDC - International Data Corporation

IM - Integrated Management

INRECA - Integrated Reasoning from Cases

ISO - Internatin Standardization Organization

ITU-T - International Telecommunications Union - Telecommunications Standardization Section

JNSM - Journal of Network and Systems Management

KM – Knowledge Management

LANONS - Latin-American Network Operations Managament Systems

LISHA - Laboratório de Integração de Hardware e Software UFSC

MAF - Management Application Function

MCB – Management Case Base

MCF - Message Communication Function

MCT - Management Case Tree

MD - Mediation Device

MF - Mediation Function

MIB - Management Information Base

MOPs - Memory Organization Packets

NE - Network Element

NEF - Network Element Function

NOMS - Network Operations Managament Systems

OAM&P - Operação, Administração, Manutenção e Provisionamento

ODBC - Open DataBase Connectivity

OS – Operation System

OSF - Operations Systems Function

OSI - Open System Interconnection

QA – Q Adaptor

QAF - Q Adaptor Function

R^4 – Recuperação, Reutilização, Revisão e Retenção

RBC - Raciocínio Baseado em Casos

RDN - Relative Distinguish name

RF - Reasoning Functional

RM-OSI - Open System Interconnection Reference Model

SBRC - Simpósio Brasileiro de Redes de Computadores

SF – Security Function

SFE – Situação Final do Equipamento

TL1 – Transaction Language 1

TMN - Telecommunications Management Network

UISF - User Interface Support Function

UML - Unified Modeling Language

WS – Workstation

WSF - Workstation Function

WSSF - Workstation Support Function

WTMN - Workshop Nacional de TMN

RESUMO

O ambiente de redes e serviços de telecomunicações é um sistema complexo. Como tal, o seu gerenciamento também é de alta complexidade. As experiências práticas, adquiridas ao longo dos anos e acumuladas individualmente por profissionais da área, são fundamentais para o sucesso e qualidade das atividades de gerenciamento de redes e serviços de telecomunicações. A Arquitetura de Informação TMN não faz referência, não orienta e não d
para representar conhecimento oriundo de experiências práticas em gerenciamento. Experiências estas que contém importantes informações *em como gerenciar*.

Este trabalho propõe um modelo para incorporar conhecimento advindo de experiências em gerenciamento de redes e serviços à Arquitetura TMN, através da abordagem de Raciocínio Baseado em Casos. O modelo proposto foi concebido em acordo com a filosofia e recomendações do ITU-T, orientado a objetos, aberto e padronizado. Introduz à Arquitetura TMN diferentes construções e *templates* genéricos que permitem representar conhecimento advindo de diferentes experiências práticas de gerenciamento, criar bases de conhecimento formais e, em conseqüência, viabilizar o desenvolvimento de sistemas abertos de conhecimento.

ABSTRACT

The telecommunications network and services environment is a very complex system. As such, its management is also of great complexity. The practical experiences, acquired along the years and individually accumulated by professionals of this area, are fundamental for the success and for the quality of the telecommunications network and services management. The TMN Information Architecture does not refer, does not guide and does not have elements to represent knowledge originated from management practical experiences. These experiences contain important information about *how to manage.*

This work proposes a model that will incorporate the knowledge derived from experiences in network and services management to TMN Architecture through the Case-Based Reasoning approach. The proposed model was thought in accordance to the ITU-T philosophy and recommendations, object oriented, open and standardized. It introduces different constructions and generic templates to the TMN Architecture that allow to represent the knowledge from different practical management experiences, to create formal knowledge bases and, consequently, it makes possible to develop knowledge open systems.

CAPÍTULO - 1 INTRODUÇÃO

1.1 APRESENTAÇÃO

O advento das novas tecnologias e serviços de telecomunicações tem implicações diretas na área de gerenciamento de redes, que se vê obrigada constantemente a introduzir novas tecnologias e padrões. Até pouco tempo atrás, as aplicações de gerência estavam voltadas exclusivamente para operar grandes e complexas redes, e focadas basicamente em aspectos de monitoramento e controle. Estes aspectos são essenciais para operação, administração e manutenção das redes de telecomunicações.

Face à desregulamentação, privatização e livre concorrência no m
em quase todo o mundo, o gerenciamento de redes tornou-se um recurso estratégico para o estabelecimento de vantagem competitiva neste segmento de negócio. Há a demanda por soluções de gerenciamento, que garantam a sobrevivência e que integrem diferentes necessidades, serviços, tecnologias e equipamentos.

Na busca por melhores desempenhos, as empresas estão sendo forçadas a reduzir custos, diminuir o tempo de respostas, atender aos clientes, aumentar a qualidade dos serviços, baixar os preços, etc. O sucesso neste mercado dependerá da habilidade e da capacidade de gerenciamento de cada empresa.

Usuários, fornecedores e entidades de padronização estão repensando o modo de fazer gerência. Uma tomada de decisão através do ambiente de gerenciamento reflete diretamente

sobre inúmeros negócios, de diferentes empresas e segmentos de mercado, que diariamente são realizados com suporte das redes de telecomunicações, envolvendo milhões de dólares.

Conseqüentemente é extremamente importante e necessário dispor de informações úteis e de natureza distinta para tomar decisões corretas, não interferindo nos negócios dos clientes, evitando prejuízos e reduzindo custos. Em geral, as plataformas de gerência de redes de telecomunicações e os modelos de gerência não dispõem de elementos suficientes para suporte às ações de monitoração e controle de redes e serviços. A responsabilidade da decisão recai integralmente sobre a equipe técnica de gerenciamento.

Normalmente, estas decisões são tomadas em um curto espaço de tempo, sob grande pressão e tensão, a partir de dados incompletos e imprecisos, baseadas, quase sempre, no conhecimento advindo da experiência prática individual da equipe técnica de gerência.

Este conhecimento, baseado em experiências práticas, é importante não só para atividades de operação, administração, manutenção e provisionamento de redes e serviços de *Operation, Administration, Maintenance & Provisioning*), mas também para todas as atividades, relacionadas ao gerenciamento, de uma maneira geral: planejamento, modelagem, implementação, desenvolvimento. Sendo assim, torna-se relevante tratá-lo no ambiente de gerenciamento de redes e serviços de telecomunicações. O contexto deste trabalho está inserido no uso do conhecimento baseado em experiências para o gerenciamento de redes e serviços de telecomunicações.

1.2 JUSTIFICATIVAS

O ambiente de telecomunicações é complexo, englobando diversas tecnologias, equipamentos e sistemas de diferentes fabricantes e fornecedores. Consequentemente, as soluções de gerência para este ambiente também são complexas, fazendo uso de diferentes sistemas, que quase sempre são proprietários e que não interoperam de forma transparente. A experiência dos recursos humanos da empresa em atividades de OAM&P de redes e serviços de telecomunicações é um dos fatores críticos de sucesso.

As plataformas de gerenciamento oferecem um conjunto de recursos básicos para o desenvolvimento de aplicações de gerência. As soluções de gerenciamento prontas, do tipo *plug and play*, oferecidas por estas plataformas são poucas e restritas. Para atender as reais necessidades das empresas, as soluções devem ser criadas e customizadas. Sendo assim, as ferramentas de gerência têm que oferecer um conjunto mínimo de recursos para desenvolvimento, acompanhamento e depuração de software. Contudo, o ambiente de desenvolvimento das plataformas de gerenciamento é limitado.

A Arquitetura TMN (*Telecommunication Network Management*) visa resolver os problemas de integração dos sistemas de gerência de redes de telecomunicações. Trata-se de um modelo a ser seguido e não uma solução. E como tal, identifica as informações básicas e os requisitos funcionais do gerenciamento, as propriedades genéricas dos recursos lógicos e físicos que são passíveis de gerência, etc. Todavia, respeitando a independência de cada usuário, a Arquitetura TMN não define e não descreve como gerenciar, como resolver um problema de gerenciamento ou como implementar um sistema de gerenciamento. Este tipo de

conhecimento, que é básico para o sucesso do gerenciamento, não é referenciado pelos modelos de gerenciamento e nem pela Arquitetura TMN.

A complexidade do ambiente obriga às soluções de gerência a empregar conhecimentos advindos do uso de diferentes tecnologias, paradigmas, e campos da ciência, tais como: engenharia de software, banco de dados, sistemas distribuídos, inteligência artificial, dentre outros (Udupa, 1999; Aidarous & Plevyak, 1998; Mattison, 1997 e Adams & Willetts, 1996).

Normalmente, o conhecimento requerido excede a capacidade de uma equipe comum de gerência, aos recursos disponíveis nas plataformas de gerenciamento e às informações disponíveis nas bases de informações de gerência, nos livros e em manuais técnicos. Atualmente, boa parte das ações de gerenciamento é baseada na experiência dos profissionais envolvidos nesta área, tanto para o gerenciamento das redes e serviços de telecomunicações quanto para o desenvolvimento de novas aplicações de gerenciamento. São experiências adquiridas ao longo dos anos e acumuladas individualmente por profissionais da área.

A Arquitetura de Informação TMN fornece um guia, para representar e modelar as informações de gerenciamento, e define estruturas de dados para representar como estas mo são identificadas e manipuladas. Não faz referências e nem orienta como representar o conhecimento em gerenciamento TMN.

Davenport & Prusak (1998) diferenciam dados, informações e conhecimento. Dados são um conjunto de fatos sobre eventos. As informações, ao contrário dos dados, têm relevância e objetivo, têm significado dentro de um contexto. Já o conhecimento é mais amplo, profundo e rico. É uma combinação de experiências, valores, informações contextuais e discernimento, avaliação e incorporação de novas experiências e informações.

Isto se origina e é aplicado na mente dos especialistas. Não está apenas em arquivos e documentos em uma organização, mas também nas rotinas operacionais, processos, práticas e normas.

O conhecimento que é essencial para a construção de soluções de gerenciamento, não é capturado, representado, formalizado e disponibilizado ao ambiente de trabalho das instituições. Isto inviabiliza mensurá-lo, reutilizá-lo, transmiti-lo e introduzi-lo de forma estruturada e padronizada em um ambiente de administração e desenvolvimento de aplicações

Para tanto, faz-se necessário estender e adaptar a Arquitetura TMN, definindo elementos que lhe permitam agregar, além de informações, conhecimento baseado
gerenciamento, a fim de que viabilizem o desenvolvimento de sistemas abertos de conhecimento para aumentar a qualidade das atividades de gerenciamento, estabelecendo o diferencial de competitividade das empresas.

É importante a definição de um modelo aberto, genérico e padronizado que viabilize diferentes sistemas de conhecimento compartilharem experiências em como fazer gerência, possibilitando que este conhecimento seja socializado, reduzindo a complexidade do ambiente e facilitando o aprendizado.

Este tipo de abordagem permite que fabricantes de equipamentos, fornecedores de soluções, especialistas e usuários de gerenciamento de redes e serviços de telecomunicações, criem diferentes bases de conhecimento, com suas experiências práticas
troquem livremente, contribuindo significativamente para um avanço e uma melhora na qualidade das atividades de gerenciamento.

Nestes termos, a entidade de padronização internacional ITU (*International Telecommunications Union*) tem um papel de grande relevância. Através do seu setor de telecomunicações ITU-T (*International Telecommunications Union - Telecommunications*), antigo CCITT (*International Telephone and Telegraph Consultative Committee*), publica recomendações com o objetivo de padronizar as redes e serviços de telecomunicações em

Estas recomendações são divididas por áreas e aplicáveis a mais de uma tecnologia e serviço, permitindo a interoperação de sistemas abertos, ou seja, que diferentes serviços e equipamentos de fabricantes e fornecedores diversos sejam compatíveis. A Arquitetura TMN

-T. Portanto, a definição de modelos e elementos novos à Arquitetura TMN deve, também, estar em acordo com as recomendações ITU-T para garantir a compatibilidade e investimentos futuros.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

Conceber um modelo conceitual para introduzir elementos à Arquitetura TMN, em acordo

-T, para incorporar conhecimento baseado em experiências, em gerenciamento de redes e serviços de telecomunicações, através da abordagem de Raciocínio Baseado em Casos (RBC).

1.3.2 OBJETIVOS ESPECÍFICOS

- Adaptar a Arquitetura TMN para incorporar experiências práticas em um ambiente de gerenciamento TMN, a partir da abordagem RBC;

- Desenvolver um modelo conceitual genérico que suporte a introdução de diferentes experiências práticas de forma padronizada em um ambiente de gerenciamento TMN;
- Formalizar a representação de experiências práticas em gerenciamento de redes e serviços
- Demonstrar que bases de conhecimento podem ser agregadas ao ambiente de gerência de redes e serviços de telecomunicações através da abordagem RBC;
- Demonstrar que bases de conhecimento em conformidade com as recomendações TMN podem ser vendidas e compartilhadas, independente de fornecedores de soluções.

1.4 LIMITAÇÕES DO ESTUDO

Este trabalho evidencia o desenvolvimento de um modelo conceitual para formalizar e disponibilizar experiências práticas úteis aos ambientes de gerenciamento TMN. A metodologia adotada e os conceitos utilizados seguem as recomendações da entidade internacional de padronização ITU-T.

Por se tratar de um modelo genérico, a proposta está centrada na definição de elementos básicos para que experiências práticas de gerenciamento possam ser representadas formalmente, integradas de forma padronizada à Arquitetura TMN, e utilizadas por sistemas abertos de conhecimento em um ambiente de gerenciamento.

O contexto deste trabalho não focaliza o processo de aquisição do conhecimento em um ambiente de gerenciamento de redes e serviços de telecomunicações. A metodologia de aquisição de conhecimento sobre experiências práticas, procedimentos a serem aplicados, técnicas disponíveis e a forma de tradução em conhecimento útil requer um estudo sob o ponto de vista cognitivo e não faz parte do escopo deste trabalho.

Nenhum recurso novo é introduzido por este modelo que impeça a sua implementação. Todos os elementos propostos estão em conformidade com as recomendações ITU-T, amplamente difundidas e discutidas.

1.5 MOTIVAÇÃO

O ambiente de redes e serviços de telecomunicações é bastante complexo. Várias iniciativas têm sido realizadas para aumentar a qualidade das ações neste ambiente. Diversos grupos de
r diferentes organismos nacionais e internacionais.

A não trivialidade deste trabalho surge a partir da inclusão da abordagem RBC neste ambiente. A abordagem RBC já é utilizada em atividades de gerenciamento, porém a partir de
orte e mais especificamente em serviços de correlação de alarmes. A contribuição desta proposta está na mudança deste cenário, onde, com o modelo proposto, criam-se mecanismos na Arquitetura TMN para agregar bases de conhecimento padronizadas e permitir o desenvolvimento de sistemas abertos de conhecimento para o suporte às atividades de gerenciamento de redes e serviços de telecomunicações.

e, principalmente, a natureza do conhecimento embutido.

A compreensão destes dois domínios permite visualizar as funcionalidades básicas requeridas para esta integração, de modo que se mantenha a compatibilidade e a padronização, permitindo o desenvolvimento de aplicações abertas de suporte ao processo de tomada de decisão às atividades de gerenciamento de redes e serviços de telecomunicações.

Neste contexto, este trabalho está organizado em quatro capítulos, excluindo o capítulo corrente, com o seguinte conteúdo temático:

- Capítulo 2: Raciocínio Baseado em Casos:

  Apresenta uma visão geral sobre a abordagem de Raciocínio Baseado em Casos, com um enfoque inicial sob o ponto de vista cognitivo, e em seqüência descreve suas

- Capítulo 3: Arquitetura TMN

  Contém uma visão geral sobre o modelo de gerência TMN, onde são descritas as Arquiteturas Funcional, Física e de Informação.

- Capítulo 4: Modelo Conceitual Proposto

  Descreve o modelo proposto, sua filosofia, seus conceitos e elementos básicos.

- Capítulo 5: Aplicação do Modelo Proposto

  Apresenta uma aplicação do modelo proposto na criação de uma base de conhecimento específica e, em seqüência, apresenta dois cenários possíveis da utilização desta mesma base de conhecimento na criação de apli

- Capítulo 6: Considerações Finais

  Discute os resultados alcançados, destaca as contribuições do trabalho sob o ponto de vista informativo, inovador e de fomento a pesquisa, e por fim propõe um conjunto de potenciais trabalhos futuros.

- Anexo A:

  Apresenta uma especificação, através da linguagem UML (*Unified Modeling Language*), visando facilitar a compreensão da aplicação do modelo proposto por esta tese.

- Anexo B:

  Em função de serem muitos os atributos criados pelo modelo, estes são especificados

CAPÍTULO - 2 RACIOCÍNIO BASEADO EM CASOS

2.1 INTRODUÇÃO

A abordagem de Raciocínio Baseado em Casos (RBC) é aplicada para resolução de problemas e aprendizagem. Faz uso de conhecimento extraído de experiências similares, anteriormente vivenciadas, para resolver um problema atual.

É uma metodologia genérica para a construção de sistemas baseados em conhecimento específico de experiências passadas. Segundo Leake (1996), problemas similares têm soluções similares e os tipos de problemas se repetem. Algumas fundamentações e referências gerais podem ser encontradas em Riesbeck & Schank (1989), Kolodner (1993), Aamodt & Plaza (1994), Leake (1996), Watson (1997) e Lenz, Bartsch-Spörl, Burkhard & Wess (1998).

A abordagem RBC tem sua origem na ciência cognitiva e na inteligência artificial. Em função desta interdisciplinaridade, tem sido muito difundida e aplicada em diferentes domínios. Diferentemente das técnicas tradicionais de Inteligência Artificial, seu conhecimento está

O conhecimento de RBC é adquirido através de situações concretas de desenvolvimento de idéias, criação de produtos, desenvolvimento de projetos, implantação de políticas, reparo de equipamentos, tomada de decisões, aplicação de soluções, atividades de planejamento, diagnósticos, previsão, análise, modelagem, etc. Conhecimentos advindos de experiências práticas servem como guia para responder a novas situações (Greese, Ramos, Althoff, Barcia, Weber, & Martins, 1998).

Um administrador de rede, quando se depara com um conjunto de alarmes em sua console, desconhecendo a causa e o efeito destes, busca em sua memória situações anteriores vivenciadas, parecidas com a situação atual, e adapta as respectivas medidas, empregadas anteriormente, para tratar os atuais alarmes.

Para Squire (mencionado em Matlin, 1998), um problema existe quando se tem um objetivo, mas não é óbvio como alcançá-lo. Para Richard (1990), uma situação é identificada como problema, quando não se dispõe dos conhecimentos necessários para decidir qual ação tomar ou solução adotar. Neste caso, faz-se necessário criar um espaço para pesquisar e achar uma solução no interior deste espaço.

Segundo Matlin (1998), o processo de resolução de problemas é muito influenciado pelo uso de conhecimento prévio. Entretanto, este não é obrigatoriamente aplicado de forma rotineira e reprodutiva. Existe a capacidade de alterar e moldar seletivamente experiências anteriores, de forma totalmente conceitual, para torná-las aplicáveis em situações novas e inesperadas (Eysenck & Keane, 1990).

Neste contexto, a abordagem RBC é empregada. Permite a criação de uma base de conhecimento com experiências, vivenciadas na prática, que são modeladas através de casos. Esta base de casos é o espaço cri

Diante da situação problema, um sistema RBC recupera, em sua base de conhecimento, um caso com características similares às da situação problema. De acordo com as particularidades da situação problema, tenta aplicar a mesma solução fornecida pelo caso recuperado, adaptando o que for necessário ao contexto atual.

A resolução da situação problema transforma-se em uma experiência vivenciada. Independentemente do sucesso ou do fracasso da solução adotada, tem-se mais um caso para ser adicionado à base de conhecimento, o que permite um aprendizado constante.

A pesquisa sobre a resolução de problemas é um dos principais objetos de estudo para a abordagem da ciência cognitiva. Diversos estudos destacam a importância do conhecimento advindo de experiências prévias no processo de execução de tarefas e aprendizagem (Eysenck & Keane, 1985; Getner, 1983; Ross, 1989; Schank, 1982; Carbonell, 1986; Anderson, 1983; Matlin, 1998 e Richard, 1990).

Dentre estes, pode-se destacar o trabalho Schank (1982) e seu modelo de memória dinâmica que para muitos é tido como a procedência da abordagem RBC. Neste modelo, como premissa, os processos de recordação, compreensão, experimentação e aprendizagem não podem ser tratados isoladamente. A memória, que é -se conforme as experiências vivenciadas pelo indivíduo.

A compreensão dá-se pela tentativa de integrar as novas experiências com o que já é conhecido, acarretando, constantemente, a reorganização e o refinamento da memória. Nunca a se comporta exatamente do mesmo modo, devido ao dinamismo que é proporcionado pelas experiências vivenciadas. As experiências provêm expectativas que tornam o processo da compreensão mais fácil e simples.

O Modelo de Memória Dinâmica de Roger Schank, introduz o conceito de Pacotes de Organização de Memória - MOPs (*Memory Organization Packets*), criando uma estrutura hierárquica onde experiências similares são agrupadas em um MOP comum.

ria, generalizações e recuperação na memória. Tem como base de conhecimento a agenda do secretário norte-americano Cyrus Vance com suas experiências vivenciadas em viagens e reuniões.

A partir do sistema CYRUS, inúmeros outros sistemas, comerciais e não comerciais foram desenvolvidos, permitindo um grande avanço para a abordagem RBC, bem como a sua disseminação para os mais variados domínios, tais como: justiça, rede de computadores, finanças, medicina, educação, seguros, manufatura, transporte, etc., e para diversos tipos de aplicações: suporte a clientes, configuração de equipamentos, tratamento de falhas de equipamentos, jurisprudências, suporte a decisão, planejamento, diagnóstico, previsão, criação, treinamento, aprendizado, classificação, dentre outras (Watson, 1997).

Segundo Webber (1998), a abordagem RBC apresenta as seguintes vantagens:

- Extração do conhecimento: a extração do conhecimento e sua representação são realizadas através dos fatos que descrevem uma experiência;
- Representação do conhecimento: a representação do conhecimento resume-se em escolher o tipo de estrutura da base de casos. Em algumas aplicações é necessária a criação de

- Reutilização do conhecimento: o conhecimento contido nos casos pode ser utilizado, combinado e adaptado para gerar novas soluções, além das originalmente presentes na

- Aprendizagem: a atualização do conhecimento é feita automaticamente na medida que as experiências são utilizadas, assim o sistema pode crescer e, incrementar sua robustez e

- Justificativas: justificativas são sempre consistentes quando as soluções são as próprias experiências, representando aspectos de proximidade ao comportamento humano do sistema. Além disso, as justificativas podem avisar sobre possíveis riscos que o uso de determinada abordagem pode implicar;

- Consciência: se o sistema não encontra casos com a devida similaridade com o caso de entrada, não é gerada solução. Isto evita possíveis problemas gerados por outros sistemas

- Fácil acesso às soluções: o RBC reduz o espaço de busca para a solução. O problema pode ser identificado pelo sistema, o suficiente para recuperar soluções, não sendo necessário que o mesmo entenda perfeitamente as condições e circunstâncias para propor uma

  O RBC também proporciona um meio de resolução de problemas, quando não houver um algoritmo disponível para avaliação e solução do mesmo;

- Raciocínio implícito: a incerteza implícita nas informações contidas nos casos é utilizada sem a necessidade de um tratamento específico.

Watson (1997) apresenta e analisa um conjunto de diferentes ferramentas de software que viabilizam o desenvolvimento de aplicações RBC: ART*Enterprise; Case1; Case Advisor; Case Power; CBR3; Eclipse; The Easy Reasoner; Esteem; KATE; ReCall; ReMind; CASUEL; CASPIAN; Recon; e CBRWorks.

Estas ferramentas oferecem um conjunto de recursos para se criar bases de conhecimento, representar e recuperar casos, definir métricas de similaridade, reutilizar e adaptar casos,

ção e exportação de bases de dados, dentre outras facilidades. A

criação de bases de conhecimento garante os investimentos anteriores realizados pelas empresas e permite acelerar o processo de formação de novos especialistas, assegurando o aprendizado continuado dentro da organização.

A abordagem de RBC tem sido usada para a construção de bases de conhecimento e foi destaque na 3ª Conferência Internacional de Raciocínio Baseado em Casos - ICCBR-99 (*Third Conference on Case Based Reasoning*), realizada em julho de 1999, na Alemanha (Aha & Avila, 1999), onde teve uma seção especial focada na aplicação de RBC na construção e gerenciamento de bases de conhecimento em diferentes áreas de aplicação.

2.2 ARQUITETURA RBC

No modelo proposto por Aamodt & Plaza (1994), universalmente, conhecido como modelo R^4 (Figura 2.1), a abordagem RBC é constituída por quatro etapas:

- Recuperação do caso mais similar da base de casos;
- Reutilização das informações disponíveis no caso para solucionar a situação problema;
- Revisão: reavalia a solução proposta;
- Retenção o conhecimento extraído da experiência vivenciada para uso futuro.

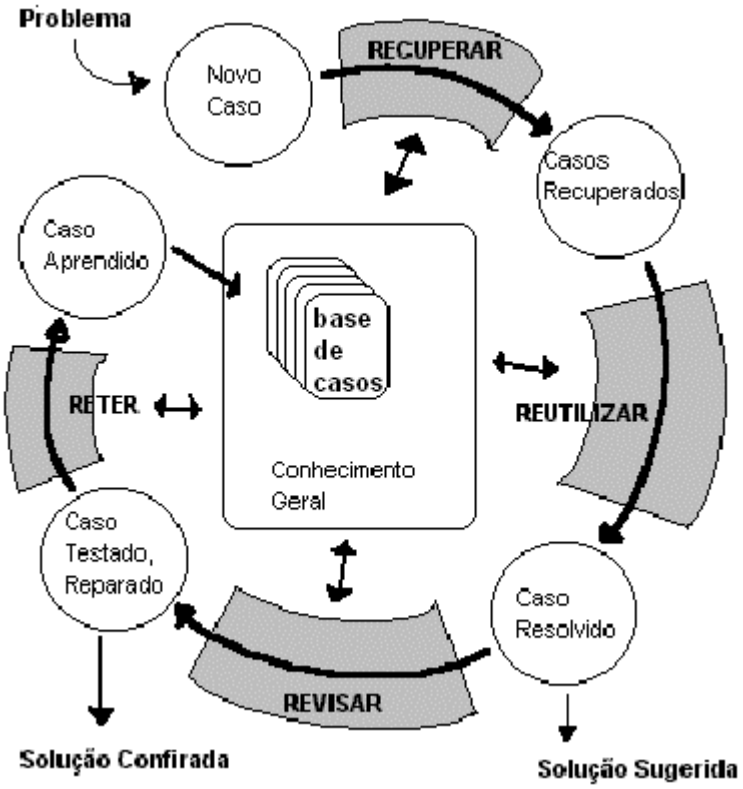


Figura 2.1 - Ciclo RBC (Aamodt & Plaza, 1994)

Tais etapas constituem um instrumento de orientação para a aplicação prática de RBC, -requisitos necessários. É um ciclo contínuo de raciocínio. Além destas etapas, uma aplicação RBC é composta também por uma base de dados, chamada conceitualmente de Base de Casos, que é mantida em memória e serve para armazenar e organizar os casos.

Existem diferenças quanto ao número e nome das etapas entre alguns modelos sugeridos e aplicados para o desenvolvimento de sistemas RBC. Entretanto, todos estes enfoques metodológicos estão baseados na filosofia RBC. Em particular, cada etapa é constituída de um conjunto de tarefas que podem ser implementadas através de diferentes técnicas.

As etapas de um sistema RBC não podem ser analisadas independentemente. Devem ser trabalhadas como um todo, pois são totalmente dependentes e inter-relacionadas. Tanto a

modelagem quanto a linguagem empregadas na representação de casos estão diretamente ligados às particularidades do domínio da aplicação.

Inicialmente, a metodologia de desenvolvimento de uma aplicação RBC requer a definição da forma de representação de casos. Este é um importante elemento que diretamente influencia e determina a implementação do modelo R^4.

2.2.1 REPRESENTAÇÃO DE CASOS

Um caso é a formalização de um conhecimento, advindo de uma experiência prática, vivenciada anteriormente e associada a um contexto, que é construída pela ação a partir da resolução de um problema. Independentemente dos resultados alcançados serem bons ou ruins, de ter sido um sucesso ou não, o importante é que a experiência tenha algo útil a transmitir. Os casos são organizados em uma base de dados, chamada conceitualmente de Base de Casos, que é mantida em memória.

À medida que um sistema RBC tem como essência os casos, a sua representação é fundamental. É uma das primeiras etapas a serem cuidadosamente estudadas e definidas. A representação de casos envolve a descrição dos elementos e das propriedades relevantes da experiência modelada, a organização e o método de acesso destes em memória.

Existem três abordagens principais para a representação de casos que diferem segundo a origem, a forma e o conhecimento empregado (Bergmann, Breen, Göker, Manago & Wess, 1999):

- Abordagem baseada em textos;

- Abordagem baseada em conversação;
- Abordagem estruturada.

Na abordagem baseada em textos, os casos são representados em forma de texto livre. É muito usada em ambientes onde já existe uma grande quantidade de documentos, normalmente chamados de *how to do* - como fazer e/ou listas de discussão FAQs (*Frequently Asked Questions*). Estes documentos permitem o usuário rapidamente utilizar o conhecimento descrito.

Esta abordagem é indicada para domínios que não dispõem de muitos casos (menos do que uma centena). Em contrapartida, recuperam um grande número de casos que são irrelevantes. O custo de controle de qualidade é alto.

A abordagem baseada em conversação é voltada para domínios onde uma grande quantidade de problemas simples deve ser resolvida constantemente (ex. suporte a usuários do *windows*, a usuários da internet, etc.). Este sistema guia o cliente e o técnico de suporte através de -definidos.

A base de casos é organizada normalmente pelo autor do caso, o que torna esta atividade complexa e de alto custo. É voltada para aplicações onde poucas questões são necessárias para a tomada de decisão. O custo de manutenção é alto.

A abordagem estrutural descreve casos através de atributos e valores pré-definidos. Os atributos podem ser estruturados em tabelas relacionais ou segundo o paradigma da orientação a objetos.

O modelo do domínio define um padrão para representar casos, especificando o conjunto de atributos usados. A definição de novos casos é de alta qualidade e o custo da manutenção é baixo, porém o investimento inicial para produzir o modelo do domínio é alto. Apresenta melhores resultados do que as outras duas abordagens. Em geral, um caso pode ser representado a partir de alguma estrutura de dados (par: atributo, valor) para descrever um problema e uma solução.

2.2.1.1 Descrição do Problema

A descrição do problema identifica os elementos de entrada de um sistema RBC. Ou seja, descreve as características da situação problema, as quais um sistema RBC faz uso para recuperar casos similares da base casos.

Segundo Webber (1998), a descrição deve ter elementos suficientes para poder identificar o caso e sua similaridade com outros casos. São usados nomes, números, funções, ou textos e servem para representar características, objetivos, metas, restrições, condições, etc.

Para Kolodner (1983), a descrição do problema possui três componentes básicos:

- Os objetivos a serem alcançados na resolução do problema;
- As restrições a estes objetivos;
- As características da situação problema.

Os objetivos podem ser concretos ou abstratos: configurar rotas; executar um programa; monitorar um circuito; coletar alarmes; modelar um recurso gerenciado; classificar um evento; diagnosticar uma falha; correlacionar alarmes; criar um modelo de informação; projetar uma rede, etc.

As restrições são as condições postas para alcançar estes objetivos:

- Ao configurar rotas, determinadas máquinas ou aplicações devem usar circuitos

- Determinado programa ao ser executado requer uma memória de 32Mb disponível;
- A coleta de alarmes deve indicar o grau de severidade dos mesmos;
- A modelagem dos recursos gerenciados é sob o ponto de vista da área funcional de

- Projeto da rede não pode gastar mais do que um determinado valor;

As características da situação problema são um conjunto de informações que são relevantes para descrever o contexto no qual o problema está inserido:

- Identificar a marca do roteador ou a versão do sistema que está rodando;
- Identificar a versão do programa ou o ambiente para qual foi desenvolvido (Unix, NT, etc.);
- Identificar a velocidade do circuito;
- Identificar a freqüência da coleta dos alarmes;
- Identificar o fabricante do recurso gerenciado, o modelo.

2.2.1.2 Descrição da Solução

A descrição da solução identifica os elementos da ação empregada para a resolução do problema, descrevendo desde a própria solução como também o raciocínio empregado, o conjunto de soluções possíveis de serem empregadas, a solução escolhida, as justificativas, as adaptações feitas.

Em certas aplicações, onde o domínio do conhecimento é complexo ou imaturo, faz-se necessário modelar os resultados alcançados com a solução proposta. Muitas vezes, uma solução pode ser uma ação cujos efeitos não são conhecidos a prior

casos são representados a partir da descrição do problema, da solução e também dos resultados.

Os resultados descrevem as conseqüências da ação, a aplicabilidade, ou seja, se a ação foi um sucesso ou não, a estratégia utilizada para reparar algum insucesso da ação, o que poderia ter sido feito para evitar este insucesso, etc.

2.2.1.3 Organização da Memória

A organização da memória define como os casos são armazenados e, consequentemente, também, recuperados na base de casos. A estrutura usada pode variar em função do tipo da abordagem usada para representar casos. As abordagens são voltadas para propósitos completamente distintos; consequentemente, estruturas diferentes podem adequar-se melhor a um tipo do que outro.

Em geral, a estrutura adotada é definida em função da quantidade de casos existentes. A representação de casos define o tipo de organização que será suportada pela aplicação RBC. As estruturas mais comuns são: a organização seqüencial ou hierárquica dos casos. Na seqüencial, não existe uma estrutura lógica para os casos. Estes são armazenados em seqüência, um após o outro. É uma estrutura simples e fácil de implementar, porém não é boa para grandes quantidades de casos.

Na estrutura hierárquica os casos são categorizados e organizados em um modelo de árvore, requer a utilização de índices que são definidos de acordo com as particularidades do domínio. Entretanto, há também modelos que combinam os dois tipos, organizando os casos em categorias hierárquicas e dentro de cada categoria emprega a organização seqüencial.

2.2.1.4 Métodos de Acesso

Em função de como a base de casos está organizada, a aplicação RBC pode recuperar casos de diferentes maneiras. Em uma base de casos organizada em seqüência, o acesso é restrito à ordem em que os casos são gravados, sendo que a inclusão de novos casos só é possível no fim da base de casos.

Um método de acesso mais direto é chamado de acesso indexado ou acesso por chave. A base de casos deve possuir uma área de índices, onde existam ponteiros para os diversos casos. Sempre que uma aplicação RBC desejar acessar um caso, deverá especificar uma chave através da qual o sistema pesquisará, na área de índices, o ponteiro correspondente. A partir

-se um acesso direto ao caso desejado.

O acesso indexado pode ser combinado com o acesso seqüencial, onde a partir de uma chave, uma categoria ou classe de casos é alcançada e, dentro desta classe, o acesso ocorre de forma seqüencial a cada caso em particular.

Em uma base com um número muito grande de casos, o acesso seqüencial torna-se inviável. O uso de índices permite acelerar o processo de recuperação de casos. Segundo Webber (1998), a indexação de casos é um instrumento para orientar a similaridade. É necessário identificar quais características de um caso são relevantes para determinar a sua similaridade

com outros casos. Estas características relevantes são definidas como chaves e usadas como índices para organizar a base de casos. Este aspecto faz da indexação uma etapa fundamental para a representação de casos baseada na abordagem estruturada.

2.2.1.5 Linguagem para Representação de Casos

A definição de uma linguagem permite a uniformização e a normatização da implementação de uma base de casos, através da introdução de mecanismos
propriedades e dos elementos de um caso. A linguagem é formada por um conjunto de palavras chave e por uma notação comum para a modelagem dos vários aspectos de um caso.

Se a linguagem usada for muito abrangente, pode tornar difícil a interpretação da natureza do problema e do raciocínio usado para construir a solução proposta. Por outro lado, se a linguagem for muito restrita e não dispor de recursos, pode-se não conseguir representar a importância dos detalhes das informações para a aplicação RBC.

Segundo Lewis (1995), definir uma linguagem ideal não é uma tarefa simples, No começo, deve-se encontrar um meio termo, a linguagem deve ser nem muito abrangente e nem muito restritiva. À medida que novos casos forem sendo definidos, a linguagem preliminar vai se refinando e convergindo para uma linguagem ideal.

Como exemplo, pode-se destacar a linguagem CASUEL (Manago, Bergmann, Conruyt, Traphöner, Pasley, Le Renard, Maurer, Wess Althoff & Garry, 1994) oferecida pelo projeto INRECA (*Integrated Reasoning from Cases*). É uma linguagem orientada a objeto para descrição de casos. É uma linguagem aberta que permite que sejam agregadas novas funcionalidades e aplicadas a diferentes tipos de aplicações.

2.2.2 RECUPERAÇÃO DE CASOS

A recuperação é uma função comum em bases de dados. As informações são recuperadas a partir de uma busca por atributos ou chaves iguais. Todavia, em uma base de casos, a busca dá-se por atributos como valores parecidos e/ou semelhantes. Uma aplicação RBC deve possuir recursos para identificar casos similares.

Aamodt & Plaza (1994) apresentam dois tipos de similaridade: sintática e semântica. A similaridade sintática, mais superficial, é estabelecida em termos da semelhança sintática dos atributos, como sinônimos, análise de perfil, categorias, qualificadores, dentre outros. A similaridade semântica, mais complexa, propõe-se a englobar o significado dos casos.

A definição das características similares entre casos ou classes de casos é fruto do conhecimento de um especialista. Há necessidade de um conhecimento profundo sobre o domínio da aplicação

Existem diferentes algoritmos voltados para a recuperação de casos, como por exemplo: *Nearest-neighboor; Kd-tree; Fish-and-Sink, Crash memory model; Knowledge-directed Spreading Activation* (KDSA); *Case Retrieval Nets* (CRNs); *Objectdirected Case Retrieval Nets* (OCRNs); e várias extensões destes (Lenz, Burkhard & Brückner, 1996). Em ferramentas comerciais, duas técnicas são mais usadas: *nearest-neighboor* e *inductive retrieval* (Watson, 1997).

A técnica *nearest-neighboor* retrieval faz uso de uma fórmula para calcular a distância Euclideana entre dois casos. Distância Euclideana é uma medida numérica de similaridade, onde quanto menor for esta medida maior é o grau de similaridade entre os casos.

$$(P,Q) = \left(\sum_{i=1}^{n} (x^p_i - x^q_i)^2 \right)^{1/2}$$

P e Q são dois casos descritos pelos vetores de atributos x^p e x^q respectivamente.

A técnica *inductive retrieval* constrói, a partir de um atributo chave, um árvore de decisão indexada que é usada para organizar e recuperar os casos.

2.2.3 REUTILIZAÇÃO DE CASOS

A etapa de reutilização é responsável por verificar se a solução apresentada pelo caso recuperado pode ser diretamente aplicada ao novo caso ou não. Existem situações onde soluções anteriormente adotadas necessitam ser adaptadas às necessidades do novo caso.

Aamodt & Plaza (1994) apresentam dois tipos de adaptação:

- Adaptação derivativa: reutiliza o método usado para construir a solução;
- Adaptação transformativa: reutiliza a solução do caso passado.

As técnicas de adaptação variam desde as mais simples até as mais complexas. Pode-se destacar algumas, tais como:

- Adaptação nula: não faz uso de adaptação.
- Adaptação por substituição: um ou mais componentes de um caso são substituídos por novos elementos.
- Ajuste de parâmetros: compara parâmetros específicos do caso recuperado com o novo caso e modifica a solução no contexto apropriado.

- Reinstanciação: instancia características de uma antiga solução com novos valores aos atributos que descrevem o caso.

A adaptação é um recurso muito importante na abordagem RBC. Agrega valor ao sistema, principalmente se ocorre de forma automática, sem intervenção humana. Através de um conjunto de regras, previamente definidas por um especialista, o sistema RBC é capaz de identificar as diferenças, entre o caso recuperado e o novo caso, e inferir as adaptações necessárias para reutilizar a solução anterior.

Ë importante destacar que a adaptação automática apesar de ser um recurso que agrega flexibilidade e conhecimento a um sistema RBC, não é um recurso essencial. A adaptação pode ser realizada pelo próprio usuário do sistema ou, simplesmente, não ser nem necessária. A adaptação automática é um recurso complexo e requer o conhecimento de um especialista para definir o contexto no qual deve ser aplicada.

A adaptação deve suportar as diferenças entre todos os futuros casos e os casos conhecidos, disponíveis na base, e determinar as adaptações requeridas por estas diferenças. Isto requer um profundo conhecimento do domínio da aplicação, o que muitas vezes inviabiliza o seu uso. Parte do sucesso da abordagem RBC tem sido atribuído justamente a redução da dependência de um engenheiro do conhecimento. Watson (1997) sugere que a menos que seja usada em um domínio bem conhecido, a adaptação deve ser evitada.

2.2.4 REVISÃO DE CASOS

Uma vez que o caso foi recuperado e reutilizado, a etapa de revisão visa avaliar a solução adotada para o novo caso. Caso a solução não tenha alcançado o resultado esperado, identificar quais os problemas encontrados e corrigi-los. Independente de que a solução

adotada tenha sido a mais apropriada ou não, este tipo de indicativo é fundamental para a qualidade de um sistema RBC.

Esta avaliação permite agregar mais conhecimento ao sistema, oriundo de um caso real, de uma experiência prática vivenciada com a utilização da solução proposta. Kolodner (1983) ressalta a importância deste tipo de informação e inclui os resultados de um caso como componentes específicos na representação de casos. A partir destes componentes é possível agregar conhecimento e evitar futuros erros na reutilização de casos.

2.2.5 RETENÇÃO DE CASO

A etapa de retenção consiste em selecionar quais informações do novo caso que devem ser retidas e incorporadas a base de conhecimento. Novos casos podem ser agregados a base de casos e a função de similaridade pode ser refinada. Um sistema pode ser modificado em função de sua utilização. Segundo Kamp, Lange e Globig (1998), o processo de alteração de um sistema, em resposta ao seu ambiente, faz parte de um processo de aprendizagem que é uma das vantagens da abordagem RBC. O aprendizado é fruto do processo de revisão da solução proposta e da possibilidade de reparar as ações que não alcançaram os resultados esperados.

2.3 METODOLOGIA PARA O DESENVOLVIMENTO DE APLICAÇÕES RBC

Soluções RBC não são soluções de prateleira, prontas e disponíveis no mercado. São construídas. Existem poucas instituições que são especializadas no desenvolvimento de aplicações RBC. Não existem orientações ou métodos que possam dar suporte aos desenvolvedores de novas projetos e nenhum recurso que preserve e disponibilize

experiências de outros projetos. Isto pode causar sérios problemas quando a mudança na equipe de desenvolvimento de software ou quando novos membros precisam ser treinados e incorporados a equipe. O resultado é um desenvolvimento de software ineficiente.

A fim de incentivar e disseminar a abordagem RBC, o consórcio europeu INRECA, criado em 1992, desenvolveu uma metodologia para dar suporte às atividades de desenvolvimento e manutenção de aplicações RBC (Bergmann & Althoff, 1998
Manago & Wess, 1999).

Dentre outros benefícios, o uso apropriado da metodologia provê:

- Aumento de produtividade.
- Aumento de qualidade.
- Referencial para comunicação entre equipes de desenvolvimento.
- Gerenciamento de atividades através de uma base sólida para planejamento, alocação e

A metodologia define um conjunto de orientações para as atividades que necessitam ser executadas a fim de obter sucesso no desenvolvimento de aplicações RBC. São apresentadas nove etapas que devem ser consideradas durante o planejamento, a implementação e execução de um sistema RBC:

- Definir claramente os objetivos;
- Aplicação deve ser orientada às necessidades do cliente;
- Definir, no início do processo, o contexto no qual a aplicação irá ser desenvolvida;
- Considerar seriamente o processo de manutenção do sistema;

- Testar continuamente o sistema;
- Definir métricas para controle de qualidade;
- Reduzir o custo da manutenção e permitir que o cliente tenha acesso à tecnologia;
- Apresentar claramente as vantagens do sistema e estimulo o uso;
- Expandir para outros domínios de aplicação.

A metodologia INRECA faz uso de um paradigma de engenharia de software, chamado *experience factory* (Basili et al, 1994; citado por Bergmann & Althoff, 1998), que permite a reutilização de experiências de desenvolvimento de software. Este paradigma provê uma arquitetura organizacional para armazenar, acessar e estender orientações para desenvolvimento de aplicações RBC, permitindo reutilizar conhecimento adquirido em outras experiências em desenvolvimento de software.

A metodologia consiste em coletar experiências no desenvolvimento de aplicações RBC, representadas como modelos de processos de software e armazenar em uma base de experiências. Fornece também uma estrutura organizacional para acessar esta experiência e fazer uso do conhecimento que ela transmite.

2.4 CONCLUSÕES

A alta competitividade, neste mundo globalizado dos negócios, requer que as organizações incorporem novos canais de interação com os clientes para poderem sobreviver. O atendimento, com qualidade e que atenda às necessidades do cliente, é um fator de diferenciação de mercado, principalmente quando não existem diferenças entre produtos e serviços, e onde ocorrem situações de alta competitividade, de saturação e de mercados globalizados.

-venda. Sistemas de suporte RBC são altamente importantes para corporações que procuram estabelecer relacionamento, a longo prazo, com seus clientes.

Na literatura, são encontrados alguns sistemas de suporte, baseados em RBC, tais como *Prism: A Case Base Telex Classifier; Canasta: A Crash Analysis Troubleshooting Assistant; Smart: A Case Base Reasoning Call Support System* (Lewis, 1995). No ambiente de gerenciamento de redes corporativas e de telecomunicações, a abordagem de RBC tem sido muito aplicada a tratamento de falhas e correlação de alarmes (Penido, Nogueira & Machado, 1999; Melchiors & Tarouco, 1999; Weiner, Thurman & Mitchell, 1995; Lewis, 1995; Caulier & Houriez, 1995 e Stadler, 1993).

Algumas plataformas comerciais de gerenciamento, também, disponibilizam ferramentas para a criação de bases de conhecimento. Dentre estas, podem ser citadas: *Tivoly da IBM, TNG Unicenter da CA, Remedy, Vantive, PATROL da BMC, Spectrum da Cabletron*. Em geral, estas ferramentas permitem criar um histórico da ocorrência de problemas e armazenar um texto descritivo do problema e da solução proposta. Algumas fazem uso da abordagem de RBC, como é o caso das ferramentas Remedy e Spectrum.

Entretanto, de uma maneira geral, observa-se duas restrições importantes quanto a filosofia destas ferramentas:

- São softwares proprietários e não de domínio público;
- Dificuldade de importar e exportar a base de conhecimento.

Estas restrições criam uma dependência para o usuário em relação à ferramenta utilizada e limitam a troca de conhecimento, conseqüentemente inibem a evolução das ações de gerenciamento de redes.

CAPÍTULO - 3 ARQUITETURA TMN

3.1 INTRODUÇÃO

A fim de garantir a integração entre vários tipos de sistemas de operação e de redes de telecomunicações, o ITU-T (*International Telecommunications Union - Telecommunications*) definiu uma arquitetura de gerência para o ambiente de telecomunicações, sendo amplamente conhecida como TMN (*Telecommunications Management Network*) .

A TMN é uma arquitetura genérica para a implementação de sistemas abertos, voltados para a operação, administração, manutenção e provisionamento - OAM&P (*Operation, Administration, Maintenance & Provisionment*) das redes e serviços de telecomunicações. É descrita através de uma série de recomendações, denominadas M.3000. Esses documentos definem diferentes arquiteturas TMN, interfaces, serviços e funções.

As recomendações da série M.3000 é composta pelos documentos:

- M.3010 - Princípios para a Rede de Gerência de Telecomunicações;
- M.3020 - Metodologia para Especificação de Interface TMN;
- M.3100 - Modelo Genérico de Informação de Rede para TMN;
- M.3101 - Relatório de Conformidade para o Modelo de Informação Rede Genérica.
- M.3180 - Catálogo de Informação de Gerência TMN;
- M.3200 - Serviços de Gerência TMN;
- M.3300 - Capacidades de Gerenciamento TMN na Interface F;
- M.3320 - Requisitos de Gerenciamento para a interface X TMN;
- M.3400 - Funções de Gerenciamento TMN.

A TMN faz uso dos princípios de gerente, agente e objeto gerenciado do modelo OSI (*Open System Interconnection*) para gerência de redes de computadores (X.701 | ISO/IEC 10040), além de incluir especificações para o serviço de gerenciamento CMIS (*Common Management Information Service*) (X710 | ISO/IEC 9595), para o protocolo de gerenciamento CMIP (*Common Management Information Protocol*) (X.711 | ISO/IEC 9596), e para a representação das informações de gerência através de GDMO (*Guidelines for the Definition of Managed Objects*) (X.722 | ISO/IEC 10165-4).

Na prática, a TMN é implementada através de uma rede de computadores, onde aplicações de gerência monitoram e controlam um ambiente de rede de telecomunicações. Este ambiente é definido a partir dos equipamentos telecomunicações digitais e analógicos, dos equipamentos de infra-estrutura associados e dos serviços oferecidos aos usuários.

A TMN é uma rede conceitualmente separada que interage com a rede de telecomunicações em vários pontos, através de interfaces padronizadas, podendo utilizar parte da rede de ar suas funções. (Figura 3.1). A TMN considera as redes e os serviços de telecomunicações como um conjunto de sistemas cooperativos e visa gerenciá-los de forma integrada.

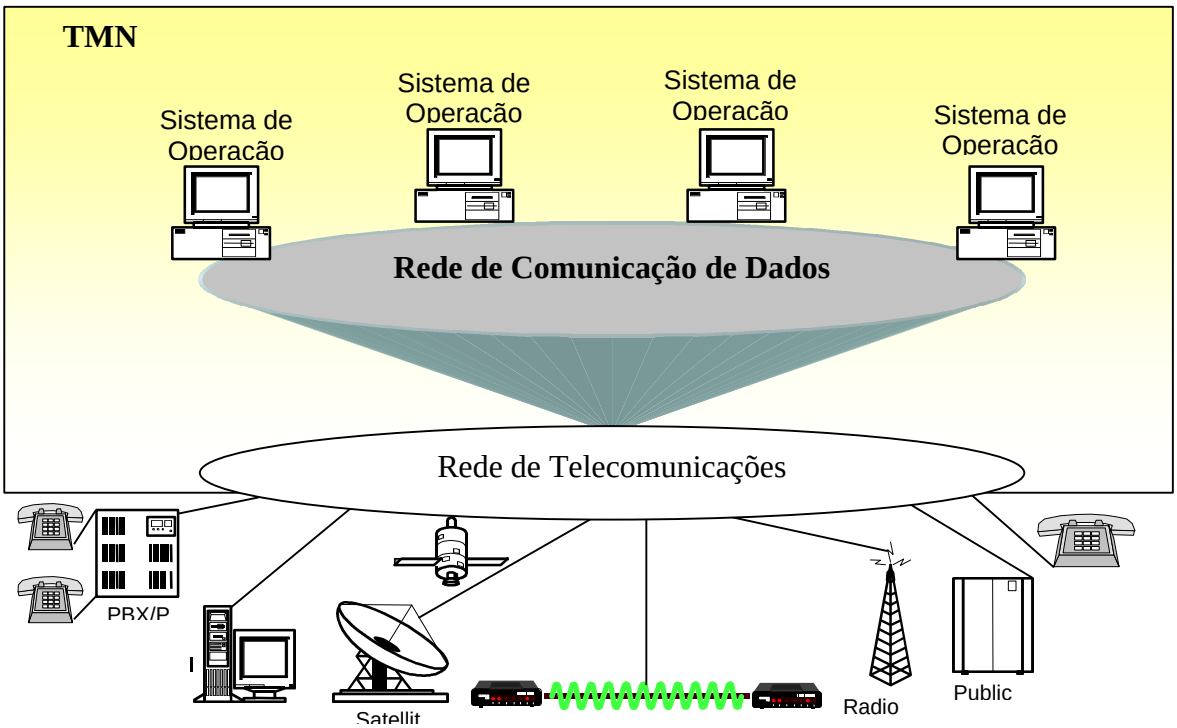


Figura 3.1 - Relacionamento entre a TMN e a Rede de Telecomunicações

Neste capítulo, a Arquitetura TMN é introduzida, bem como seus conceitos básicos, seus aspectos estruturais, funcionais, físicos e de informação. Além das recomendações TMN, destaca-se (Udupa, 1999; Aidarous & Plevyak, 1998; Bartholomew, 1997; Mattison, 1997 e Adams & Willets, 1996) como referências de estudo sobre TMN e o ambiente de telecomunicações:

Na seção 3.2 é descrito o ambiente de gerenciamento. A seção subsequente 3.3 apresenta a estrutura de camadas da TMN. Dentro do planejamento e do desenvolvimento de uma TMN, três aspectos básicos são considerados separadamente: arquitetura funcional, arquitetura física e arquitetura de informação. As seções 3.4, 3.5, 3.6 descrevem estas arquiteturas.

3.2 AMBIENTE DE GERENCIAMENTO

O ambiente das redes de telecomunicações é distribuído, consequentemente o seu gerenciamento dá-se por um conjunto de aplicações distribuídas.

Neste contexto, os processos de aplicações de gerência podem assumir os seguintes papéis:

- Gerente: parte da aplicação distribuída que solicita operações de gerenciamento e que recebe notificações;
- Agente: parte da aplicação distribuída que é responsável por executar as operações de gerenciamento sobre os recursos passíveis de gerência e por emitir notificações de

Gerentes e agentes trocam informações de gerência através protocolo de gerência CMIP (X.711 | ISO/IEC 9596), conforme Figura 3.2 Os recursos passíveis de gerenciamento podem ser lógicos ou físicos (ex. central de comutação ou um arquivo de *log*). Estes recursos são modelados e representados através de objetos gerenciados. O conjunto de objetos gerenciados forma a base de informações de gerenciamento - MIB (*Management Information Base*).

Um sistema de gerenciamento pode ser composto por um ou mais processos gerente comunicando-se com um ou vários processos agente. Por sua vez, um processo agente também pode trocar informações com um ou mais processos gerente.

A comunicação entre processos gerente e agente dá-se através de protocolos padrões. No caso da TMN, é o protocolo CMIP. Entretanto a troca de informações entre processos agentes e os objetos gerenciados não é padronizado, ficando a cargo de cada implementação.

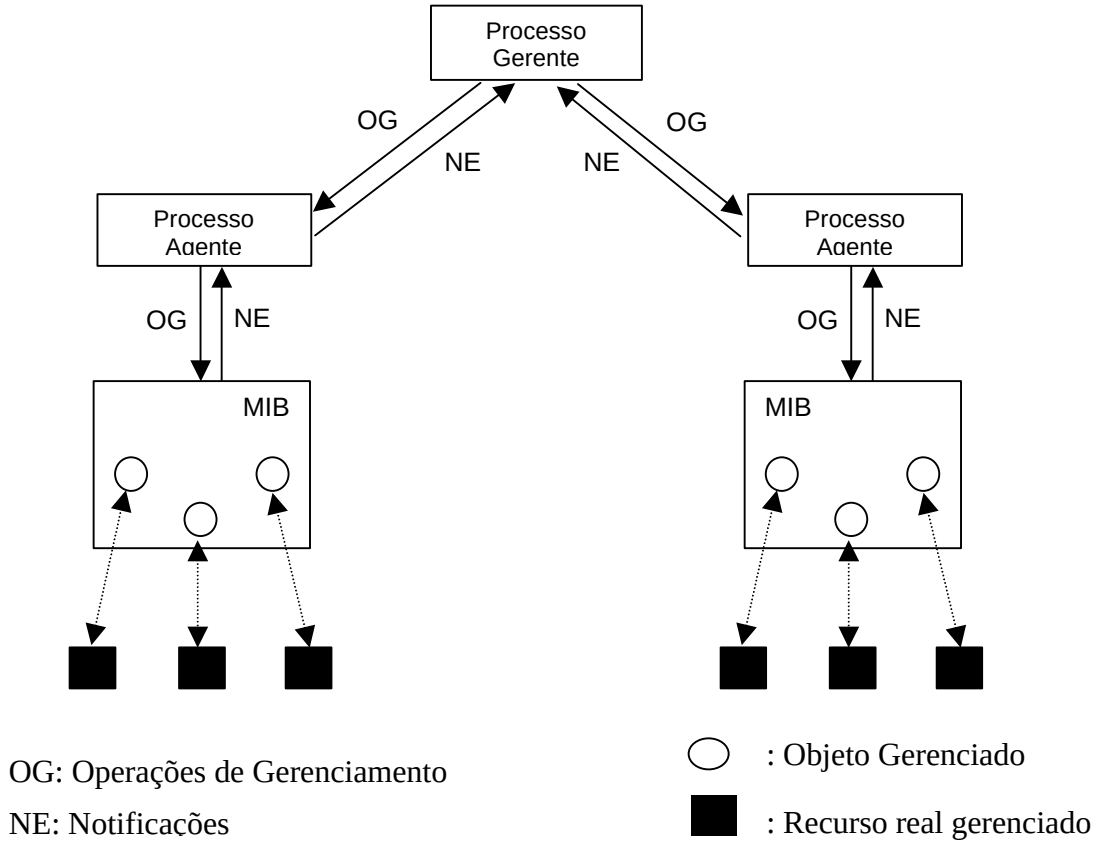


Figura 3.2 - Relacionamento Gerente e Agente

3.3 CAMADAS DE GERENCIAMENTO TMN

Dada a complexidade do ambiente de telecomunicações, a recomendação M.3010 apresenta uma estrutura hierárquica, dividida funcionalmente em camadas de gerenciamento, que facilita a identificação das atribuições, responsabilidades e abrangências de cada processo e usuário da TMN. Cada camada tem seus próprios objetivos, funções e atribui

além de regras para interagir com as outras camadas. O escopo das camadas superiores é maior que o das camadas inferiores. Os níveis inferiores executam funções mais específicas no contexto TMN.

Baseado no trabalho originalmente desenvolvido, em 1987, por um grupo da British Telecommunicatios, liderado por Keith Willets (Adams & Willets, 1996), a TMN é estruturada em cinco camadas funcionais de gerenciamento (Figura 3.3):

- Gerência de Negócio: relacionada ao negócio da empresa, fornece elementos que servem de parâmetros para o planejamento estratégico, a definição das políticas da empresa, os investimentos, as vantagens competitivas, os segmentos de mercados, etc.
- Gerência de Serviço: monitorar e controlar os serviços oferecidos, garantindo performance, qualidade, redução de custos, etc.;
- Gerência de Rede: visa garantir a conectividade e desempenho da rede, consequentemente garante a prestação dos serviços;
- Gerência de Elemento de Rede: monitora e controla os segmentos de rede, os elementos de rede individualmente;
- Elemento de Rede: são os processos de agentes conectados diretamente aos objetos gerenciados. Executa funções de coleta de dados, de diagnósticos, de tratamento de alarmes, conversão de protocolos, resolução de endereços, conversão de dados, etc.

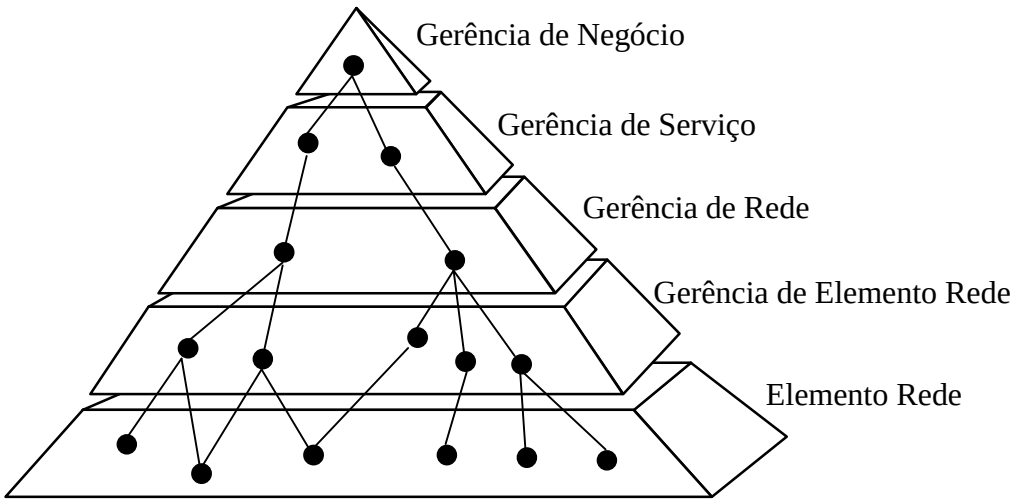


Figura 3.3 - Estrutura de Camadas TMN

3.4 ARQUITETURA FUNCIONAL

A arquitetura funcional descreve a distribuição funcional requerida pela TMN, reduzindo a complexidade e viabilizando a implementação da TMN. Esta arquitetura introduz três locos funcionais, componentes funcionais e pontos de referência. A partir dos blocos funcionais e dos pontos de referencia entre os blocos, são especificadas as interfaces padrão TMN (Schonberger, 1998).

3.4.1 BLOCOS FUNCIONAIS

O domínio da TMN é dividido em diferentes blocos funcionais. Cada bloco funcional executa uma função específica de gerenciamento. Alguns dos blocos funcionais são internos e outros externos ao domínio da TMN (Figura 3.4).

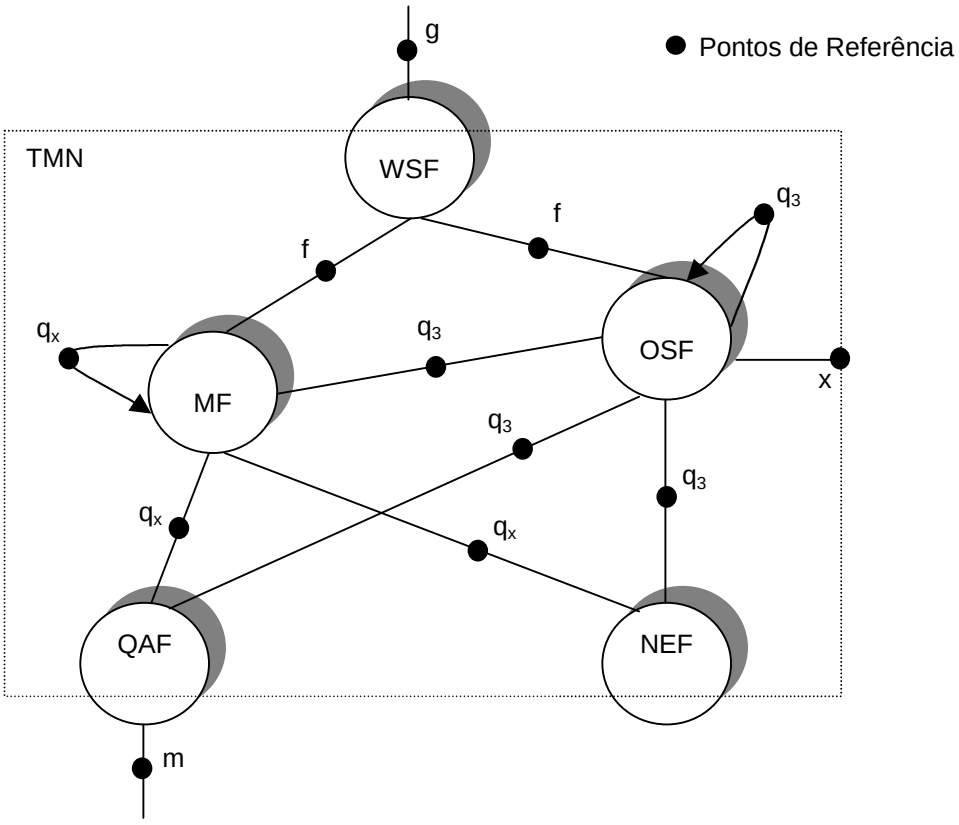


Figura 3.4 - Arquitetura Funcional TMN (IEEE, 1994)

Os blocos funcionais são:

- OSF (*Operations Systems Function*): Bloco Funcional Sistema de Suporte a Operações. Processa informações relacionadas ao gerenciamento de telecomunicações para fins de monitoramento, coordenação e/ou controle das funções de telecomunicações, incluindo as prias funções TMN. O bloco OSF está associado a alguma aplicação de gerenciamento.

- NEF (*Network Element Function*): Bloco Funcional Elemento de Rede. Componentes de rede de telecomunicações são representados por um ou mais blocos NEF para fins de gerenciamento. Os blocos NEF são monitorados e controlados pela TMN.

- WSF (*Workstation Function*): Bloco Funcional Estação de Trabalho. Representa a funcionalidade que permite a interface com os usuários de gerência a TMN.

- QAF (*Q Adaptor Function*): Bloco Funcional Adaptador Q. Conecta à TMN entidades -TMN, ou seja que não suportam interfaces padronizadas TMN. O bloco adapta entidades proprietárias a entidades padrão TMN.

- MF (*Mediation Function*): Bloco Funcional Mediação. É um tipo de *gateway* entre os blocos funcionais OSF e NEF (ou QAF) para garantir a compatibilidade da informação trocada entre estes blocos. Pode adaptar, armazenar, filtrar, endereçar, rotear e condensar

3.4.2 COMPONENTES FUNCIONAIS

Cada bloco funcional é composto de um número de componentes funcionais que representam os elementos básicos que implementam a funcionalidade de cada bloco. Os diferentes

- MAF (*Management Application Function*): Função Aplicada de Gerência. Implementa os serviços de gerenciamento. Também inclui o gerenciamento de informação. Pode atuar no papel de processo gerente ou processo agente. Dependendo do bloco funcional ao qual

integra, é referenciada como MF-MAF, OSF-MAF, NEF-MAF e QAF-MAF, respectivamente.

- ICF (*Information Conversion Function*): Função de Conversão de Informação. Provê mecanismos para converter modelos de informação. Converte representações entre objetos gerenciados. A tradução pode ser ao nível de sintaxe e/ou semântica.

- MCF (*Message Communication Function*): Função de Comunicação de Mensagem. Usada para a troca de mensagens entre os blocos funcionais. É necessário para todo bloco funcional que necessita de uma interface física. Faz uso de uma pilha de protocolos Este protocolo não necessariamente tem que ser a pilha OSI. Pode ser c
  função DCF e basicamente provê mecanismos de transporte de informação. A recomendação M.3010 apresenta o relacionamento entre blocos funcionais, através dos componentes funcionais MCF e DCF (Figura 3.5).

- DCF (*Data Communication Function*): Função de Comunicação de Dados. Provê funções de interconexão, roteamento, retransmissão. DCF provê as camadas 1 a 3 do modelo de referência OSI ou seus equivalentes. Pode ser enlaces ponto a ponto, redes locais e/ou redes de longa distância (Udupa, 1999).

- WSSF (*Workstation Support Function*): Função de Suporte à Estação de Trabalho. Requerida para implementação do bloco funcional WSF, incluindo acesso e manipulação de dados, invocação e confirmação de ações, transmissão de notificações, e torna transparente para o usuário WSF a existência de NEFs e outros OSFs que não aquele que ele se comunica.

- UISF (*User Interface Support Function*): Função de Suporte a Interface de Usuário. Traduz a informação do usuário para o modelo de informação TMN e vice-versa. É responsável por apresentar a informação de forma correta e consistente com a interface de

- DSF (*Directory System Function*): Função Sistema de Diretório. É baseado na recomendação X.500. Contém informações sobre sistemas com os quais podem ser feitas associações.

- DAF (*Directory Access Function*): Função de Acesso a Diretório. Requerida por todos os blocos funcionais que necessitam acessar ao diretório.

- SF (*Security Function*): Função de Segurança. Provê segurança aos blocos funcionais Os e segurança são autenticação, confiabilidade, controle de acesso, integridade, e -repudiação.

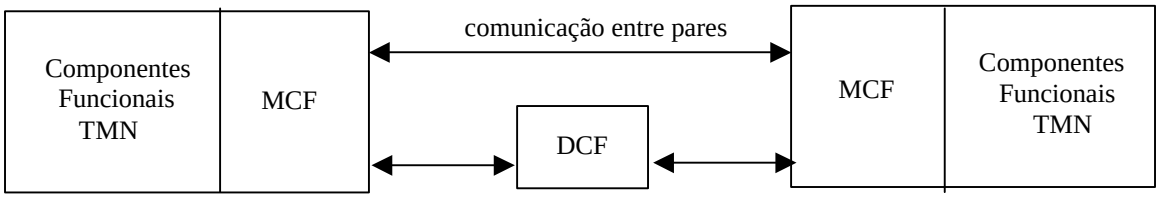


Figura 3.5 - Relacionamento entre MCF e DCF

A Tabela 3.1 apresenta os componentes funcionais possíveis em cada bloco funcional.

Blocos Funcionais	Componentes Funcionais
OSF	OSF-MAF (A/M), WSSF,ICF, DSF, DAF, SF
WSF	UISF, DAF, SF
MCF	NEF-MAF(A), DSF, DAF, SF
MF	MF-MAF (A/M), ICF, WSSF, DSF, DAF, SF
QAF	QAF-MAF (A/M), ICF, DSF, DAF, SF
A/M - Agent/Manager DAF - Directory Access Function DSF - Directory System Function ICF - Information Conversion Function MCF - Message Communication Function MAF - Management Application Function SF - Security Function UISF - User Interface Support Function WSSF - WorkStation Support Function	

Tabela 1 - Relação entre Blocos Funcionais e Componentes Funcionais

3.4.3 PONTOS DE REFERÊNCIA

Os blocos funcionais são separados por limites conceituais conhecidos como pontos de referências. Caracterizam as fronteiras entre os blocos funcionais. Visam identificar as informações trocadas entre blocos funcionais. Existem três classes de pontos de referência:

- Classe *q* - entre OSF, QAF, MF e NEF;
- Classe *f* - para ligação de estações de trabalho (ou WSF);
- Classe *x* - entre OSFs de duas TMNs ou entre uma OSF de uma TMN e um bloco funcional com funcionalidade equivalente de outra rede.

São definidas ainda outras duas classes de pontos de referência que não pertencem à TMN mas também são muito importantes:

- Classe *g* - entre a estação de trabalho e o usuário;
- Classe *m* - entre QAF e entidades não TMN;

Existem ainda dois tipos de pontos de referência classe *q*:

- *Q3*: limites da TMN.
- *Qx*: fora dos limites da TMN.

3.4.4 ESTRUTURA FUNCIONAL

Dentro da visão estruturada em camadas da TMN, a base da hierarquia de gerenciamento, camada de gerência de elemento de rede, é formada por blocos funcionais NEFs. As outras camadas são formadas por blocos funcionais OSFs (Figura 3.6).

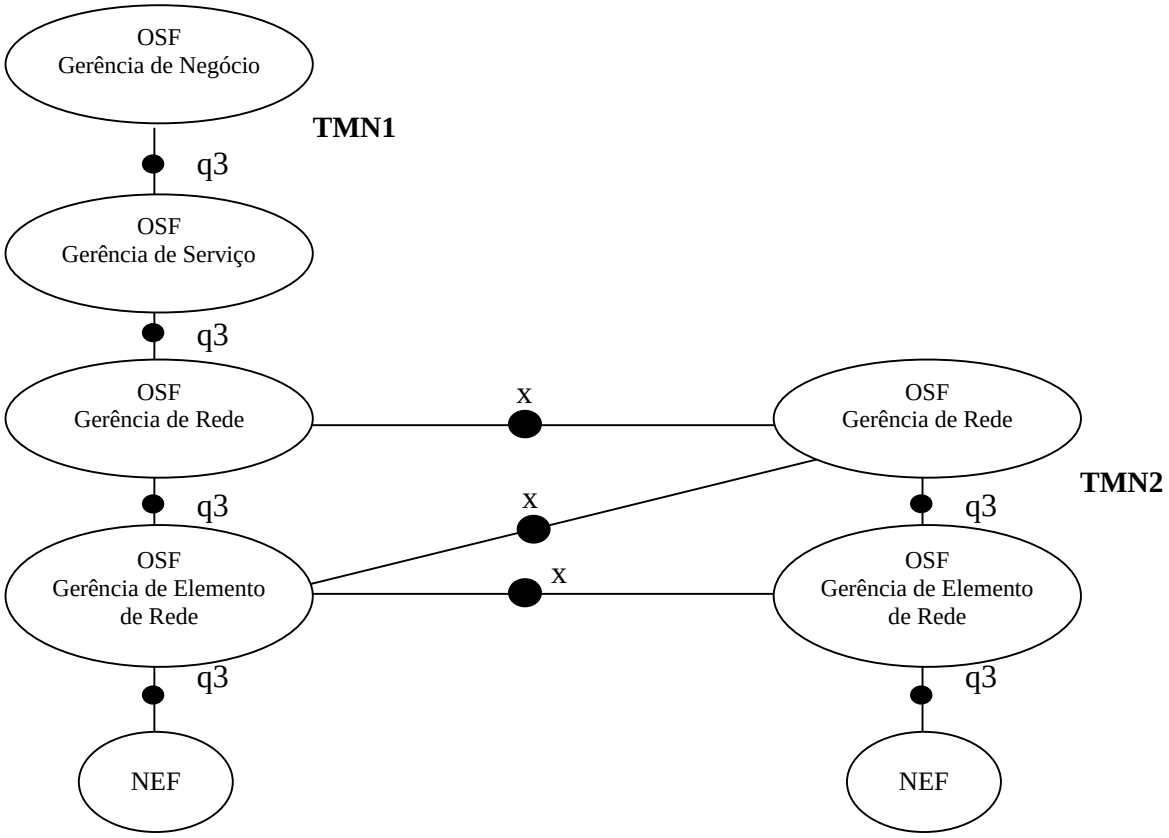


Figura 3.6 - Estrutura Funcional e os Blocos Funcionais (Udupa, 1999)

3.5 ARQUITETURA FÍSICA

A Arquitetura Física define os elementos que servem de suporte a implementação dos blocos funcionais TMN. Diferentes configurações físicas podem ser projetadas para dar suporte a funcionalidade TMN. A Figura x apresenta um exemplo simplificado. Para cada componente físico tem-se um bloco funcional associado. Os componentes físicos são:

- OS (*Operation System*): provê suporte ao processamento de informações relacionadas às atividades de OAM&P das redes de telecomunicações. Os sistemas OSs são análogos ao processos gerentes. Está associado ao bloco funcional OSF;
- DCN (*Data Communication Network*): compreende a conexão física e a rede onde as informações de gerência trafegam. Está associado ao bloco funcional DCF;

- MD (*Mediation Device*): inclui funções tais como conversão de protocolo, conversão de mensagens, conversão de sinal, tradução de endereços e roteamento implementação das funções de mediação MFs;

- WS (*WorkStation*): é a interface de entrada e saída que permite aos usuários TMN acessarem informações de gerência. Permite a interação com OSs e MFs. Está associado ao bloco funcional WSF;

- NE (*Network Element*): corresponde aos equipamentos e às facilidades TMN passíveis de gerenciamento. É similar ao processo agente e está associado ao bloco funcional NEF;

- QA (*Q Adaptor*): adaptador que converte dados em formato não padrão TMN para o e vice-versa. Por exemplo, mensagens TL1, de dispositivos não padronizados, são convertidas para o formato CMIP e transmitidas via DCN.

Os pontos de referência da Arquitetura Funcional são implementados através das interfaces. As interfaces interconectam os elementos da Arquitetura Física de forma padronizada (Figura 3.7). As interfaces padronizadas TMN são:

- Interface Q: aplicada no ponto de referência Q. Pode ser Q_3 ou Q_x. Interfaces Q_3 fazem uso de toda a pilha de protocolos das sete camadas do modelo de referência OSI (ISO/IEC 7498). É especificada através das recomendações Q.811 e Q.812. Interface Q_x é uma interface mais simples, voltada para elementos que não suportam Q_3 ou que não necessitam usar toda a pilha de protocolos do modelo de referência OSI.

- Interface X: usada para interconectar OS em diferentes sistemas TMN ou para conectar um sistema TMN a outro sistema de gerenciamento com interfaces tipo TMN.

- Interface F: interface que provê acesso para os usuários. Interconecta WSSF, OSF e MF via DCN.

- Interface M: está fora do domínio da TMN, provê conexão entre um QAF e uma entidade

- Interface G: não é considerada como parte da TMN, provê acesso ao usuário via WS.

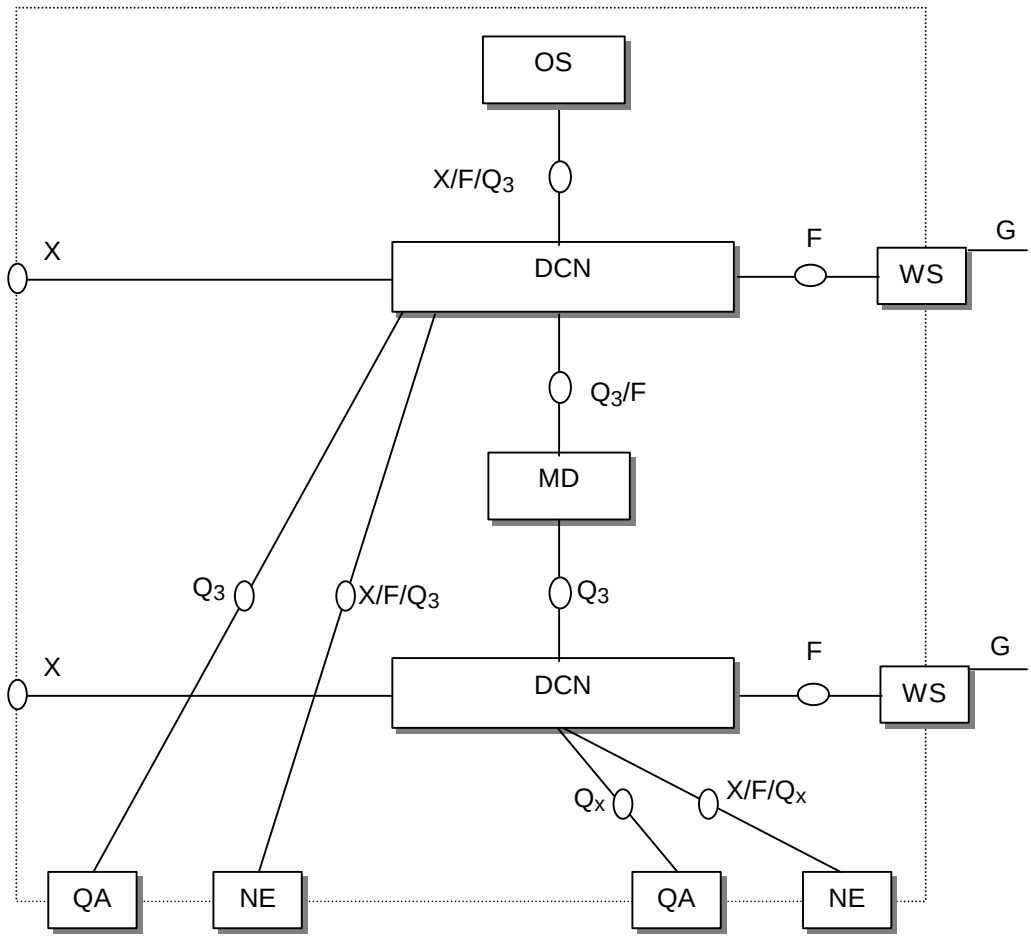


Figura 3.7 - Arquitetura Física TMN

3.6 ARQUITETURA DE INFORMAÇÃO

A Arquitetura de Informação fornece uma estrutura para a informação de gerenciamento. Descreve como os princípios de gerenciamento de sistemas OSI e princípios X.500 podem ser aplicados à TMN para estruturar as informações transportadas pelos protocolos de gerenciamento e para modelar os recursos passíveis de gerenciamento.

Conforme visto na seção 3.2, agentes e gerentes trocam informações dos recursos passíveis de gerência, usando protocolos de gerenciamento. Estes protocolos na TMN estão baseados no

compartilha as mesmas propriedades com outros membros desta mesma classe.

Um conjunto de classes genéricas de objetos gerenciados é definido na Recomendação M.3100 e podem ser usadas para diferentes aplicações TMN. Entretanto, novas classes podem ser derivadas de acordo com as necessidades de cada aplicação. O uso da metodologia definida na Recomendação M.3020 permite identificar extensões adicionais requeridas de acordo com as particularidades de cada sistema.

As classes derivadas são chamadas de subclasses e as classes das quais derivaram são chamadas de superclasses. As subclasses são especializações de suas superclasses, onde herdam suas características e acrescentam novas propriedades, obedecendo às regras de compatibilidade e herança definidas pelas normas (X.720 | ISSO/IEC 10165-1). O relacionamento entre classes resulta em uma hierarquia, chamada de árvore de hierarquia de

herança (Figura 3.8), tendo como ponto inicial a classe *top* que é a superclasse de todas as outras classes. Esta classe não é instanciável.

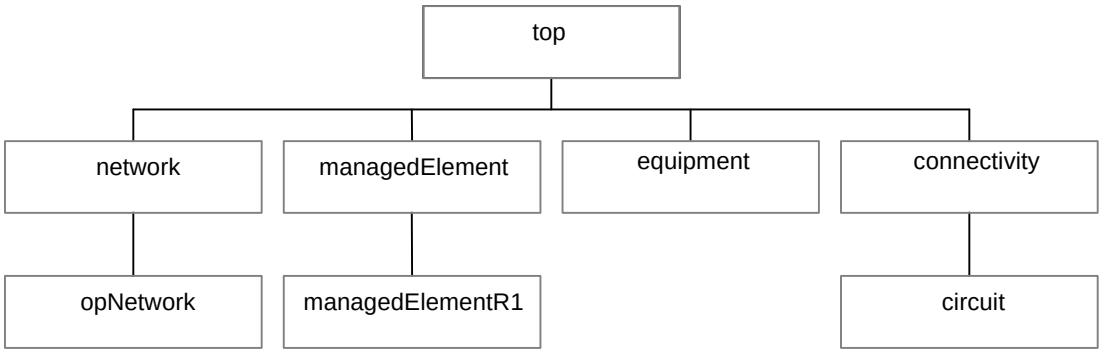


Figura 3.8 - Hierarquia de Herança

Um objeto gerenciado de uma classe pode conter outros objetos gerenciados de uma mesma classe ou de outras classes. Este relacionamento é chamado de *containment*, ocorre apenas entre instâncias de objetos e não entre classes, podendo ser visualizado através da árvore de hierarquia de nomeação (Figura 3.9). Esta árvore de nomeação tem como ponto inicial um objeto nulo e sempre existente, chamado de *root* (raiz).

A árvore de nomeação é usada também para identificar os objetos gerenciados. Todo objeto gerenciado deve ter um único nome para identificá-lo. Uma estrutura de nomes locais - RDN (*relative distinguish name*) e globais - DN (*distinguish name*) é usada para nomeação. Cada objeto, particularmente, dentro de uma árvore de *containment*, é identificado com um nome local RDN. O nome global deste objeto é atribuído a partir do objeto *root*, através da concatenação dos RDNs de suas superclasses, até o objeto gerenciado em questão.

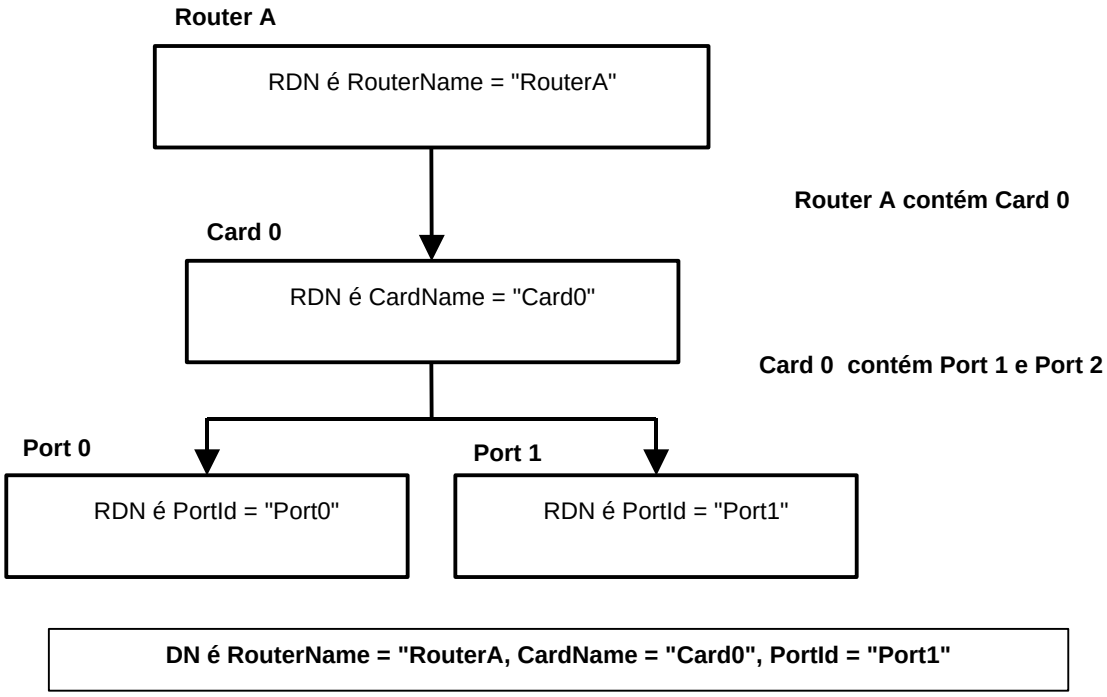


Figura 3.9 - Hierarquia de Nomeação

3.6.1 TERMOS E CONCEITOS

3.6.1.1 Classe de Objetos Gerenciados

Objetos gerenciados que possuem características similares são agrupados dentro de uma mesma classe de objetos. Uma classe de objetos gerenciados é uma coleção de pacotes. Cada coleção de atributos, operações, notificações e comportamento. Pacotes podem ser obrigatórios ou condicionais.

A definição de classes de objetos, como especificado nos *Templates*, consiste de:

- Posicionamento da classe dentro da hierarquia de herança, isto é, identificação da superclasse da qual as classes de objetos são derivadas, herdando suas características;
- Os pacotes obrigatórios contendo atributos, operações, notificações e comportamentos;

- Os pacotes condicionais contendo atributos, operações, notificações e comportamentos, juntamente com a condição na qual o pacote estará presente.

3.6.1.2 Pacotes

Um pacote é uma coleção de caraterísticas, que constitui na definição de uma classe de objetos. Os pacotes podem ser obrigatórios ou condicionais. Um pacote obrigat
presente em todas as instâncias de uma classe de objetos. Um pacote condicional é um pacote que deve estar presente em um objeto, somente quando a condição associada a existência ou não do pacote é satisfeita. Os pacotes possuem as seguintes propriedades:

- Somente uma instância de um determinado pacote pode existir em um objeto;
- Os pacotes estão encapsulados no objeto, portanto são acessíveis somente como parte daquele objeto;
- Um pacote não pode ser instanciado sem o objeto que o encapsula;
- Um pacote deve ser instanciado no mesmo momento que o objeto que o encapsula;
- Pacotes devem ser removidos juntamente com a remoção do objeto;
- Operações são realizadas em cima dos objetos gerenciados, em não em cima dos pacotes.

3.6.2 ATRIBUTOS

Os dados que são encapsulados em um objeto gerenciado são chamados atributos. Cada atributo corresponde a uma das características do recurso que o objeto representa. O atributo é formado por um nome, um tipo e um ou mais valores que refletem o estado corrente do recurso gerenciado. A sintaxe de um atributo deve ser descrita em ASN.1. Os valores dos atributos podem ser lidos ou modificados.

3.6.3 COMPORTAMENTO

Objetos gerenciados podem sofrer eventos internos ou externos. O comportamento de um objeto descreve como ele deve reagir quando na ocorrência de determinados eventos. O comportamento de uma classe de objetos define:

- A semântica dos atributos, operações e notificações;
- A resposta das operações de gerência;
- As circunstâncias na qual as notificações serão emitidas;
- Os valores de atributos que decidirão na ausência ou presença de um pacote condicional;
- As restrições de consistência de um atributo;
- As pré-condições que identificam as condições que as operações e notificações podem assumir para serem consideradas válidas;
- As pós-condições que identificam o resultado de uma operação ou notificação;
- As propriedades de sincronismo dos objetos.

3.6.4 ESPECIALIZAÇÃO E HERANÇA

Uma classe de objetos é uma especialização de outra quando possui uma extensão das propriedades da outra classe. A herança é a propriedade onde uma classe mais especializada (subclasse) herda as propriedades da classe mais geral (super classe). A subclasse herda as operações, notificações, pacotes e comportamento da super classe. Herança múltipla é a habilidade de uma classe herdar características de uma ou mais classes superiores.

3.6.5 OPERAÇÕES DE GERÊNCIA

As operações que podem ser realizadas em um objeto gerenciado devem fazer partes da sua definição, assim como os efeitos causados no recurso gerenciado. Uma operação de gerência em um objeto somente se concretizará se o sistema invocador da operação possuir as autorizações necessárias para a realização da operação e se as regras de consistência não forem violadas.

As operações podem ser confirmadas ou não confirmadas, isto é, exigir ou não uma resposta de retorno, indicando se a operação foi bem sucedida ou não. Existem dois tipos de operações de gerência: operações sobre atributos e operações sobre o objeto gerenciado como um todo.

As operações sobre atributos podem ser:

- *Get attribute value*: ler uma lista ou todos os valores dos atributos especificados;
- *Replace attribute value*: alterar os valores dos atributos especificados com novos valores;
- *Replace-with-default value*: substituir o valor de determinados atributos para seu valor default;
- *Add member*: substituir o conjunto de valores existentes pelo conjunto que será resultado da união do conjunto existente com o novo conjunto especificado;
- *Remove member*: para cada conjunto especificado de valores de atributos, esta operação substitui o conjunto de valores existente pela diferença entre o conjunto existente e o novo conjunto especificado.

As operações sobre o objeto, como um todo, podem ser:

- *Create*: cria e inicia um objeto gerenciado;

- *Delete*: Remove um objeto gerenciado;
- *Action*: Solicita a um objeto que execute uma determinada ação e retorne o resultado.

3.6.6 NOTIFICAÇÕES

Os objetos gerenciados reportam seu estado através de notificações, em resposta a algum evento interno ou externo. As notificações podem ou não, serem transmitidas ao processo de gerenciamento. O envio de uma notificação depende da satisfação de condições que podem ser definidas como parte da notificação.

3.6.7 NAME BINDING

Possibilita a definição da estrutura de nomeação hierárquica onde um objeto de uma dada inado a um outro objeto de uma outra classe, sendo que em tal hierarquia o objeto subordinado aponta para o objeto superior.

3.7 FORMALIZAÇÃO DO MODELO DE INFORMAÇÃO

A informação de gerenciamento deve ser apresentada formalmente, através de ferramentas apropriadas, visando a identificação e documentação de todos os seus elementos, para que possa ser transladada, sem ambigüidades, para qualquer forma de implementação.

3.7.1 CONSTRUÇÃO DOS TEMPLATES

Identificados os objetos gerenciados e seus componentes, definidos os relacionamentos de herança e *conteainment*, formaliza-se tudo através de um conjunto de *templates* estabelecidos pela Recomendação X.722 (X.722 | ISSO/IEC 10165-4).

Esses *templates* fornecem uma notação para representação formal de vários aspectos da ão de gerenciamento, possibilitando a especificação de identificadores, relacionamentos de herança e retenção, comportamento, agrupamentos de componentes, modo como o valor de um atributo pode ser testado, sintaxe dos atributos e das notificações, etc. Possibilitam, também, assinalar um nome para registro e documentação das instâncias desses moldes.

Os templates para definição de classes de objetos gerenciados, atributos, pacotes, *naming-bindings* e notificações derivam desse template genérico. A Figura 3.10 apresenta a estrutura genérica do *templates.*

```
<template-label>   TEMPLATE-NAME
                   CONSTRUCT-NAME          [<construct-argument>]
                   [CONSTRUCT-NAME         [<construct-argument>]]
[REGISTERED AS     <object-identifier>]
```

Figura 3.10 - Estrutura genérica dos templates

3.7.1.1 Template para Classes de Objetos Gerenciados

Este *template* fornece a base para a definição formal das classes de objetos gerenciados, possibilitando a descrição dos seus componentes (obrigatório e condicionais) e o relacionamento de herança entre as classes (Figura 3.11).

```
<classe-label>  MANAGED OBJECT CLASS
 [DERIVED FROM    <nome da superclasse>]
[CARACTERIZED BY
                  <atributos>
                  <pacotes>
                  <operações>
                  <ações>
                  <notificações>]
 [CONDITIONAL PACKAGES
                  <pacotes condicionais>]
 [REGISTRED AS    <identificador do objeto>]
```

Figura 3.11 - Template para classe de objetos gerenciados

3.7.1.2 Template para Atributos

Caracteriza-se por definir o tipo abstrato de dado do atributo e a forma como ele pode ser comparado (Figura 3.12).

```
<attribute-label>    MANAGED OBJECT CLASS
                  WITH ATTRIBUTE SINTAXE  <tipo abstrato de dado>
                  MATCHES FOR             <qualificador>]
 [REGISTRED AS   < identificador de atributo>]
```

Figura 3.12 - Template para Atributos

3.7.1.3 Template para Pacotes

Permite agrupar um conjunto de atributos, ações e notificações que podem ser inseridos como obrigatórios ou opcionais em uma instância de um molde de classe de objeto gerenciado. Cada atributo incluído indica as operações que ele pode suportar (Figura 3.13).

```
<package-label>    PACKAGE
                   [ATTRIBUTES          <atributos>]
                   [ACTIONS             <ações>]
                   [NOTIFICATIONS       <notificações]
[REGISTRED AS    <identificador do pacote>]
```

Figura 3.13 - Template para pacotes

3.7.1.4 Template para Naming-binding

Através deste *template* é possível formalizar o relacionamento de retenção e estabelecer uma hierarquia de nomes para os objetos gerenciados (Figura 3.14).

```
<name-binding-label>    NAME BINDING
                SUBORDINATE OBJECT CLASS       <nomes das classes subordinadas>
                SUPERIOR OBJECT CLASS          <nomes das classes superiores>
                WITH ATTRIBUTE            <nome do atributo para construção do RDN>]
REGISTRED AS   <identificador do naming binding>
```

Figura 3.14 - Template para name-binding

3.7.1.5 Template para Notificações

Utilizado para formalizar a sintaxe e os atributos associados a cada tipo de notificação, de forma que possibilite a comunicação de eventos (Figura 3.15).

```
<notification-label>  NOTIFICATION
                MODE   <modo de confirmação>
                [WITH INFORMATION SINTAXE   <sintaxe>]
REGISTERED AS  <identificador da notificação>
```

Figura 3.15 - Template para notificações

3.7.2 FORMALIZAÇÃO ATRAVÉS DA ASN.1

A ISO desenvolveu uma linguagem abstrata formal, a ASN.1 (X.208 | ISSO/IEC 8824) para estruturar a informação transferida pelas entidades (protocolos) da camada de aplicação, através da padronização de um conjunto próprio de tipos abstratos de dados.

Um tipo abstrato de dados é um conceito utilizado para descrever uma estrutura de dados de forma bem definida e padronizada. Embora a representação da estrutura de dados possa ter uma representação concreta num determinado sistema local (exemplo, uma *struct* na linguagem de programação “C”), sua sintaxe é definida de maneira independente da

Assim, o modelo de gerenciamento OSI utiliza-se dos formalismos estabelecidos pela ASN.1 na definição de uma estrutura para a informação de gerenciamento, de forma que:

- Possa ser transportada remotamente e entendida por entidades pares de gerenciamento, independentemente de linguagens de programação e compilação e da arquitetura da trem essas entidades;
- Seja possível num domínio de gerência, a um sistema aberto conhecer, sem ambigüidades, a informação de gerenciamento, contida em outro sistema aberto.

Para atender os requisitos acima e compatibilizar os dados de gerenciamento armazenados na MIB com os tipos de dados abstratos manipulados pelos protocolos de aplicação, é importante, embora não obrigatório, que a informação de gerenciamento seja formalizada dentro de módulos ASN.1, de modo que estes possam ser mapeados para estruturas de uma linguagem de programação.

Num módulo devem ser agrupados os objetos gerenciados que se relacionem na busca de um objeto comum (por exemplo, os objetos gerenciados referentes a uma das cinco funcionalidades). A estrutura genérica de um módulo ASN.1 tem o formato apresentado na Figura.3.16.

```
<<module>>  DEFINITIONS ::= BEGIN
            <<linkage>>
            <<declarations>>
END
```

Figura 3.16 - Módulo ASN.1

No termo <<module>> é especificado o nome do módulo. O termo <<linkage>> permite que módulos, colocados numa biblioteca, possam ser importados por e exportados para outros <<declarations>> contém as definições dos tipos de dados e dos valores, os quais são instâncias de um tipo de dado, das informações contidos no módulo.

A linguagem ASN.1 define quatro classes de tipos abstratos de dados: universal, específicas a um contexto, uso privado e *application-wide*. O modelo de gerenciamento OSI utiliza os tipos definidos na classe universal como demonstra a Tabela.3.2.

UNIVERSAL 1	BOOLEAN
UNIVERSAL 2	INTEGER
UNIVERSAL 3	BIT STRING
UNIVERSAL 4	OCTET STRING
UNIVERSAL 5	NULL
UNIVERSAL 6	OBJECT IDENTIFIER
UNIVERSAL 7	Object Descriptor
UNIVERSAL 8	EXTERNAL
UNIVERSAL 9	REAL
UNIVERSAL 10	ENUMERATED
UNIVERSAL 16	SEQUENCE, SEQUENCE OF
UNIVERSAL 17	SET, SET OF
UNIVERSAL 18	NumericString
UNIVERSAL 19	PrintableString
UNIVERSAL 20	TelexString
UNIVERSAL 21	VideoString
UNIVERSAL 22	IA5String
UNIVERSAL 23	UTCTime
UNIVERSAL 24	GeneralizedTime
UNIVERSAL 25	GraphicsString
UNIVERSAL 26	VisibleString
UNIVERSAL 27	GeneraString
UNIVERSAL 28	CharacterString

Tabela 2 - Tipo de dados “Universal” da ASN.1

3.8 SERVIÇOS E FUNÇÕES DE GERÊNCIA

Do ponto de vista da TMN, os serviços e as funções de gerência são descritos pelas Recomendações M.3200 e M.3400 respectivamente. O relacionamento entre os serviços e as funções de gerência é descrito na Recomendação M.3020.

Os serviços TMN são modelados de acordo com as percepções do usuário e provêm suporte para as atividades de OAM&P de uma rede de telecomunicações. A Recomendação M.3200 os de gerência já identificados, mostrados na Tabela 3.3:

Serviços de Gerência TMN definidos pela recomendação M.3200	
1	Administração de cliente
2	Administração de encaminhamento e análise de dígitos
3	Administração de medidas e análise de tráfego
4	Administração de tarifas e faturamento
5	Gerência de segurança TMN
6	Gerência de tráfego
7	Gerência do acesso do cliente
8	Gerência da rede de transporte
9	Gerência da comutação
10	Gerência dos equipamentos em instalações dos clientes
11	Administração de instalação dos sistemas
12	Administração da qualidade do serviço e do desempenho da rede
13	Gerência do serviço controlado pelo cliente
14	Gerência do sistema de sinalização do canal comum
15	Gerência de redes inteligentes
16	Restauração e recuperação
17	Gerência de materiais
18	Gerência da força de trabalho
19	Gerência da TMN

Tabela 3 - Serviços de Gerência TMN

Alguns destes serviços já têm descrito seus requisitos, objetivos e contextos de gerência pelo ITU-T, enquanto outros estão em estudos. Esta lista de serviços de gerência apresentada pelo ITU-T não é exaustiva, podendo ser ampliada quando novos serviços forem identificados. Um

novo serviço de gerência pode ser definido a partir da metodologia descrita na recomendação M.3020.

Uma função de gerência TMN é a menor parte de um serviço de gerência como percebido pelo usuário do serviço, e consistirá, geralmente, em uma seqüência de ações sobre um ou mais objetos gerenciados definidos. As funções de gerência são agrupadas em conjuntos de ia que é a menor parte reutilizável, e são classificadas em cinco Áreas Funcionais de Gerência: desempenho, falha, configuração, contabilização e segurança.

A Recomendação M.3400 apresenta um conjunto de funções de gerência para uso dos a. Os conjuntos de funções são agrupados e classificados pela área funcional de gerência que atendem e procuram refletir os aspectos gerais da rede de telecomunicações. A Tabela 3.4 apresenta uma síntese dos grupos funcionais de cada área.

Área Funcional	Grupos de Conjunto de Função
Gerência de Configuração	Engenharia e planejamento de rede
	Provisionamento
	Estado e controle
	Instalação
	Negociação e planejamento de serviço
Gerência de Falha ou Manutenção	Supervisão de alarmes
	Localização de falhas
	Teste
	Correção de falhas
	Administração de bilhete de anormalidade
Gerência de Desempenho	Monitoração de desempenho
	Controle de gerência de desempenho
	Análise de desempenho
Gerência de Contabilização	Funções de Faturamento
	Funções de Tarifação
	Funções de Contabilização
Gerência de Segurança	Administração de Segurança
	Auditoria
	Controle de Acesso

Tabela 4 - Área e Grupos Funcionais de Gerência

Um serviço de gerência pode ser provido por uma ou mais funções de gerência. Em geral, as ncia estão incorporadas por uma aplicação gerente ou por um sistema de operação OS. Os dados relacionados com uma função de gerenciamento são coletados através de aplicações agentes, através dos protocolos de gerência.

Uma função de gerência é na realidade uma seqüência de ações executadas por um agente sobre um ou mais objetos gerenciados. Um objeto gerenciado representa as propriedades de um recurso físico através de atributos e comportamentos. Uma função pode executar objetos gerenciados e receber notificações através de atributos que mostram alterações sofridas pelo mesmo (Schönberger, 1998) (Figura 3.17).

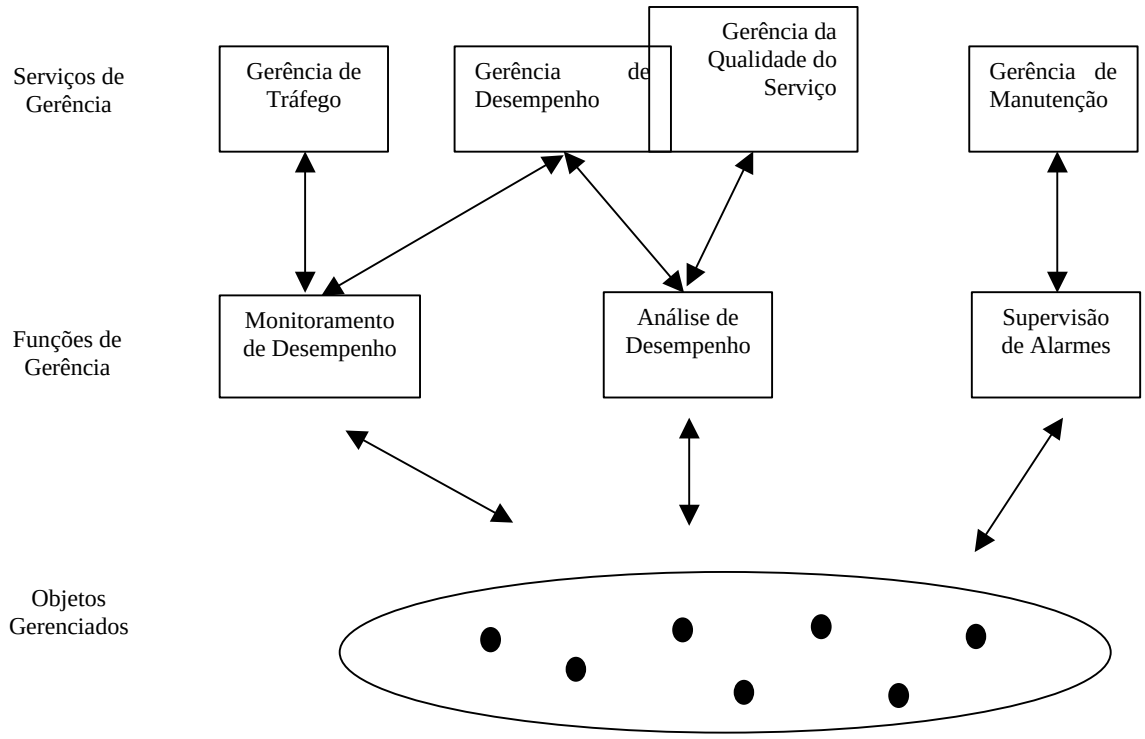


Figura 3.17 - Serviços e Funções de Gerência

3.9 CONCLUSÕES

No ambiente de telecomunicações, o termo TMN é usado amplamente para cobrir todos os tipos de soluções de gerenciamento de redes. Porém, de uma forma restrita, refere-se somente ao conjunto de soluções de gerenciamento de redes que satisfazem e estão em conformidade com os padrões de gerenciamento de rede definidos pelo ITU.

As recomendações TMN não especificam aspectos de implementação. Os padrões TMN existentes consideram que questões de implementação estão fora do escopo. Isto se deve ao fato de respeitar a independência de cada usuário. Entretanto, observa-se que dada à complexidade e à dificuldade de se implementar soluções TMN, faz-se necessário agregar novas perspectivas ao modelo a fim de permitir que este seja introduzido e se prolifere rapidamente em ambientes comerciais.

conhecimento tem como objetivo dar suporte ao processamento requerido para alcançar um objetivo, concluir alguma coisa, resolver um problema, selecionar uma solução, dentre outras.

No caso da abordagem de RBC, que faz uso do conhecimento baseado em experiências, o processamento requerido está voltado para encontrar casos similares, fazer adaptação e aprendizagem, sendo implementado através de um determinado método de raciocínio. A abordagem de representação do conhecimento e o método
da natureza do problema e da aplicação (ver item 2.2.1).

A partir do paradigma de RBC e baseado nos princípios da TMN, o modelo proposto descreve uma arquitetura global para agregar conhecimento baseado em experiências à Arquitetura TMN. Esta arquitetura global é orientada a objetos e incorpora dois elementos primários à

Arquitetura TMN, garantindo a compatibilidade e a padronização: *Arquitetura Funcional*; e *Arquitetura de Casos*.

A Arquitetura Funcional descreve a funcionalidade requerida para a (re)utilização do conhecimento baseado em experiências de gerenciamento. É representada por um bloco funcional de raciocínio, que provê as funções específicas relacionadas com os métodos de raciocínio. No contexto de RBC, é composto por funções que implementam os mecanismos de indexação, armazenamento, recuperação, adaptação e aprendizagem.

A Arquitetura de Casos descreve os elementos essenciais para a representação de casos de gerenciamento TMN, usando o guia para definição de objetos - GDMO e a notação abstrata ASN.1. Provê *templates* genéricos que consistem de palavras-chave para especificar componentes de um caso. Faz uso de uma abordagem estruturada para representação do conhecimento.

A partir do modelo proposto, sistemas abertos de conhecimento, baseados em experiências, podem ser desenvolvidos e utilizados nos ambientes de gerenciamento de redes e serviços de telecomunicações. O conceito de sistema aberto de conhecimento é introduzido neste trabalho e identifica as aplicações RBC que dão suporte às atividades de gerenciamento, fazendo uso do conhecimento baseado em experiências, formalizado por este modelo.

A ISO introduziu o conceito de sistema aberto de interconexão - OSI. Segundo a ISO, um sistema aberto refere-se a sistemas que estejam em conformidade com as suas recomendações para a comunicação com outros sistemas.

A ISO definiu o Modelo de Referência para Interconexão de Sistemas Abertos (RM-OSI - *Open System Interconnection Reference Model*) (ISO/IEC 7498) que constitui em uma série de recomendações para a interconexão de sistemas. Um sistema que adote estas *sistema aberto*. O fato de a interconexão ser aberta não implica no uso de nenhuma implementação específica, tecnologia ou modo de interconexão -se ao reconhecimento e suporte dos padrões ISO para intercâmbio de dados.

O RM-OSI não define ou fornece detalhes suficientes para a implementação precisa de serviços e protocolos de comunicação. O modelo fornece um esquema conceitual que permite que equipes de especialistas trabalhem de forma produtiva e independente no desenvolvimento de soluções.

Seguindo esta filosofia, a arquitetura global proposta por este modelo é genérica e como tal *sistemas abertos de conhecimento baseado em experiências.* Um sistema aberto de conhecimento baseado em experiências refere-se a sistemas que estejam em conformidade com este modelo proposto, ou seja, que estejam estruturados em termos de uma Arquitetura Funcional e de uma Arquitetura de Casos. Estes sistemas devem ainda fazer uso dos *templates* genéricos propostos para a representação de casos.

O modelo proposto não define como deve ser a implementação de um sistema aberto de conhecimento, mas fornece um esquema conceitual que permite que equipes de especialistas trabalhem de forma cooperativa e independente no desenvolvimento de bases de conhecimento.

O uso do guia GDMO permite que se conheça a semântica da representação do conhecimento. o a representação é interpretada. Para sistemas RBC, uma vez que se tem acesso à semântica, a representação do conhecimento é formal (Lenz, Bartsch-Spörl, Burkhard & Wess, 1998).

Para que dois sistemas abertos possam compartilhar conhecimento, cada sistema deve ser capaz de interpretar a representação do conhecimento do outro, ou seja, conhecer a semântica própria de cada um. Sendo adotada uma representação formal, isto é possível.

O compartilhamento de conhecimento entre sistemas abertos permite que bases de conhecimento sejam importadas e/ou exportadas entre diferentes plataformas, esta flexibilidade torna o usuário independente de um único fornecedor de soluções. Qualquer especialista pode criar uma base de conhecimento e esta ser importada e/ou exportada para diferentes ambientes de forma transparente e padronizada. Isto é viável a partir da utilização de uma representação formal para o conhecimento.

O modelo proposto é genérico, o que permite a sua aplicação de diversas maneiras e em Pode ser aplicado para domínios distintos dentro da TMN, tais como suporte à gerência de falhas, de contabilidade, de desempenho, de segurança, de configuração, de força de trabalho, etc., ou até mesmo para suporte a atividades de planejamento, desenvolvimento de aplicações de gerência, de modelagem de informações de gerência, dentre outras.

Sendo assim, o modelo proposto define *templates* genéricos para a representação do conhecimento. Isso significa que cada domínio possui particularidades específicas que devem ser tratadas individualmente. Não existe um *template* único para todos os domínios.

A flexibilidade do guia GDMO fornece diferentes construções que possibilitam representar as particularidades de cada domínio, sem ferir o modelo proposto. Os pacotes,
a serem representados, bem como o método de similaridade, estão diretamente relacionados com a natureza do domínio da aplicação RBC a ser construída.

Baseado nas observações prepostas, para incorporar a arquitetura global do modelo pr
Arquitetura TMN, foram descritas classes básicas[1] para dar suporte à Arquitetura de Casos e a Arquitetura Funcional:

- Classe *knowledge*: classe genérica usada para descrever o conhecimento incorporado ao ambiente de gerência TMN. É composta por uma representação do conhecimento e por

- Classe *representation*: classe genérica usada para descrever a base de conhecimento de um sistema.
- Classe *reasoning*: classe genérica usada para descrever o método de raciocínio usado para inferir sobre a base de conhecimento.

O relacionamento destas classes básicas, propostas com as classes padrões da Arquitetura de Informação TMN, é descrito através da hierarquia de herança e o diagrama de composição apresentadas na Figura 4.1 e 4.2 respectivamente.

Foi usado o diagrama de classes da notação UML (*Unified Modeling Language*) para expressar o relacionamento entre classes, visto que a hierarquia de nomeação da Arquitetura de Informação TMN (ver item 3.6), que é a hierarquia que representa a relação de
da TMN, é usada para descrever a composição de objetos gerenciados e não suas classes.

A utilização de nomeclatura em inglês visa manter a compatibilidade com as recomendações do ITU-T.

A Figura 4.1 apresenta a hierarquia de herança onde as três classes básicas propostas por este modelo derivam da classe *top* da Arquitetura TMN, apresentada no item 3.6. Este é o ponto de integração entre o modelo proposto e a Arquitetura TMN de forma transparente, padronizada e estruturada.

A Figura 4.2 descreve o relacionamento de composição do modelo proposto. A classe *knowledge* agrega as classes: *representation* e *reasoning*. A agregação indica que o relacionamento entre classes é do tipo *todo-parte*, ou seja, o conhecimento é expresso a partir de uma representação e de um método de raciocínio.

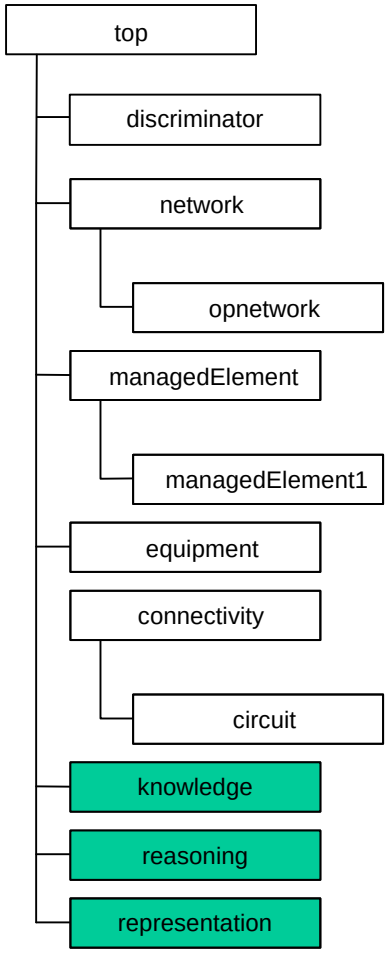


Figura 4.1 - Hierarquia de Herança TMN e o Modelo Proposto

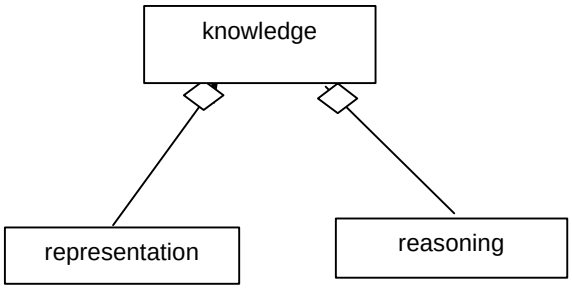


Figura 4.2 - Diagrama de Composição de Classes do Modelo Proposto

4.2 ARQUITETURA FUNCIONAL

A Arquitetura Funcional apresentada neste modelo é uma extensão da Arquitetura Funcional TMN. É descrita através de um bloco funcional de raciocínio RF (*Reasoning Function*), totalmente compatível com a Arquitetura Funcional TMN. O bloco funcional RF está sendo proposto neste modelo para dar suporte ao desenvolvimento de aplicações inteligentes e/ou sistemas de conhecimento na Arquitetura TMN.

Como qualquer outro bloco funcional (ver item 3.4.2), o bloco funcional RF provê funções genéricas que capacitam uma aplicação TMN a executar funções de raciocínio em sistemas de gerenciamento. É composto por diferentes componentes funcionais que provêem funções especificas que permitem o processamento sobre uma base de conhecimento. Os componentes funcionais podem ser do tipo RBC, sistemas especialistas, redes neurais, dentre outros.

O bloco funcional RF permite o desenvolvimento de aplicações inteligentes, tais como: agentes inteligentes, discriminadores de eventos inteligentes, tradutores e/ou adaptadores inteligentes. É possível integrá-lo a quaisquer um dos outros blocos funcionais TMN: Por exemplo:

- OSF: bloco funcional que está ligado às aplicações de suporte a operação, pode interagir com o bloco funcional RF para criar-se agentes inteligentes de suporte a operação;
- NEF: bloco funcional que representa os elementos de rede. O bloco funcional RF pode ser incluído nos elementos de rede para inserir conhecimento, criando elementos de rede gentes.
- WSF: bloco funcional que representa a funcionalidade das interfaces com os usuários de gerência. Interfaces mais inteligentes podem ser desenvolvidas, incluindo este bloco funcional.

- QAF e MF: blocos funcionais de adaptação e mediação respectivamente. O bloco RF permite o desenvolvimento de adaptadores, conversores, tradutores e mediadores inteligentes.

Conforme abordado, no item 3.7, uma função de gerência é uma seqüência de ações executadas por um agente sobre um ou mais objetos gerenciados. De acordo com este modelo, um agente de conhecimento RBC executará funções de raciocínio sobre casos de gerência contidos em uma base de conhecimento.

Na prática, algumas aplicações, interfaces, adaptadores, medidadores e elementos de rede de raciocínio e dispõem de alguma base de conhecimento, porém esta funcionalidade não é formalizada e nem descrita pelo modelo, o que permite a criação de soluções proprietárias. A importância da definição deste bloco funcional é destacável, pois representa formalmente a funcionalidade requerida por funções de raciocínio para o desenvolvimento de aplicações inteligentes.

A partir deste bloco funcional, são especificados componentes e funções inteligentes genéricos que uma vez inseridos nas recomendações TMN tornam-se padrões, facilitando cada vez mais o desenvolvimento de soluções integradas. As funções de raciocínio podem ser genéricas, porém a base de conhecimento é específica de cada aplicação.

No contexto deste trabalho, o bloco funcional RF é composto apenas pelo componente funcional do tipo RBC que descreve as funções pertinentes ao processo de raciocínio da abordagem RBC: indexação, armazenamento, recuperação, adaptação e aprendizagem. Estas funções poderiam ser descritas a partir de uma recomendação
3.7). A série de recomendações M.3400 descreve as funções de gerência TMN que são o menor componente reutilizável em um serviço de gerenciamento.

A nova recomendação, que descreveria[2] o processo de raciocínio RBC, poderia ser chamada de CBRF (*Case Base Reasoning Function*). Esta função descreveria o processo de raciocínio RBC. No caso, o modelo mais comum é o de Aamodt & Plaza (1994), que poderia ser definido como modelo genérico. As definições de modelos genéricos são usuajis nas recomendações do ITU-T (ver como exemplo: ISO/IEC 10164-9 – *Objects and Attributes for Access Control Function*). Em anexo, a nova recomendação CBRF poderia descrever componentes funcionais específicos, que implementassem determinados métodos de

o: *nearest-neighbour*, *ID3*, *k-nearest-neighbour*, dentre outros. Estes componentes uma vez implementados podem ser reutilizados em diferentes aplicações de suporte RBC.

A Figura 4.3 apresenta a modelagem da Arquitetura Funcional através da hierarquia de herança de classes. Diferentes componentes funcionais RBC poderiam ser definidos e descritos através de classes padronizadas e incorporadas à Arquitetura TMN.

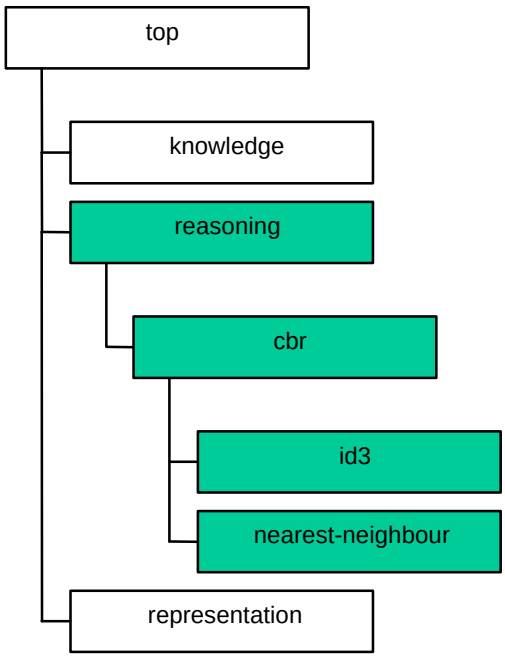


Figura 4.3 - Hierarquia de Classes da Arquitetura Funcional

[2] Vale a pena ressaltar o porque de estar se colocando estas afirmativas em um futuro condicional: deveria, descreveria, etc. Deve-se ao fato de que somente o ITU-T, que é organismo de padronização internacional, pode descrever e publicar recomendações TMN. Portanto, estas definições cabem como sugestões ao ITU-T.

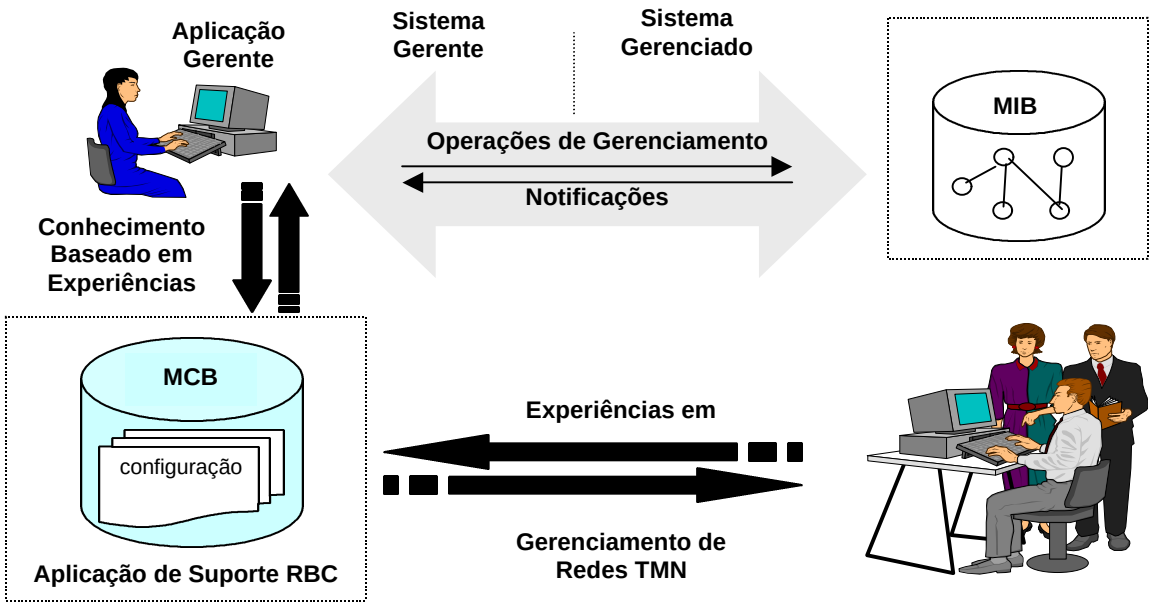
4.3 ARQUITETURA DE CASOS

A Arquitetura de Casos descreve a base de conhecimento de um sistema RBC. É o elemento essencial para se garantir a interoperabilidade entre os sistemas de suporte de gerenciamento e as diferentes plataformas de gerenciamento TMN. Sendo uma extensão da Arquitetura de ambém orientada a objetos. Logo, cada caso de gerência é visto como um objeto gerenciado.

De acordo com o modelo proposto e os princípios da Arquitetura de Informação TMN, um caso de gerência TMN é uma abstração para um conhecimento implícito, representando uma experiência útil em gerência TMN, é estruturado em termos de um objeto gerenciado, seus atributos, as operações de gerenciamento que podem ser executadas sobre ele e notificações que ele pode emitir.

Os mesmos princípios para nomeação de objetos gerenciados e seus atributos são também atribuídos aos casos de gerência TMN, de modo que eles podem ser identificados e acessados por protocolos de gerenciamento. Conceitos tais como classes de objetos gerenciados, herança, especialização, *containment* e alomorfismo também se aplicam.

Para destacar das informações de gerência do Modelo de Informação TMN, o conjunto de casos de gerenciamento são armazenados e mantidos em uma base de informações de casos, definida apenas a nível conceitual, denominada Base de - MBC (*Management Case Base*), que é agregada à Arquitetura TMN, conforme Figura 4.4. A base MCB pode ou não ser integrada em parte ou no todo à base MIB.



O conhecimento baseado em experiência pode ser representado por um ou mais casos, onde cada um dos casos representa um conhecimento distinto ou similar em relação a um determinado domínio. Os casos diferenciam-se através de suas propriedades associadas. Estas definem os atributos de um caso. Casos com experiências similares são agrupados por categorias de problemas, formando uma estrutura em árvore hierárquica de classes de casos, referenciada como Árvore de Casos de Gerência - MCT (*Management Case Tree*) (Figura 4.5).

Um caso de gerência é denominado uma instância de uma classe de casos e compartilha experiências similares com outros membros da mesma classe. Novas classes podem ser incluídas através do processo de especialização, permitindo um maior refinamento do conhecimento, tornando-o mais específico em um determinado tipo de problema (ex. classe configuração de equipamento ou classe de configuração de sistemas, ambas derivadas da classe configuração). Associado ao processo de -se também o conceito de herança.

A árvore MCT é uma subárvore da árvore de hierarquia de herança da Arquitetura de Informação TMN, apresentada no item 3.6. A árvore MCT começa com a classe *case* que é derivada da classe *representation* (proposta por este modelo), e, esta por sua vez, derivada da classe *top*. Todas as classes de casos de gerência são subclasses da classe *case*. Este é o ponto de agregação de conhecimento baseado em casos - experiências - à Arquitetura TMN de forma transparente, padronizada e estruturada.

Além de estruturar a modelagem dos casos de gerência, a árvore MCT também padroniza a forma primária de indexação, de armazenamento e de recuperação de casos, permitindo assim o compartilhamento de bases de casos sob diferentes plataformas e sistemas de gerência TMN.

Um objeto gerenciado pode estar contido dentro de outro objeto gerenciado, através de um relacionamento de *containment* - agregação. Um caso de gerenciamento também pode estar contido dentro de um outro caso de gerenciamento. Um caso de falha de uma central pode ter como parte da solução um caso específico de configuração da mesma central. Ou seja, a falha ocorria devido a problemas de configuração.

Um objeto gerenciado é dito alomórfico quando se comporta como um objeto de uma outra classe. Os casos de gerenciamento de uma determinada classe podem se comportar como se fossem instâncias de uma outra classe, visto que a experiência que representa pode ser útil a diferentes tipos de categorias de problema.

A base MCB pode ser acessada tanto remotamente através do protocolo de gerenciamento CMIP, como também pode ser acessada localmente sem fazer uso de protocolo. Operações sobre objetos gerenciados e sobre os atributos deles tais como *create, delete, action, get, set, event-report* também se aplicam, sendo possível tanto criar e eliminar casos de gerenciamento, como executar ações sobre eles.

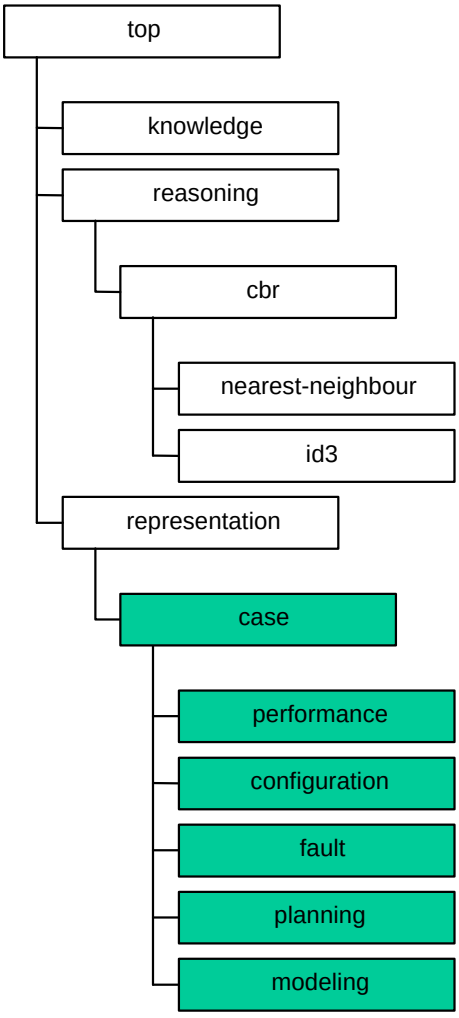


Figura 4.5 - Árvore de Hierarquia de Classes

A linguagem define os termos básicos usados para descrever o problema, a solução, o resultados e o contexto no qual o caso está inserido. Para garantir a compatibilidade e a integração com a Arquitetura TMN, adotou-se também o GDMO e seus templates para modelar e representar os casos de gerenciamento TMN (Ramos, Mota & Santos, 1998).

Um caso de gerenciamento TMN é um objeto gerenciado do tipo *case*, portanto os mesmos elementos adotados para descrever um objeto gerenciado são usados para descrever um caso de gerência TMN: classes, atributos, pacotes, comportamento, ações, operações, metas, objetivos, restrições, condições, etc.

4.4.1 TEMPLATE PARA CLASSES DE CASOS DE GERÊNCIA

Um objeto gerenciado do tipo *case* pode ser descrito a partir de um template genérico. O template genérico proposto provê um conjunto de construções e uma notação comum para representar vários aspectos de um caso de gerência TMN. Isto permite que classes de objetos gerenciados, do tipo *case*, definidas por um especialista, possam ser interpretadas e importadas para diferentes aplicações de suporte.

O template para classes de casos de gerência contém uma ou mais construções usadas para a definição de classe de objeto gerenciado. A Figura 4.6 apresenta o formato de sua estrutura e as construções básicas são descritas a seguir:

- MANAGEMENT OBJECT CLASS: inclui o nome da classe que pode ser usada para referenciar a classe por uma aplicação RBC.
- DERIVED FROM: identifica a superclasse da qual esta classe de caso é diretamente derivada.
- CHARACTERIZED BY: define as características do caso. São usadas para modelar as informações relevantes do caso. Esta construção permite a inclusão de um ou mais pacotes específicos. Estes pacotes são empregados para descrever os elementos de um caso, associados a um contexto, a uma descrição de um problema, aos resultados esperados, ao método de similaridade, dentre outros que sejam importantes para sua representação.
- REGISTERED AS: esta construção provê um identificador global único para a definição da classe.

```
<class-label>          MANAGEMENT OBJECT CLASS
[DERIVED FROM              <superclass-label>];
[CHARACTERIZED BY
                       [<package-case- label>                  PACKAGE];
                       [<package-context- label>       PACKAGE];
                       [<package-problem-label>        PACKAGE];
                       [<package-solution-label>       PACKAGE];
                       [<package-outcome-label>        PACKAGE];
                       [<package-similarity-label>     PACKAGE];
]
REGISTERED AS  <object identifier>;
```

Figura 4.6 - Template Genérico para definição de Classes de Casos

- Package-case: descreve os aspectos que caracterizam a base de casos, sendo utilizado para identificar o tipo de representação de casos adotada e quem é o desenvolvedor da base de casos.

- Package-context: descreve os aspectos relevantes de um caso, o contexto do qual foi capturado: o ambiente operacional, a empresa, a plataforma utilizada, etc;

- Package-problem: define qual o problema que foi tratado, incluindo qualquer outra informação que possa descrever sobre a situação para alcançar os objetivos: descrição do problema, o objeto do problema, os alarmes gerados, qual era a tarefa sendo executada, etc;

- Package-solution: provê a descrição da solução derivada e as justificativas para as ações executadas para resolver o problema;

- Package-outcome: especifica os resultados alcançados com a solução proposta;

- Package-similarity: especifica qual o método de raciocínio associado a esta base de conhecimento, quais são os índices relevantes, etc.

A Figura 4.7 apresenta o formato de um *template* genérico de um pacote, onde as construções básicas são descritas a seguir:

- PACKAGE TYPE: identifica o tipo do pacote (*package-context, package-problem, package-solution, package-outcome, package-adaptation, package-similarity*) ;

- BEHAVIOUR: construção que permite descrever o comportamento (semântica) associado ao pacote. Quando um objeto gerenciado é definido, a construção *behaviour* especifica como os atributos, operações e notificações se comportam. No caso dos objetos gerenciados do tipo caso, a construção behaviour permite descrever mais detalhes sobre o componente do caso, tipo:

- A semântica dos atributos, das ações, etc;

- Descrever detalhes das soluções apresentadas pelo caso;

- As circunstâncias sob as quais as ações foram executadas;

- Descrever as ações executadas para solucionar o problema;

- Descrever o processo de raciocínio aplicado na adaptação.

- ATTRIBUTES: construção que permite definir os atributos do pacote;

- ACTIONS: construção que identifica as ações automáticas aplicadas na solução e/ou na

- REGISTERED AS: construção que provê um identificador global único para identificar o pacote.

```
<package-label>              PACKAGE-TYPE

[BEHAVIOUR          <behaviour-definition-label>;]
[ATTRIBUTES         <attribute-label> [,<attribute-label>]*;]
[ACTIONS            <action-label> [,<action-label>]*;]

REGISTRED AS                 <object identifier>;
```

Figura 4.7 - Template genérico para a definição de pacotes

As construções ATTRIBUTES, ACTIONS e BEHAVIOUR são baseadas nas recomendações [X.720] e [X.722].

4.5 ESPECIFICAÇÃO DE CLASSES DE CASOS BÁSICAS DO MODELO PROPOSTO

Para definição de cada uma das classes propostas pelo modelo, deve-se utilizar os princípios e as ferramentas notacionais para a especificação das informações de gerenciamento, estabelecidas nas recomendações X.720 e X.722. De acordo com estas recomendações, o *template* genérico a ser utilizado na definição de classes é apresentado no item 3.7.1.

As classes *knowledge*, *representation* e *reasoning* são especificadas respectivamente nas Figuras 4.8, 4.9, 4.10, de acordo com este template genérico. Entretanto, a classe *cbr* (Figura 4.11), a classe *case* (Figura 4.12) e suas derivadas são especificadas de acordo com o *template* genérico de definição de casos de gerenciamento, proposto por este modelo no item 4.4.

```
Knowledge          MANAGED OBJECT CLASS

          DERIVED FROM  top;
          CHARACTERIZED BY     knowledgePackage        PACKAGE;
REGISTERED AS   knowledgeMObjectClass;
```

Figura 4.8 - Classe knowledge

```
                   representation   MANAGED OBJECT CLASS
          DERIVED FROM  top;
          CHARACTERIZED BY     representationPackage    PACKAGE;
REGISTERED AS   representationMObjectClass;
```

Figura 4.9 - Classe representation

```
reasoning          MANAGED OBJECT CLASS
        DERIVED FROM  top;
        CHARACTERIZED BY     reasoningPackage        PACKAGE;
```

Figura 4.10 - Classe reasoning

```
cbr                MANAGED OBJECT CLASS
        DERIVED FROM  reasoning;
        CHARACTERIZED BY     cbrReasoningPackage     PACKAGE;
```

Figura 4.11 - Classe cbrReasoning

```
case               MANAGED OBJECT CLASS
 DERIVED FROM   representation;
 CHARACTERIZED BY
                   CasePackage                PACKAGE;
                   caseContextPackage         PACKAGE;
                   caseProblemPackage         PACKAGE;
                   caseSolutionPackage        PACKAGE;
                   caseOutcomePackage         PACKAGE;
                   caseSimilarityPackage      PACKAGE;
REGISTERED AS  caseMObjectClass;
```

Figura 4.12 - Classe case

Conforme pode ser observado, cada uma das classes é caracterizada por um pacote, contendo as definições dos atributos, grupos de atributos, ações, comportamentos e notificações inerentes aos objetos gerenciados que serão instanciados. A especificação de cada um dos pacotes é apresentada no item seguinte.

4.5.1 ESPECIFICAÇÃO DOS PACOTES DE CLASSES DO MODELO PROPOSTO

A especificação dos pacotes segue o *template* definido na recomendação [X.722], conforme apresentado no item 3.7.1.3. A especificação dos pacotes utilizados na definição das classes propostas é apresentada nas Figuras 4.13, 4.14, 4.15, 4.16, 4.17, 4.18, 4.19, 4.20, 4.21 e 4.22.

```
KnowledgePackage                PACKAGE
        BEHAVIOUR knowledgePackageBehaviour;
        ATTRIBUTES
                KnowledgeId             GET,
                KnowledgeTitle                  GET,
                RepresentationId        GET,
                reasoningId                 GET;;;
REGISTERED AS  knowledgePackage1;
KnowledgePackageBehaviour       BEHAVIOUR
                                DEFINED AS
                !Este pacote é utilizado para identificar o tipo de conhecimento que está sendo
                        incorporado. Os tipos de representação e de raciocínio que caracterizam o
                        conhecimento.!;;
```

Figura 4.13 - Pacote knowledgePackage

```
representationPackage       PACKAGE
                BEHAVIOUR representationPackageBehaviour;
                ATTRIBUTES
                        RepresentationId                GET,
                        RepresentationTitle     GET,
                        RepresentationType      GET,
                        SupportReasoning                GET;;;
REGISTERED AS   representationPackage1;

RepresentationPackageBehaviour  BEHAVIOUR
                                DEFINED AS
                        !Este pacote é utilizado para identificar a representação de conhecimento que
                        está sendo incorporada e quais métodos de raciocínio suporta.!;;
```

Figura 4.14 - Pacote representationPackage

```
reasoningPackage  PACKAGE
                BEHAVIOUR reasoningPackageBehaviour;
                ATTRIBUTES
                        ReasoningId             GET,
                        ReasoningTitle                  GET,
                        ReasoningType                   GET,
                       SupportRepresentation    GET;;;
REGISTERED AS   reasoningPackage1;

reasoningPackageBehaviour       BEHAVIOUR
                                DEFINED AS
                        !Este pacote é utilizado para identificar o método de raciocínio e quais tipos
                        de representação de conhecimento suporta.!;;
```

Figura 4.15 - Pacote reasoningPackage

```
cbrReasoningPackage        PACKAGE
                BEHAVIOUR  cbrReasoningPackageBehaviour;
                ACTIONS         retrieveCase;
                                        reuseCase;
                                        reviseCase;
                                        retainCase;

REGISTERED AS     cbrReasoningPackage1;

cbrReasoningPackageBehaviour     BEHAVIOUR
                                DEFINED AS
                        !Este pacote é utilizado para identificar o método de raciocínio  RBC e Quais tipos de
                        representação de conhecimento suporta.!;;
```

Figura 4.16 - Pacote cbrReasoningPackage

```
casePackage         PACKAGE
                BEHAVIOUR  casePackageBehaviour;
                ATTRIBUTES
                        caseId                  GET,
                        caseTitle               GET,
                        caseType                GET,
                        caseDeveloper           GET;;;
REGISTERED AS     casePackage1;

casePackageBehaviour       BEHAVIOUR
                        DEFINED AS
                          !Este pacote é utilizado para identificar o tipo de representação de casos
                          adotada e quem é o desenvolvedor da base de casos.!;;
```

Figura 4.17 - Pacote casePackage

```
caseContextPackage          PACKAGE
                BEHAVIOUR   caseContextPackageBehaviour;
                ATTRIBUTES
                        caseContextId                   GET,
                        caseContextType                 GET;;;
REGISTERED AS   caseContextPackage1;

caseContextPackageBehaviour     BEHAVIOUR
                                DEFINED AS
                        !Este pacote é utilizado para identificar os aspectos relevantes de um caso, o contexto do
                        qual foi capturado: o ambiente operacional, a empresa, a plataforma utilizada, etc.!;;
```

Figura 4.18 - Pacote caseContextPackage

```
caseProblemPackage          PACKAGE
                BEHAVIOUR   caseProblemPackageBehaviour;
                ATTRIBUTES
                        caseProblemId                           GET,
                        caseProblemType             GET,
                        caseObjectProblem           GET,
                        caseCauseProblem            GET,
                        caseEffectProblem           GET;;;
REGISTERED AS   caseProblemPackage1;

caseProblemPackageBehaviour     BEHAVIOUR
                                DEFINED AS
                        !Este pacote é utilizado para descrever informações importantes sobre o
                        tipo do problema, causas do problema, conseqüências, sintomas e outros
                        comentários relevantes!;;
```

Figura 4.19 - Pacote caseProblemPackage

```
caseSolutionPackage         PACKAGE
                BEHAVIOUR caseSolutionPackageBehaviour;
                ATTRIBUTES
                        caseSolutionId                  GET,
                        caseSolutionType                GET;;;

REGISTERED AS     caseSolutionPackage1;

caseSolutionPackageBehaviour      BEHAVIOUR
                                  DEFINED AS
                  !Este pacote é utilizado para descrever justificativas para a solução adotada
                   e informações relevantes para explicar a solução aplicada!;;
```

Figura 4.20 - Pacote caseSolutionPackage

```
caseOutcomePackage        PACKAGE
                BEHAVIOUR caseOutcomePackageBehaviour;
                ATTRIBUTES
                        caseOutcomeAssessment       GET;;;
REGISTERED AS     caseOutcomePackage1;

caseOutcomePackageBehaviour     BEHAVIOUR
                                DEFINED AS
                    !Este pacote é utilizado para descrever os resultados da solução aplicada, se foi um
                    sucesso ou    não!;;
```

Figura 4.21 - Pacote caseOutcomePackage

```
caseSimilarityPackage        PACKAGE
                BEHAVIOUR caseSimilarityPackageBehaviour;
                ATTRIBUTES
                        caseSimilarityId                        GET,
                        caseSimilarityGoal                      GET
                        caseSimilarityRelevantIndexes           GET;;;
REGISTERED AS   caseSimilarityPackage1;

caseSimilarityPackageBehaviour     BEHAVIOUR
                                   DEFINED AS
                    !Este pacote é utilizado para descrever as informações que são relevantes para a
                    determinação de um valor de similaridade. A relevância está diretamente relacionada com
                    o objetivo a que esta se propõe!;;
```

Figura 4.22 - Pacote caseSimilarityPackage

4.5.2 ESPECIFICAÇÃO DOS TEMPLATES DE AÇÕES DO MODELO PROPOSTO

A especificação das ações, definidas pelas classes do modelo proposto, segue o *template* definido na recomendação [X.722]. As Figuras 4.23, 4.24, 4.25, 4.26 descrevem tais ações.

```
retrieveCase          ACTION
                BEHAVIOUR          retrieveCaseActionBehaviour;
                MODE CONFIRMED;
                      WITH INFORMATION SYNTAX OBJECT IDENTIFIER;
                      WITH REPLY SYNTAX OBJECT IDENTIFIER;
REGISTERED AS   retrieveCaseAction1;

retrieveCaseActionBehaviour          BEHAVIOUR
                                     DEFINED AS
                    !Este pacote é utilizado para descrever as informações que são relevantes para a
                    execução da ação de recuperação de casos. A recuperação de casos é implementada de
                    acordo com o método de similaridade!;;
```

Figura 4.23 - Ação retrieveCase

```
reuseCase           ACTION
                BEHAVIOUR       reuseCaseActionBehaviour;
REGISTERED AS   reuseCaseAction1;

reuseCaseActionBehaviour BEHAVIOUR
                            DEFINED AS
                    !Este pacote é utilizado para descrever a etapa de reutilização em um ciclo RBC!;;
```

Figura 4.24 - Ação reuseCase

```
reviseCase          ACTION
                BEHAVIOUR       reviiseCaseActionBehaviour;
REGISTERED AS   reviseCaseAction1;

reviseCaseActionBehaviour BEHAVIOUR
                            DEFINED AS
                    !Este pacote é utilizado para descrever a etapa de revisão em um ciclo RBC!;;
```

Figura 4.25 - Ação reviseCase

```
retainCase          ACTION
                BEHAVIOUR       retainCaseActionBehaviour;


REGISTERED AS   retainCaseAction1;

retainCaseActionBehaviour BEHAVIOUR
                            DEFINED AS
                    !Este pacote é utilizado para descrever a etapa de aprendizagem em um ciclo RBC!;;
```

Figura 4.26 - Ação retainCase

4.5.3 ESPECIFICAÇÃO DOS ATRIBUTOS DEFINIDOS PELO MODELO PROPOSTO

O modelo proposto introduz uma série de atributos ao modelo de informação para dar suporte às classes definidas para agregar conhecimento à Arquitetura TMN. A especificação de *templates* requer a definição da sintaxe que deverá ser usada para permitir a importação e/ou exportação dos valores de um atributo em uma base de conhecimento. Para isto são usadas definições de tipos ASN.1[X.208]. A especificação das definições ASN.1 e dos *templates* dos atributos encontram-se respectivamente nos itens seguintes: 4.5.3.1 e 4.5.3.2.

4.5.3.1 Definições ASN.1

```
ObjectIdentifier ::= OBJECT IDENTIFIER
AditionalInformation ::= CHARACTER STRING
GenericIdentifier ::= GraphicString
IdentifierSet ::= SET OF GraphicString
Peso ::= REAL {0 ..1}
Indexes ::= SET OF SEQUENCE {AttributeId, Peso}
```

4.5.3.2 Templates de Atributos

Apenas os templates de atributos do pacote *casePackage* são apresentados neste item a título de exemplo (Figura 4.27). A especificação de todos os atributos propostos pelo modelo encontram-se no Anexo.1.

```
caseId          ATTRIBUTE
                WITH  ATTRIBUTE SINTAX  ASN.1 Defined GenericIdentifier;
                MATCHES FOR EQUALITY;
                BEHAVIOUR           caseIdBehaviour BEHAVIOUR
                DEFINED AS
                "O valor deste atributo identifica a base de casos.";;
REGISTERED AS {representationAttribute6};

caseTitle       ATTRIBUTE
                WITH  ATTRIBUTE SINTAX  ASN.1 Defined GenericIdentifier;
                MATCHES FOR EQUALITY;;
                BEHAVIOUR           caseTitleBehaviour BEHAVIOUR
                DEFINED AS
                "O valor deste atributo identifica um rótulo para a base de casos.";;
REGISTERED AS {representationAttribute16;};

caseType        ATTRIBUTE
                WITH  ATTRIBUTE SINTAX  ASN.1 Defined GenericIdentifier;
                MATCHES FOR EQUALITY;;
                BEHAVIOUR           caseTypeBehaviour BEHAVIOUR
                DEFINED AS
                "O valor deste atributo identifica o tipo da base de casos.";;
REGISTERED AS {representationAttribute17};

caseDeveloper       ATTRIBUTE
                WITH  ATTRIBUTE SINTAX  ASN.1 Defined  AditionalInformation;
                MATCHES FOR EQUALITY;
                BEHAVIOUR                   caseDeveloperBehaviour BEHAVIOUR
                DEFINED AS
                "O valor deste atributo identifica o responsável pelo desenvolvimento da base de
casos";;
REGISTERED AS {representationAttribute4};
```

Figura 4.27 - Templates de atributos do pacote *casePackage*

4.6 CONCLUSÕES

Ë importante ressaltar que nenhuma construção nova foi acrescentada ao GDMO. Todos os *templates* definidos neste modelo, para suportarem a incorporação de conhecimento baseado em experiências práticas à Arquitetura TMN, foram especificados de acordo com as -T, portanto totalmente padronizados e genéricos.

As experiências práticas em gerenciamento de redes e serviços de telecomunicações agregam conhecimento útil às aplicações de gerência, portanto devem ser formalizadas e incluídas à Arquitetura TMN.

O uso de RBC deve-se a natureza desta abordagem que faz uso de conhecimento advindo de situações concretas e não de modelos abstratos. Porém, a abordagem RBC por si só não resolve o problema da complexidade e do aprendizado em um ambiente de gerenciamento TMN. Outros paradigmas de IA devem ser incorporados, em acordo com as recomendações padrão, à Arquitetura TMN, tais como: redes neurais, algoritmos genéticos, sistemas especialistas, lógica fuzzy, etc. A combinação de diferentes técnicas permite o desenvolvimento de soluções mais eficientes para ambientes complexos e dinâmicos tais

O modelo proposto é flexível e aberto o suficiente para incorporar diferentes paradigmas de IA. Dado a sua arquitetura orientada a objetos, permite que diferentes tipos de representação do conhecimento e métodos de raciocínio sejam modelados e combinados. Goonatilake & Khebbal (1995) afirmam ser a orientação a objetos o modelo natural para sistemas híbridos inteligentes.

pertinentes a determinadas bases de casos está fora do contexto deste trabalho. Identificadas as propriedades de uma base de casos, os *templates* propostos oferecem, variados e ricos, recursos para representar os elementos de um caso.

Uma vez criada e representada uma base de casos em conformidade com as recomendações do modelo proposto, esta pode ser integrada a diferentes sistemas abertos de conhecimento baseados em experiências. De acordo com a filosofia das recomendações OSI, ITU-T e do modelo proposto, a implementação de sistemas de conhecimento, que façam uso desta base de

casos, é aberta, pode se dar através de ferramentas comerciais RBC, ou através da criação de agentes específicos dentro de uma plataforma de gerência TMN, ou mesmo através do desenvolvimento de soluções proprietárias.

A seguir, é apresentada a formalização de uma base de casos, a título de exemplo da aplicação dos *templates* propostos em uma base de casos real, e são descritos possíveis cenários para o desenvolvimento de aplicações RBC, utilizando a base de casos formalizada.

5.1 EXEMPLO DA FORMALIZAÇÃO DE UMA BASE DE CASOS

A fim de demonstrar a utilização da representação de casos proposta pelo modelo, uma base de caso do tipo Boletins de Atividades, usualmente conhecida nas empresas operadoras de telecomunicações como BAs, foi modelada e é apresentada a seguir.

Os BAs são registros de atividades internas das operadoras. Em geral, podem ser de natureza preventiva, corretiva ou de serviços. Contêm grande quantidade de informações voltadas para monitoração e controle das atividades. Os BAs são boletins criados em função da ocorrência de problemas nas redes e serviços oferecidos pelas operadoras.

Os BAs registram desde informações comuns de controle: data da identificação da anormalidade, data de criação do BA, fonte do informante, qual o distrito, qual o equipamento, a prioridade, a matrícula de quem analisou o BA; como também informações relevantes do conhecimento corporativo da organização: sintoma, descrição da atividade e situação final do equipamento.

Dada a ocorrência de um problema, um BA é aberto e encaminhado à divisão técnica pertinente, onde um especialista é alocado para resolver o problema, identificado no BA *sintoma* e *descrição da atividade*. Quando o problema for sanado, então o especialista finaliza o BA, complementando o campo *descrição de atividade* e preenchendo o campo *SFE* (Situação Final do Equipamento) que descreve a solução implementada (Figura 5.1).

Boletim de Atividade	
Número do BA	199706230387
Distrito	O
Matrícula Funcionário	035661
Fonte Informante	T20 – Alarme de Telesupervisão
Valor do BA	04
Prioridade	20
Data Abertura	199706232221
Data Anormalidade	199706232221
Natureza	Corretiva
Horário da Abertura	199706240621
Término Previsto	199706240621
Sintoma	ALD – Alarmes Diversos
Estação	QBO MORM
N.º Acionamentos	02
Área Técnica	Transmissão
Descrição Atividade	TSP20 56 AL.017 FONTE RADIO UHF 1+1 / AL.018 RX RADIO UHF 1+1 / AL.019 TX RADIO UHF 1+1 /// DIRECAO SIGO MORAL.040, 60, 80 E 100 VAGOS. OBS.: AL'S PRESENTES PQ RADIO FOI ENVIADO P/ LABORATORIO.
Tipo Modelo	2999
SFE	CRL – COR. C/REC.LOCAL

Figura 5.1 - Boletim de Atendimento

A fim de que possam transformar-se em uma base de casos formal e serem compartilhados entre diferentes sistemas abertos de conhecimento, os BAs foram modelados, usando a arquitetura global proposta por este modelo, e estruturados em termos de uma Arquitetura Funcional e de uma Arquitetura de Casos.

Em termos funcionais, definiu-se qual método de raciocínio RBC correspondente, qual algoritmo de similaridade aplica-se a esta base de casos. Em termos de representação de casos, identificou-se nos BAs as características que são relevantes a fim de se estabelecer a similaridade entre casos, definindo os índices relevantes para o processo de raciocínio RBC. Uma vez definido estes parâmetros, faz-se uso dos *templates* genéricos para descrição formal da base de casos.

Os BAs foram modelados para serem utilizados por um componente funcional de recuperação do tipo *nearest-neighbour – exactly match,* onde a similaridade é determinada para cada atributo entre o novo caso e o caso recuperado. Esta medida pode ser multiplicada por um fator de relevância. Em seguida, é calculado o somatório da similaridade de todos os atributos. O caso recuperado mais similar com o novo caso será o que tiver o maior somatório. A similaridade empregada pode ser descrita através da equação apresentada na Figura 5.2.

Similaridade$(T,S) = \sum f(Ti,Si) * wi$

T - caso recuperado
S - novo caso
n - número de atributos de cada caso
i - individual atributo de 1 a n
f - função de similaridade para o atributo i no caso T e S
w - fator de relevância do atributo i

Figura 5.2 - Fórmula da Similaridade

A modelagem dos BAs resultou na criação das classes *ba* e *nearest-neighbor*, descritas no item 5.1.1 e 5.1.2 respectivamente. A Figura 5.3 descreve o relacionamento destas classes criadas, com as demais classes padrões, definidas por este modelo (ver item 4.1), onde a classe *nearest-neighbor* é uma subclasse da classe *reasoning* e a classe *ba* é uma subclasse da classe *case*.

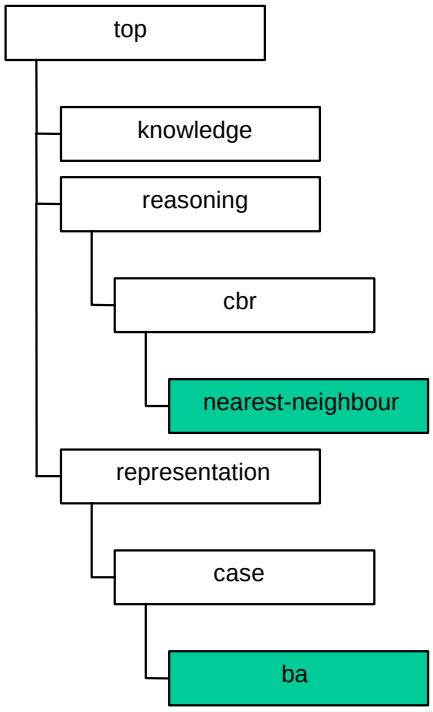


Figura 5.3 - Hierarquia de Herança – Modelagem de Boletins de Atividade

5.1.1 TEMPLATES DA CLASSE BA

```
ba              MANAGED OBJECT CLASS
 DERIVED FROM     representation;
 CHARACTERIZED BY
                 casePackage                 PACKAGE;
                 caseProblemPackage          PACKAGE;
                 caseSolutionPackage         PACKAGE;
                 caseSimilarityPackage       PACKAGE;
REGISTERED AS    boletimAtividadeCaseClass1;

casePackage PACKAGE
                 BEHAVIOUR          baCasePackageBehaviour;
                 ATTRIBUTES
                      caseId                 GET,
                      caseType               GET,
                      caseDeveloper          GET;;;
REGISTERED AS    casePackage1;

baCasePackageBehaviour  BEHAVIOUR
                                DEFINED AS
                      “A base de casos BA  contém 600 casos gerados na Operadora X, no período entre


baCaseProblemPackage    PACKAGE
                 ATTRIBUTES
                      baId                   GET,
                      baNatureza             GET,
                      baSintoma              GET,
                      baAreaTecnica                GET,
                      baTipoModelo           GET;;;
REGISTERED AS    baCaseProblemPackage1;

caseSolutionPackage       PACKAGE
                 ATTRIBUTES
                      baSFE                  GET,
                      baDescriçãoAtividade   GET;;;
REGISTERED AS    caseSolutionPackage1;




baCaseSimilarityPackage   PACKAGE
```

BEHAVIOUR baCaseSimilarityPackageBehaviour;

ATTRIBUTES

caseSimilarityId GET,

caseSimilarityRelevantIndexes GET;;;

REGISTERED AS baCaseSimilarityPackage1;

caseSimilarityPackageBehaviour BEHAVIOUR

DEFINED AS

"Esta base de casos foi modelada para suportar o método de raciocínio nearest-neighbor, do tipo exactly match, com cálculo de relevância para os índices definidos no atributo baCaseSimilarityRelevantIndexes.";;

5.1.2 DEFINIÇÕES ASN.1

Natureza ::= CHOICE OF {

[0] Corretiva

[1] Preventiva

[2] Serviços

}

Sintoma ::= CHOICE OF {

[0] Alarme canal comum

[1] Alarme juntores

[2] Alarme AXE

[3] Alarme MUX PDH

[4] Alarme rota

[5] Alarme roteador não responde

[6] Sem portadora

[7] Volta tom de discar

[8] Toca só uma vez e para

[9] Perda de tráfego maior 50%

[10] Alarme EWSD

[11] Completa sem ficha

}

```
AreaTecnica ::= CHOICE OF {
                                        [0] Transmissão
                                        [1] Comutação
                                        [2] Dados
                                        [3] Serviços
                                    }
sfe ::= CHOICE OF {
                                        [0] Defeito no Jump-DG
                                        [1] Roteador defeito geral
                                        [2] Cabo primário substituído
                                        [3] Modem inserindo erros
                                        [4] UDD/UDA com defeito
                                        [5] Cor c/ recup mant - laboratório
                                        [6] Cor c/ rec local
                                        [7] Cor c/ repos manual
                                        [8] Cor c/ repos remota
                                        [9] Sub Rack : posição c/ defeito geral}
```

5.1.3 DEFINIÇÕES DOS ATRIBUTOS

IMPORTS

AttributeId FROM CMIP {joint-iso-ccitt ms(9) cmip(1) modules(0) protocol(3)}
smi2AttributeID ObjectIdentifier, AditionalInformation, GenericIdentifier, IdentifierSet
{joint-iso-ccitt ms(9) smi(3) part2(2) attribute(7)}

```
baId                ATTRIBUTE
              WITH  ATTRIBUTE SINTAX  ASN.1 Defined  GenericIdentifier;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR           baIDBehaviour BEHAVIOUR
                          DEFINED AS
                     "O valor deste atributo identifica o boletim de atividade.";;
             REGISTERED AS {baAttribute1;};
```

```
baNatureza          ATTRIBUTE
                 WITH  ATTRIBUTE SINTAX  ASN.1 Defined  Natureza;
                     MATCHES FOR EQUALITY;
                     BEHAVIOUR              baNaturezaBehaviour BEHAVIOUR
                             DEFINED AS
                      “O valor deste atributo identifica a  natureza do BA.”;;
                REGISTERED AS {baAttribute2;};

baSintoma           ATTRIBUTE
                 WITH  ATTRIBUTE SINTAX  ASN.1 Defined  Sintoma;
                     MATCHES FOR EQUALITY;
                     BEHAVIOUR              baSintomaBehaviour BEHAVIOUR
                             DEFINED AS
                      “O valor deste atributo identifica o sintoma que gerou a abertura do BA.”;;
                REGISTERED AS {baAttribute3;};

baAreaTecnica       ATTRIBUTE
                 WITH  ATTRIBUTE SINTAX  ASN.1 Defined  AreaTecnica;
                     MATCHES FOR EQUALITY;
                     BEHAVIOUR              baAreaTecnicaBehaviour BEHAVIOUR
                             DEFINED AS
                      “O valor deste atributo identifica para qual área técnica foi encaminhado o BA.”;;
                REGISTERED AS {baAttribute4;};

baTipoModelo        ATTRIBUTE
                 WITH  ATTRIBUTE SINTAX  ASN.1 Defined  GenericIdentifier;
                     MATCHES FOR EQUALITY;
                     BEHAVIOUR              baTipoModeloBehaviour BEHAVIOUR
                             DEFINED AS
                      “O valor deste atributo identifica o tipo e o modelo do equipamento ao qual está associado o
                                ;
                REGISTERED AS {baAttribute5;};
```

baSFE ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined sfe;

MATCHES FOR EQUALITY;

BEHAVIOUR baSFEBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica a situação final do equipamento, qual após a atividade

REGISTERED AS {baAttribute6;};

baDescriçãoAtividade ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined AditionalInformation;

MATCHES FOR EQUALITY;

BEHAVIOUR baDescriçãoAtividadeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo descreve quais atividades foram executadas para reparar o sintoma

REGISTERED AS {representationAttribute7};

5.1.3.1 Templates da Classe Nearest-neighbor

Nearest-neighbor MANAGED OBJECT CLASS

DERIVED FROM cbr;

CHARACTERIZED BY cbrReasoningPackage PACKAGE;

REGISTERED AS nearest-neighborCbrReasoningClass1;

cbrReasoningPackage PACKAGE

BEHAVIOUR cbrReasoningPackageBehaviour;

ACTIONS nearest-neighborRetrieveCase;

REGISTERED AS cbrReasoningPackage1;

cbrReasoningPackageBehaviour BEHAVIOUR

DEFINED AS

"Este método de recuperação calcula a similaridade do tipo exactly match com índices de

nearest-neighborRetrieveCase ACTION

MODE CONFIRMED;

WITH INFORMATION SYNTAX OBJECT IDENTIFIER;

WITH REPLY SYNTAX OBJECT IDENTIFIER;

REGISTERED AS nearest-neighborRetrieveCase Action1;

5.2 INTEGRAÇÃO COM UMA FERRAMENTA COMERCIAL RBC

Uma base de casos descrita formalmente, através de *templates* GDMO propostos pelo modelo, pode ser importada para qualquer ferramenta RBC, comercial ou não, que suporte a importação de bases de casos. Para tanto, faz-se necessário que se tenha acesso aos *templates* que descrevem as classes de casos contidas na base de casos importada. Além disso, deve haver compatibilidade entre o componente funcional de similaridade oferecido pela ferramenta RBC com o componente funcional descrito através da arquitetura funcional da base de casos importada.

Quando uma empresa fabricante de equipamentos ou mesmo quando um profissional experiente resolver disponibilizar uma base de casos para agregar valor a um equipamento ou serviço de gerência, além de fornecer os casos propriamente ditos em arquivo digital, deve também fornecer os *templates* das classes, pacotes, atributos e a respectiva árvore de herança para que o cliente possa ter acesso não somente à sintaxe como também à semântica da base de casos, o que possibilita a importação de forma transparente.

Tomando como exemplo a ferramenta comercial CBR-Works - Release 4.1.10, a importação de bases de casos ocorre via utilitário ODBC (*Open DataBase Connectivity*) (Tecinno Gmbh. (1999). Após a conexão do CBR-Works com a base de casos via ODBC, o usuário deve criar novos tipos e conceitos, dentro da ferramenta CBR-Works, para suportar a sintaxe e a semântica específica da base de casos. A partir desta adequação, a base de casos pode ser importada e tratada como uma base própria da ferramenta CBR-Works (Figura 5.4).

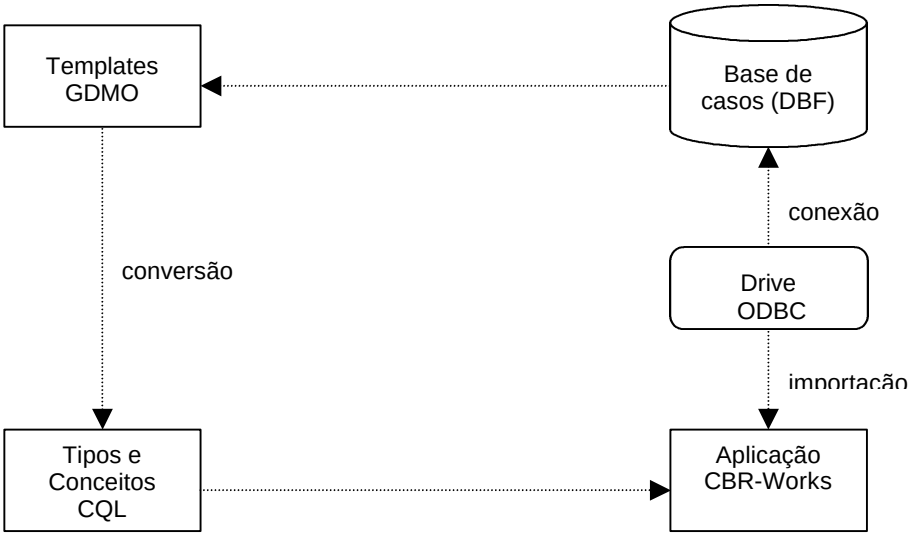


Figura 5.4 - Integração com CBR-Works

5.3 INTEGRAÇÃO COM UMA PLATAFORMA DE GERÊNCIA TMN

A integração, com uma plataforma de gerência TMN, é a mais transparente possível. Através de ferramentas específicas são capazes de compilar *templates* GDMO e gerar classes de objetos C^{++}.

Uma vez geradas as classes de objetos do tipo *case*, que representam as classes de uma base de casos, agentes de conhecimento RBC são criados da mesma forma como são criados quaisquer agentes nestas plataformas. A diferença é que est

controlam elementos e serviços de uma rede de telecomunicações, mas sim identificam similaridade entre casos de gerência.

Por exemplo, quando um objeto gerenciado, representando um elemento ou serviço de rede,

de alarme para uma aplicação agente, que se encarrega de repassá-las a uma aplicação gerente, através de um relatório de eventos. O gerente da rede, através desta aplicação de gerenciamento, monitora na console de operação da rede os alarmes enviados.

Dada a ocorrência de alguma anormalidade, a partir da aplicação de gerência, o gerente da rede pode emitir uma operação de gerenciamento, para o agente de conhecimento RBC, recuperar na base de casos experiências anteriores similares com a atual. Através de um relatório de eventos, o agente de conhecimento RBC identifica quais são os casos mais similares disponíveis na base. A partir deste relatório, o gerente pode adaptá-los ao atual contexto e executar a ação de controle pertinente (Figura 5.5).

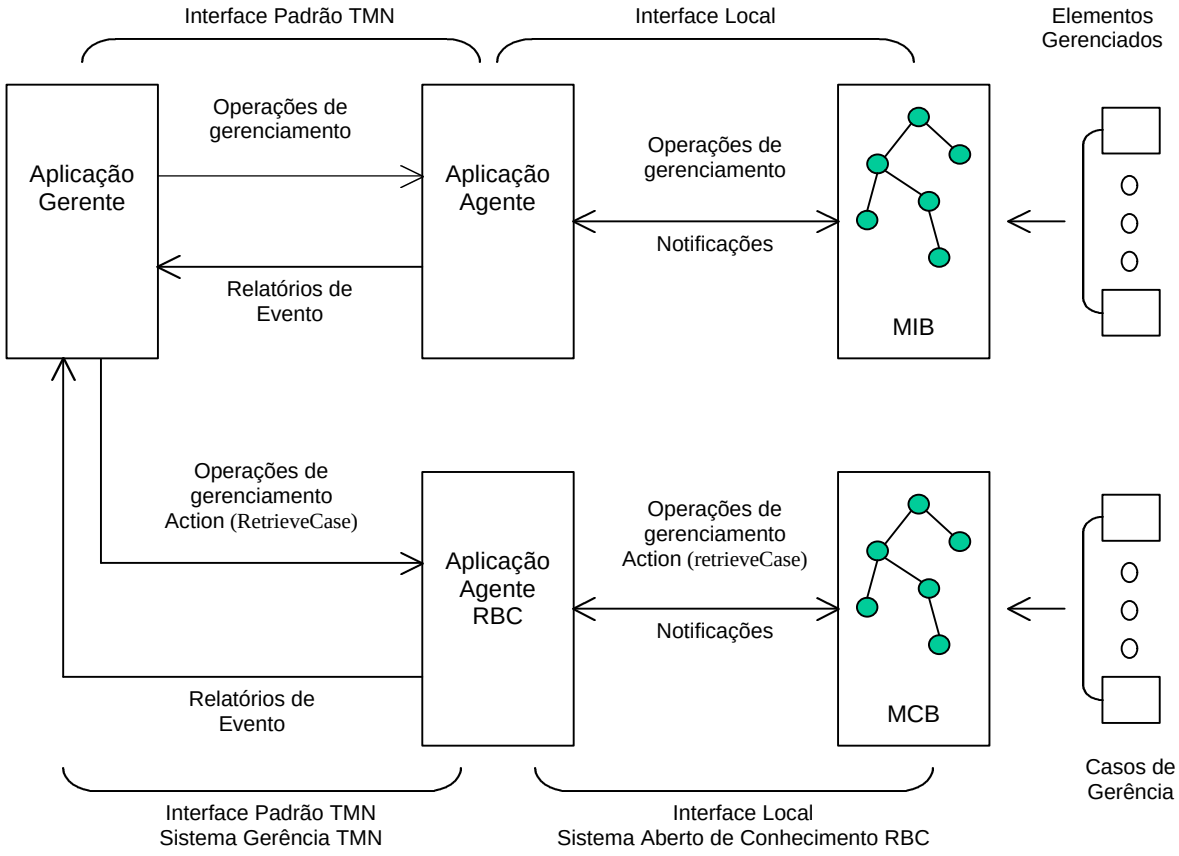


Figura 5.5 - Integração com Sistemas de Gerência TMN

O Anexo A, item A.1, apresenta a especificação de uma aplicação básica, através da linguagem UML (Eriksson & Penker, 1998), que descreve os requisitos funcionais da interação de um Agente RBC com os casos de gerência armazenados na base MCB.

5.4 CONSIDERAÇÕES SOBRE A APLICAÇÃO DO MODELO PROPOSTO

Diferentes tipos de aplicações RBC, desde as mais simples como também as mais complexas, podem ser desenvolvidas a partir do modelo proposto, devido à diversidade de construções que os *templates* GDMO oferecem.

Recentes aplicações da abordagem RBC descrevem uma tendência para arquiteturas abertas e flexíveis. Jaczynsky & Trousse (1998) definem uma arquitetura aberta, orientada a objetos, para facilitar o desenvolvimento de aplicações RBC, principalmente pela reutilização de projetos e implementações anteriores. Classes para métodos de raciocínio, indexação, adaptação são especificadas. Não tratam a respeito da representação dos casos propriamente dita, não fazem uso de nenhuma linguagem formal para a especificação dos elementos de um caso.

Neste contexto, o modelo proposto também acompanha esta linha e incorpora elementos bastante flexíveis. Estruturado sob o paradigma de orientação a objetos, conceitos como abstração, encapsulamento, herança, troca de mensagens, alormofismo, etc. são naturalmente oferecidos pelos *templates* propostos. Suporta totalmente a arquitetura proposta por Jaczynsky & Trousse e estende através do emprego de uma linguagem formal para especificação dos elementos de um caso, bem como para a especificação de diferentes métodos de raciocínio.

Gresse (2000) apresenta uma arquitetura para reutilização de experiências em mensuração de software, utilizando um mecanismo de recuperação orientado a objetivos que suporta o dinâmica e flexível, em função do objetivo específico da busca por casos similares.

Em termos de métodos de raciocínio, foi representado a título de exemplo, no capítulo anterior, um método de recuperação de similaridade bem simples, porém os *templates* GDMO dispõem de construções genéricas que permitem a utilização de mecanismos mais aprimorados.

Tendo como exemplo a base de casos Boletim de Atividade, definida no item 5.1, suponha que um sistema aberto de conhecimento de uma operadora de telecomunicações tenha dois perfis de usuários distintos: gerência de serviços e gerência de operação. Para a gerência de serviços os índices relevantes são os atributos *baSintoma, baNatureza* e *baAreaTecnica*. Para a gerência de operação os atributos *baSintoma* e *baTipoModelo* são de maior peso.

Os *templates* GDMO oferecem a construção do tipo pacote condicional, que incorpora bastante flexibilidade a um modelo de informação. Um pacote condicional é um pacote comum, porém só está presente em todas as instâncias de um objeto se determinadas condições associadas forem satisfeitas.

A Figura 5.6 apresenta a utilização dos pacotes condicionais e mostra toda a flexibilidade que este recurso oferece a um especialista na modelagem de uma base de casos de gerenciamento. As construções básicas dos pacotes condicionais incluem:

- CONDITIONAL PACKAGE: permite a inclusão de um ou mais pacotes específicos, para descrever elementos de um caso. Estes pacotes estão associados a uma condição que deve ser satisfeita para que sejam instanciados em todos os objetos da classe.
- PRESENT IF: identifica as condições que devem ser satisfeitas a fim de que um pacote seja instanciado.

Na Figura 5.6, a construção CONDITIONAL PACKAGE identificou três pacotes condicionais que serão instanciados de acordo com o objetivo da recuperação que está sendo feita. Se a recuperação for para atender aos objetivos da gerência de serviços, o pacote condicional *baCaseSimilarityPackage1* é instanciado na criação de todos os objetos gerenciados desta classe. Caso o objetivo seja atender a gerência de operação, o pacote *baCaseSimilarityPackage2* será utilizado. E finalmente, quando nenhum objetivo específico for estabelecido, o *baCaseSimilarityPackage* será então usado, como *default*, pelo método de raciocínio para estabelecer a similaridade entre os casos.

A utilização de pacotes condicionais pode requerer implementações mais complexas de mecanismos de recuperação, entretanto não menos eficientes, fazendo uso de instanciação dinâmica de objetos. Face ao objetivo do novo caso, as inst
criadas dinamicamente com o pacote de similaridade pertinente.

O Modelo de Informação TMN especifica uma classe de objetos gerenciados do tipo *discriminator* que definem critérios de controle de serviços (X.721 | ISO/IEC 10165-2). Associado ao emprego de recursos dinâmicos, objetos do tipo *discriminator* podem ser muito úteis na identificação dos objetivos de um novo caso, como também para selecionar quais classes de problemas podem ser similares à categoria do novo caso.

Sendo assim, uma nova classe poderia ser criada, chamada *caseDiscriminator*, derivada da classe *discriminator*, que seria a responsável por identificar, dentre as várias classes de casos existentes na base de casos, quais as classes, que teriam instâncias criadas dinamicamente, com seus respectivos pacotes condicionais, para serem comparadas com o novo caso.

```
REGISTERED AS     boletimAtividadeCaseClass1;

baCaseSimilarityPackage    PACKAGE
                BEHAVIOUR  baCaseSimilarityPackageBehaviour;
                ATTRIBUTES
                        baSimilarityGoal                GET,
                        baRelevantIndexes               GET;;;
REGISTERED AS     baCaseSimilarityPackage;
baCaseSimilarityPackage1  PACKAGE
                BEHAVIOUR  baCaseSimilarityPackage1Behaviour;
                ATTRIBUTES
                        baSimilarityGoal                GET,
                        gsRelevantIndexes               GET;;;
REGISTERED AS     baCaseSimilarityPackage1;
baCaseSimilarityPackage2  PACKAGE
                BEHAVIOUR  baCaseSimilarityPackage2Behaviour;
                ATTRIBUTES
                        baSimilarityGoal                GET,
                        grRelevantIndexes               GET;;;
REGISTERED AS     baCaseSimilarityPackage2;
```

Figura 5.6 - Uso de pacotes condicionais

Dado um novo caso, o agente de conhecimento RBC é invocado, pela aplicação gerente, para identificar na base de casos quais os casos mais similares com o atual. Este agente de conhecimento RBC invoca uma instância de objeto gerenciado da classe *caseDiscriminator*.

Esta instância, baseada em um conjunto de regras pré-estabelecidas, identifica quais classes de casos devem ter instâncias criadas para ser estabelecido o grau de similaridade com o novo caso. (Anexo A, item A.2)

A utilização de objetos discriminadores de casos é interessante quando se adota métodos de recuperação orientados a objetivos e quando se tem uma base de casos muito grande para se empregar mecanismos de poda na base, para se acelerar a recuperação. Os objetos discriminadores de casos selecionam somente as classes de casos que são relevantes para uma

O alomorfismo é um outro recurso do Modelo de Informação TMN que pode agregar muita flexibilidade a modelagem de casos de gerência. Como observado no item 4.3, um caso de uma classe pode apresentar soluções para categorias de problemas de outra classe, exibindo

rfico.

A árvore de herança MCT padroniza a forma primária de indexação da base de casos. Uma vez que um caso pertence a uma determinada classe, ele é acessado a partir daquela classe. Entretanto como visto anteriormente, pode se fazer necessário que este caso seja acessado a partir de outras classes arbitrárias. Uma classe pode ser alomórfica a uma ou mais classes. O atributo *allomorphsClassesDefault*, herdado da superclasse *top*, identifica as classes que têm comportamento alomórfico.

Por exemplo, suponha, que além da base de casos Boletim Atividade, se tivesse disponível uma outra base de casos, com experiências práticas para centrais de comutação do tipo AXE, formalizada com templates GDMO. Esta base de casos AXE contém um conjunto de casos

específicos a centrais do fabricante. Ao modelar esta base, foi criada uma subclasse *axe*, derivada da classe case, conforme Figura 5.7.

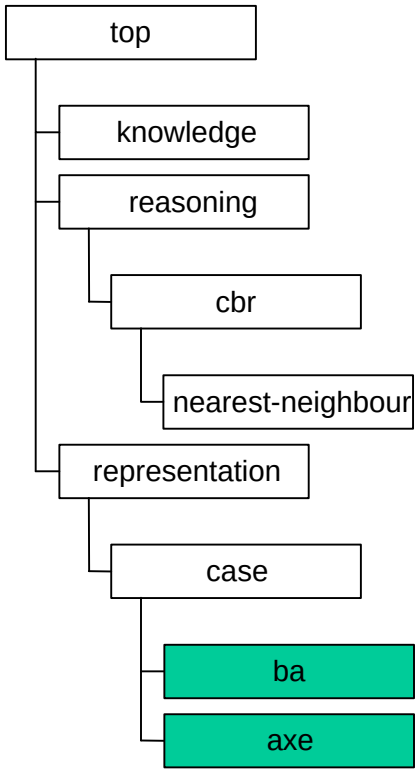


Figura 5.7 - Árvore de Herança – Estudo de Caso AXE

Entretanto, ao se analisar a base de casos BA observa-se que existem experiências úteis na resolução de problemas com centrais do tipo AXE. Logo a classe *ba* exibe um comportamento alomórfico a classe *axe*. Para representar este comportamento, o *template casePackage*, da definição da classe *ba*, incluiria o atributo *allomorphsClassesDefault* que teria associado o identificador da classe *axe* como valor *default* (Figura 5.8).

Ao objeto *caseDiscriminator* caberá a responsabilidade de garantir este relacionamento, identificando quais são as classes alomórficas de uma determinada classe base. Uma vez identificado este relacionamento, serão instanciados na base MCB objetos do tipo *case*, de

diferentes classes, porém que exibem um comportamento alomórfico. Neste caso, o contexto de uma classe se aplica a outra.

```
casePackagePACKAGE
        BEHAVIOUR       baCasePackageBehaviour;
        ATTRIBUTES
            caseId              GET,
            caseType                    GET,
            caseDeveloper               GET
            allomorphsClassesDefault    REPLACE WITH DEFAULT
            DEFAULT_VALUE,
            Attribute-Module.defaultallomorphsClasses    GET;;;

REGISTERED AS   casePackage1:
```

Figura 5.8 - Definição de alomorfismo entre a classe *ba* e a classe *axe*

Specialski (2000) definiu um modelo que introduz um conjunto de classes de objetos gerenciados para representar o grau de relacionamento e dependência, entre os objetos gerenciados, na Arquitetura TMN. Ao invés de se usar o recurso de alomorfismo, este modelo pode ser aplicado perfeitamente à modelagem de casos de gerência, para que de forma abstrata expresse o relacionamento entre eles.

Dado uma aplicação e/ou um objetivo, as instâncias de caso de uma classe podem exibir diferentes comportamentos, diferentes graus de relacionamento com outras classes. Esta variação de comportamento define quais pacotes condicionais devem ser instanciados e quais classes de objetos gerenciados devem ter instâncias criadas na base MCB.

O relacionamento entre classes de casos pode ser representado através de uma instância de um objeto gerenciado da classe *relacionamentoFuncional*, definido por Specialski (2000), e não

Face a estes números, muitas empresas têm despertado para a importância de se coletar, compartilhar e formalizar o conhecimento de seus empregados, parceiros e clientes, a fim de melhorar a qualidade dos serviços oferecidos.

Programas específicos têm sido desenvolvidos no sentido de criar-se um centro eletrônico de conhecimento, introduzindo o conceito de *eknowledge*. Em 2003, os investimentos em gerenciamento do conhecimento serão por volta de US$ 8 (oito) bilhões, enquanto que em 1999 foram de US$1,3 bilhão (IDC, 2000).

Tem sido muito enfatizado o gerenciamento do conhecimento na organização como: gerenciamento conhecimento do negócio ERP (*Enterprise Resource Planning*); gerenciamento do conhecimento do cliente – CRM (*Customer Relationship Management*). Aha (1999) destaca a sinergia KM/RBC para outros tipos de tarefas.

Quase nada se tem discutido em termos da aplicação de KM à gerência de redes e serviços de telecomunicações, exceto CRM e ERP. Analisando-se as seções técnicas dos principais congressos e revistas internacionais e nacionais, na área de gerências de redes e serviços de telecomunicações, entre os anos de 1995 e 1999, inclusive as chamadas para os eventos do ano 2000, não existem referências diretas à área de gestão do conhecimento:

- IM - Integrated Management;
- NOMS - Network Operations Managament Systems;
- APNOMS - Asia-Pacific Network Operations Managament Systems;
- LANONS - Latin-American Network Operations Managament Systems;
- JNSM - Journal of Network and Systems Management;

- SBRC - Simpósio Brasileiro de Redes de Computadores;
- WTMN - Workshop Nacional de TMN.

Neste sentido, este modelo pode ser o ponto inicial para a discussão do desenvolvimento de aplicações de gestão do conhecimento no ambiente TMN, de uma forma mais ampla para formação de recursos humanos, para armazenamento de conhecimento e para suporte a resolução de problemas (Aha, 1999).

CAPÍTULO - 6 CONSIDERAÇÕES FINAIS

6.1 AVALIAÇÃO DOS RESULTADOS

O desenvolvimento desta tese de doutorado culminou com alguns importantes resultados: a elaboração de um modelo aberto e genérico que estende a Arquitetura TMN, em acordo com -T, para incorporar conhecimento baseado em experiências em gerenciamento de redes e serviços de telecomunicações; a utilização de uma linguagem formal para a descrição de bases de casos de gerenciamento; a especificação de um modelo flexível e dinâmico, orientado a objetos, para o desenvolvimento de aplicações RBC.

Em geral, apenas informações têm sido tratadas pelos modelos de informação TMN. O modelo proposto incorpora elementos para agregar conhecimento, baseado em experiências de natureza distinta, útil ao gerenciamento de redes e serviços de telecomunicações, em acordo com as recomendações padrão ITU-T. Este tipo de conhecimento é fundamental para sistemas complexos, como é o caso das aplicações em um ambiente de gerenciamento TMN, melhorando a qualidade das atividades de gerenciamento como um todo.

A padronização serve como referência para o desenvolvimento de soluções abertas de conhecimento no mercado, evitando a dependência de fornecedores e de soluções específicas, garantindo assim os investimentos já realizados. A formalização do conhecimento permite o compartilhamento e a democratização do saber fazer gerência, de forma transparente entre diferentes sistemas abertos de conhecimento.

Tanto para clientes como fornecedores de soluções e/ou produtos de gerenciamento TMN, a socialização do saber fazer é fundamental para agregar valor as suas atividades. O fornecedor pode oferecer um produto diferenciado no mercado, que incorpora experiências práticas em -lo de maneira produtiva, permitindo o retorno esperado.

Por outro lado, o cliente também pode disponibilizar as experiências práticas, adquiridas em seu ambiente TMN para o fornecedor, incrementando assim a base de casos do produto em questão e servindo como moeda de troca em outros serviços. À medida que a base de casos de um produto torna-se mais refinada, mais valor agregado tem este produto, o que leva ao aprimoramento constante das ações de gerenciamento, acelerando o aprendizado e a formação de recursos humanos. Esse compartilhamento de experiências práticas só é possível a partir das propriedades do modelo proposto: genérico; aberto; e padronizado.

Contudo, apesar de se tratar de um modelo bastante poderoso e flexível, a sua aplicação requer, do projetista de sistemas abertos de conhecimento, sólidos conhecimentos sobre a abordagem RBC e da Arquitetura TMN; sobre o gerenciamento de redes em ambientes TMN; e sobre plataformas de gerenciamento TMN para o desenvolvimento de soluções integradas.

Em um outro contexto, se o mesmo for utilizado para o desenvolvimento de bases de casos, o projetista deve incorporar também habilidades para realizar a aquisição do conhecimento. dade de conhecimentos requeridos, faz-se necessário uma equipe técnica multidisciplinar para a utilização do modelo proposto.

Em um primeiro instante, mesmo tendo uma equipe com o conhecimento técnico requerido, as soluções geradas podem não ser as melhores possíveis, devido às particularidades da abordagem RBC e da Arquitetura TMN, bem como do domínio da aplicação. Entretanto, vale

ressaltar que o modelo proposto introduz uma nova cultura que é a utilização de conhecimento, advindo de experiências práticas, no ambiente de gerenciamento TMN, de maneira padronizada e formal. E, em conseqüência, faz-se necessário uma maior exploração e maturação do modelo proposto a fim de se alcançar o desenvolvimento de soluções com

Neste sentido, baseado no modelo proposto, foi descrito um exemplo básico de criação de uma base de casos e discutidas diferentes abordagens de desenvolvimento de sistemas abertos de conhecimento RBC. O exemplo apresentado formaliza uma base de casos real, segundo as recomendações do modelo proposto.

Em seguida, duas alternativas foram apresentadas para a utilização uma base de casos formal: através de uma ferramenta comercial RBC (item 5.2) e através da uma plataforma de gerência TMN (item 5.3). A primeira alternativa mostra que bases de conhecimentos formais podem ser trocadas livremente entre diferentes sistemas de conhecimento. A segunda alternativa mostra que o modelo está de acordo com as recomendações da Arquitetura TMN e pode ser integrado de forma transparente a este ambiente.

Em termos de modelagem, foram consideradas duas abordagens: uma abordagem mais simples que requer uma implementação estática, como foi a utilizada para a base de casos *Boletins de Atividades*, apresentada no item 5.1; e a outra uma abordagem dinâmica que requer uma implementação mais aprimorada e complexa, onde os objetos, que representam os casos de gerenciamento, são criados de acordo com os objetivos da recuperação (ver item 5.4). Ambas abordagens comprovam a flexibilidade do modelo proposto.

Em nível de implementação prática, a primeira alternativa (item 5.2) foi testada, a partir da importação da base de casos, especificada formalmente para uma plataforma comercial RBC, chamada CBR-Works. Esta abordagem fomentou uma dissertação de mest
sendo desenvolvida e será defendida ainda este ano, na Pós-graduação em Ciência da Computação da UFSC. A segunda alternativa (item 5.3) será implementada também através de uma outra dissertação de mestrado a ser orientada brevemente.

Analisando a flexibilidade do modelo proposto, tendo sua arquitetura global orientada a objetos, permite que diferentes tipos de classes sejam criados, especializando a forma de representação e dos métodos de raciocínio. Nada impede que se criem classes de objetos gerenciados, derivadas da classe *representation* e da classe *reasoning* que formalizem a representação do conhecimento e o método de raciocínio de diferentes técnicas de inteligência artificial.

Udupa (1999) ressalta que ainda não existem padrões para aplicações inteligentes na Arquitetura TMN. Neste contexto, o resultado desta tese é um modelo, em acordo com as recomendações TMN, ou seja: padrão; que permite o desenvolvimento de diferentes aplicações inteligentes. Este trabalho introduz conceitos significativos, neste sentido, tais como: bloco funcional de raciocínio, *classes knowledge, representation e reasoning*, sistemas abertos de conhecimento, objetos discriminadores de conhecimento, função CBRF, dentre outros.

Os sistemas de gerência de redes têm utilizado basicamente de sistemas baseados em regras para correlacionar alarmes. Todavia nestas aplicações, os agentes de conhecimento trabalham de forma isolada, tendo pouca eficiência em um ambiente tão complexo e dinâmico como de rede de computadores e as experiências são coadjuvantes na representação do conhecimento.

Na busca por soluções mais eficientes, o modelo proposto permite comunicação entre objetos.

-agentes podem ser desenvolvidos, em acordo com o padrão TMN. O ambiente de gerenciamento é visto como uma sociedade de agentes que interagem entre si, de forma cooperativa em busca de um objetivo comum: cumprir a política definida para a rede e/ou a política definida para os serviços de gerência.

Uma solução de gerência, combinando diferentes técnicas de Inteligência Artificial, pode aumentar significativamente a capacidade de raciocínio de qualquer sistema, ao executar determinada tarefa, obtendo melhores resultados. Sendo assim, o modelo proposto suporta também o desenvolvimento de sistemas híbridos, onde é empregada mais de uma técnica computacional inteligente.

Ramos, Alves & Specialski (1997) destacam que o uso de agentes híbridos inteligentes, em acordo com as recomendações TMN, é a base para a evolução do gerenciamento de redes e a saída para se atender tantas exigências e as constantes mudanças e necessidades emergentes deste ambiente tão complexo. A flexibilidade do modelo proposto vem de encontro a estas exigências e serve de base para o surgimento de uma nova ger

TMN com recursos para o desenvolvimento de sistemas abertos de conhecimento, multi-agentes híbrido, com suporte a tomada de decisão.

6.2 CONTRIBUIÇÕES DO TRABALHO

Dentro do escopo desta tese e dos resultados descritos anteriormente, pode-se identificar contribuições de três naturezas: a primeira de caráter exclusivamente investigativo, envolvendo pesquisa bibliográfica, aprofundamento de conhecimento e elaboração de textos explicativos sobre a abordagem RBC e a Arquitetura TMN, e identificação de necessidades

- Um modelo para incorporar conhecimento baseado em experiências à Arquitetura TMN;
- Integração da abordagem RBC à Arquitetura TMN;
- A utilização de uma representação formal para especificar casos de gerência TMN;
- A criação de templates GDMO para a representação de casos de gerência TMN;
- A introdução do conceito de sistemas abertos de conhecimento;
- A definição de recursos para a Arquitetura TMN orientar o desenvolvimento de aplicações inteligentes, tais como: bloco funcional, componentes e funções de raciocínio; e classes de icas para a modelagem de aplicações inteligentes.

Vale ressaltar que indiretamente, visto que este não era o objetivo inicial desta tese, criou-se um modelo, com uma maior abrangência, que permite incorporar conhecimento genérico à baseado em experiências práticas.

A produção científica foi estimulada no decorrer do desenvolvimento desta tese, culminando na publicação em simpósios nacionais e internacionais de artigos: Ramos (1998); Ramos(1999a); e Ramos(1999b); e fomentou também o desenvolvimento de uma dissertação de mestrado, onde uma base de casos, descrita formalmente através de templates GDMO, está sendo integrada a uma ferramenta comercial RBC.

6.3 TRABALHOS FUTUROS

Os trabalhos desenvolvidos dentro do contexto desta tese, em particular a modelo proposto, oferecem subsídios ao desenvolvimento de outros projetos de pesquisa nesta universidade e em outras instituições com interesse na área.

De imediato, identifica-se duas dissertações de mestrado e uma tese de doutorado podem ser trabalhadas e desenvolvidos bons trabalhos com repercussão internacional. Uma dissertação de mestrado para desenvolver agentes de conhecimento RBC em plataformas de gerência TMN; a outra dissertação de mestrado para implementar agências híbridas de gerenciamento de rede, com base em Ramos, Alves & Specialski (1997). E ainda, uma tese de doutorado para trabalhar quais elementos de um caso de gerenciamento TMN são relevantes para o aspecto de similaridade e definir uma metodologia para aquisição de conhecimento em um ambiente de gerenciamento TMN.

Finalizando, deve-se ainda desenvolver estudos para a aplicação do modelo proposto em outros domínios, que não o do ambiente TMN, para o desenvolvimento de ferramentas e

REFERÊNCIAS BIBLIOGRÁFICAS

Aamodt, A. e Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, e System Approaches. Artificial Intelligence Communications, 7 (1), 39-59.

Adams, E.K. & Willetts, K.J. (1996). The lean communications provider: surviving the shakeout through service management excellence. McGraw-Hill Companies, New York.

Aidarous, S. & Plevyak, T. (1998).Telecommunications Network Management: technologies and implementations. IEEE Press. Piscataway, N.J.

Aha, D.W. (1999). The AAI-99 KM/CBR Workshop: summary of contribuitions. Challenges for Case-Based Reasoning – Proceedings of the ICCBR'((Workshops, pp II.37 II.44

Aha, D.W. & Avila, H.M. (1999). Exploring Synergies of Knowledge Management and Case Base Reasoning: AAAI Workshop. Technical Report AIC-99-008. Washington.

Anderson, J. R. (1983). The architecture of cognition. Harvard University Press, Cambridge.

Bartholomew, M.F. (1997). Succesful business strategies using telecommunications services. Artech House, Inc. Norwood, MA.

Bergmann, R.; Breen, S.; Göker, M.; Manago, M. & Wess, S. (1999). Developing industrial case base reasoning applications: the INRECA methodology. Springer, Berlin.

Bergmann, R. & Althoff, K. (1998). Methodology for building CBR applications. Case base reasoning technology: from foundations to applications. Springer, Berlin., pp. 299-326.

Carbonell, J. (1986). Derivational analogy; A theory of reconstructive problem solving and expertise acquisition. Machine Learning - Ann Artificial Intelligence Approach, Vol II, Morgan Kaufmann, pp. 371-392.

Caulier, P. & Houriez, B. (1995). A case-based reasoning assistance system in telecommunication networks management. Proceedings: Workshop auf der 3. Deutschen Expertensystemtagung - XPS-95, pp. 97-105. Germany

Davenport, T.H. & Prusak, L. (1998). Working knowledge: hor organizations manage what they know. Harward Business School Press.

Eysenck, M.W. & Keane, M. T. (1990). Cognitive psychology: a student's hand book. Hove:LEA.

Eriksson, H-E. & Penker, M. (1998). UML Toolkit. John Wiley & Sons, Inc.

Framingham, M. (1999). IDC defines a path to help IT organizations build successful eknowledge. IDC Press Releases. 06/12/99.

Gentner, D. (1983). Structure Mapping – A Theoretical Framework for Analogy. Cognitive Science, Vol. 7, pp. 155-170.

Gresse, C. (2000). Reutilização Baseada em Casos de Experiência na Área de Mensuração em Engenharia de Software. Tese de Doutorado, Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina.

Gresse, C. (1999). REMEX - A case based approach for reuse of sotware measurement experienceware. In Proceedings of 3rd Int. Conference on Case-Based Reasoning, Germany.

Gresse, C.; Althoff, K. & Barcia, R.M. (1999). Intelligent Retrieval of Technologies for Packing and Reusing Software Engineering Experiences, IESE-Report, nº 055.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.

Gresse, C.; Ramos, A. M.; Althoff, K.; Barcia, R.M. & Martins, A. (1998). Case-based reasoning approach to reuse of experiential knowledge in software measurement programs. 6th German Workshop On Case-Based Reasoning, Berlin, pp 109-118.

Goonatilake, S. & Khebbal, S. (1995). Intelligent hybrid systems. John Willey & Sons Ltd.

Haley. P. (1999). Exploring synergies between knowledge management and case-based reasoning. Exploring Synergies of Knowledge Management and Case Base Reasoning: AAAI Workshop. Technical Report AIC-99-008. Washington, pp 28-29.

IDC (2000). Knowledge management hits its stride. IDC Sevices Supliers. http://www.idc.com/services/press/PR/SV052300PR.stm, 23/05/2000.

ISO/IEC 7498. (1984) - Information Processing Systems - Oprn Systems Interconnection - Basic Reference Model.

Jaczynski, M. & Trousse, B. (1998). An object-oriented framework for the design and the implementation of case-based reasoners. 6th German Workshop On Case-Based Reasoning, Berlin, pp 33-41.

Kamp, G.; Lange, S. & Globig, C. (1998). Related Areas. Case base reasoning technology: from foundations to applications. Springer, Berlin., pp. 327-351.

Kolodner, J. (1993). Case-Based Reasoning. Morgan Kaufmann, Los Altos, CA.

Leake, D. (1996). Case-Based Reasoning: Experiences, Lessons, e Future Directions. AAAI Press/The MIT Press, Menlo Park, California.

Lenz, M.; Burkhard, H. & Brückner, S. (1996). Applying case retrieval nets to diagnostic tasks in technical domains. Smith & Faltings, pp 219-233.

Lenz, M.; Bartsch-Spörl, B.; Burkhard, H. & Wess, S. (1998). Case base reasoning technology: from foundations to applications. Springer, Berlin.

Lewis, L. (1995). Managing computer networks: a case base reasoning approach. Artech House, INC. Norwood, MA.

Manago, M.; Bergmann, R.; Conruyt, N.; Traphöner, R.; Pasley, J.; Le Renard, J.; Maurer, F.; Wess, S.; Althoff, K.D. & Garry, S. (1994). CASUEL: A Commum Case Representation Language. INRECA Deliverable D1, Version 2.01.

Mattison, R. (1997). Data warehousing and data mining for telecommunications. Artech House, Inc. Norwood, MA.

Matlin, M. W. (1998). Cognition. Harcourt Brace & Company. Orlando, Florida.

Melchiors, C. & Tarouco, L.M.R. (1999). Fault management in computer networks using case-based reasoning: DUMBO System. In proceedings: Third International Conference on Case Based Reasoning, ICCBR-99, Seeon Monastery, Germany.

Murray. G. (1999). Knowledge Management Factbook. IDC Bulletin. Disponível via Internet. URL: http://www.idc.com, página acessada em 05/09/99.

M.3010. (1996). ITU-T Recomendation M.3010, Principles for a telecommunications management network.

M.3020. (1995). ITU-T Recomendation M.3010, TMN interface specification methodology.

M.3100. (1995). ITU-T Recomendation M.3100, Generic network information model.

M.3101. (1995). ITU-T Recomendation M.3101, Managed object conformance statements for the generic network information model.

M.3180. (1992). ITU-T Recomendation M.3180, Catalogue of TMN management information.

M.3200. (1997). ITU-T Recomendation M.3200, Generic network information model - Function Definition.

M.3320. (1997). ITU-T Recomendation M.3320, Management requirements framework for the TMN X interface.

M.3300. (1997). ITU-T Recomendation M.3300, F-Interface management capabilities.

M.3400. (1997). ITU-T Recomendation M.3400, Generic network information model - Service Definition.

Penido, G.; Nogueira, J. M. & Machado, C. (1999). An automatic fault diagnosis and connection system for telecommunications management. Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management, Boston, MA.

Q.811. (1997). ITU-T Recomendation Q.811, Lower Layer Protocol Profiles for the Q3 and X Interfaces.

Q.812. (1997). ITU-T Recomendation Q.811, Uper Layer Protocol Profiles for the Q3 and X Interfaces.

Ramos, A. M.; Alves, J. B. M. & Specialski, E. S. (1997). Gerenciamento de redes visto como um sistema multi-agente híbrido com suporte a tomada de decisão. Trabalho apresentado no Workshop em Gerenciamento de Redes, USP, São Paulo, 1998.

Ramos, A. M.; Alves, J. B. M. & Santos, S. P. (1998). Generic Templates for TMN Case Representation, 7th German, Workshop on CBR, Würzburg, Germany.

Ramos, A. M. & Alves, J. B. M. (1999a). Integração de raciocínio basedado em casos à arquitetura TMN, IV Workshop TMN - SBRC'99. Salvador. Brasil.

Ramos, A. M. & Alves, J. B. M. (1999b). Experiential Knowledge in TMN, IEEE LANOMS'99, Rio de Janeiro, Brasil.

Richard, Jean-François. (1990). As Atividades Mentais: Compreender, Raciocinar e Encontrar P. U. F.

Riesbeck, C.K. e Schank, R. C. (1989). Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Publishers, New Jersey.

Ross, B.H. (1989). Some psychological results on case-based reasoning. Case-Based Reasoning Workshop, DARPA. Pensacola Beach. Morgan Kaufmann, pp144-147.

Schank, R. (1982). Dynamic Memory: A theory of learning in computers e people. New York, Cambridge University Press.

Schönberger, S. (1998). Modelagem de informações para supervisão de alarmes no sistema eletrônico de comutação digital/EWSD - interface Q3. Dissertação de Mestrado, Programa de -Graduação em Ciência da Computação, Universidade Federal de Santa Catarina.

Specialski, E. (2000). Modelo de informação baseado em relacionamentos entre objetos gerenciados para a gerência integrada de ambientes de telecomunicações. Tese de Doutorado, -Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina.

Stadler, M. (1993). Case-based reasoning for network management. Proceedings: European Workshop on Case-Based Reasoning. Vol II, pp. 215-220. Germany.

Tecinno Gmbh. (1999). CBR-WORKS 4: Introduction to the database functionality. Revision 1.3,2.

Udupa, D.K. (1999). TMN: telecommunications management network. McGraw-Hill Companies, New York.

Watson, Ian (1997). Applying Case-Based Reasoning: techniques for enterprise systems. Morgan Kaufmann Publishers, Inc. San Francisco, California.

Webber, R. (1998). Pesquisa Jurisprudencial Inteligente. Tese de Doutorado, Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina.

Weiner, A.J.; Thurman, D.A. & Mitchell, C.M. (1995). Applying case-based reasoning to aid fault management in supervisory control. IEEE International Conference on Systems, Man and Cybernetics, pp. 4213-4218.

X.208 | ISSO/IEC 8824. (1988). ITU-T Recomendation X.208. Specification of Abstract Syntax Notation One.

X.500. (1993). ITU-T Recomendation X.500. Information Technology - Open System Interconnection - The Directory: Overview of concepts, models and services.

X.701 | ISO/IEC 10040. (1992). ITU-T Recomendation X.701. Information Technology - Open Systems Interconnection - System Management Overview.

X.710 | ISO/IEC 9595. (1991). ITU-T Recomendation X.710. Common Management Information Service Definition.

X.711 | ISO/IEC 9596. (1991). ITU-T Recomendation X.711. Common Management Information Protocol Specification.

X.720 | ISO/IEC 10165-1. (1992). ITU-T Recomendation X.720. Management Information Model.

X.721 | ISO/IEC 10165-2. (1992). ITU-T Recomendation X.721. Definition of Management Information.

X.722 | ISO/IEC 10165-4. (1992). ITU-T Recomendation X.722. Guidelines for Definition of Managed Objects.

ANEXO.A – ESPECIFICAÇÃO UML

A especificação apresentada neste anexo, através da linguagem UML (*Unified Modeling Language*), visa facilitar a compreensão da aplicação do modelo proposto por esta tese e, conseqüentemente, a comunicação entre técnicos e usuários. Não é voltado a soluções técnicas ou detalhes de código ou programas, mas a um entendimento dos requerimentos e de casos de uso reais do modelo proposto.

O modelo proposto nesta tese segue a filosofia de sistemas abertos, adotada pela OSI e ITU-T, onde não implica no uso de alguma implementação específica ou tecnologia. Projetistas de sistemas podem desenvolver diferentes tipos de soluções para este modelo, seguindo a arquitetura global proposta.

No sentido de facilitar o processo de desenvolvimento, é apresentado a seguir um conjunto de diagramas, usando a notação da linguagem UML, para permitir a visualização, especificação, construção e documentação de artefatos iniciais do modelo proposto.

A integração do modelo proposto, apresentada no capítulo 5 - item 5.3, é descrita, mais detalhadamente, através da apresentação de diagramas de uso de caso, da especificação textual dos casos de uso e, também, através de diagramas de seqüência.

Os diagramas de caso de uso exibem um conjunto de caso de uso e atores, e seus relacionamentos. Diagrama de caso de uso abrangem uma visão estática do sistema. Esses diagramas são importantes para a organização e modelagem do comportamento do modelo proposto.

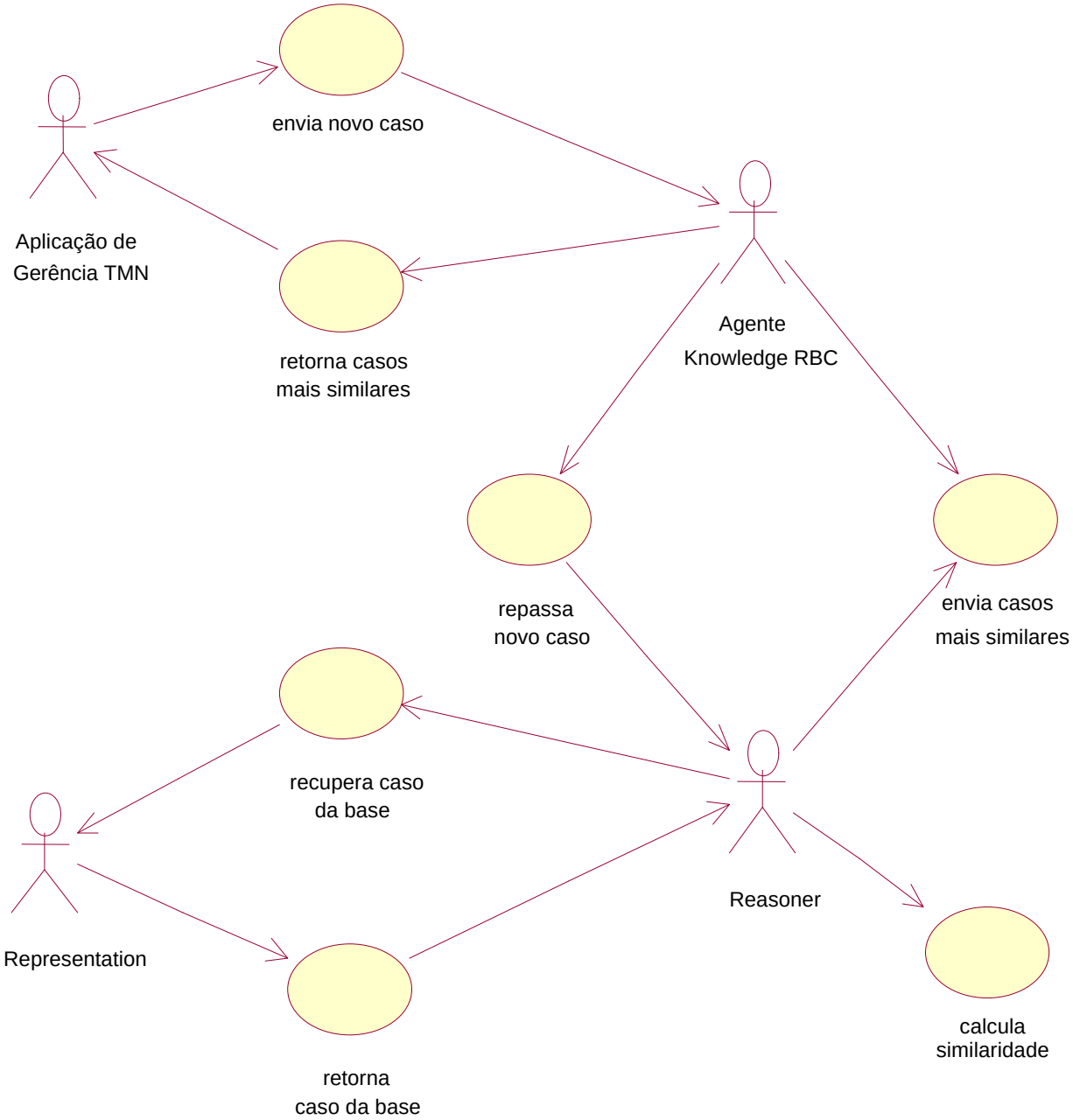
Em contrapartida, os diagramas de seqüência são diagramas de interação, enfocam a visão dinâmica de um sistema, dando ênfase à ordenação temporal das mensagens trocadas entre os objetos do sistema.

Este anexo está organizado em dois subitens: aplicação básica e aplicação avançada. A aplicação básica é desenvolvida através da utilização básica do modelo proposto, onde a integração, com uma plataforma de gerência TMN, se d
conhecimento simples, sem a utilização de objetos discriminadores (item 5.3). A aplicação avançada é mais elaborada e permite a instanciação de objetos dinamicamente na base de casos, utilizando objetos discriminadores (item 5.4).

Para cada tipo de aplicação serão apresentados os diagramas de uso de caso e suas descrições textuais individuais, onde são definidos o objetivo de cada um, suas pré-condições, pós-condições, quem os inicia e o fluxo principal.

A.1 APLICAÇÃO BÁSICA

A.1.1 Diagrama de Casos de Uso



A.1.2 Descrição Textual dos Casos de Uso

CASO DE USO: Envia novo caso

Objetivo: Invocar um agente knowledge RBC, para dar suporte a um profissional de gerência de redes e serviços de telecomunicações, através de uma aplicação

Pré-condições: Ocorrência de algum problema ou situação, no ambiente de redes e serviços de telecomunicações, onde o profissional de gerência não dispõe do conhecimento necessário para decidir qual ação tomar ou solução adotar.

Iniciado por: Processo de aplicação gerente TMN.

Fluxo principal:

- A aplicação gerente envia o novo caso a ser solucionado;
- O agente knowlege RBC é invocado para recuperar casos similares a este, em sua base de casos – MCB.

Pós-condições: Agente knowledge RBC é invocado.

CASO DE USO: Repassa novo caso

Objetivo: Repassar o novo caso a fim de que sejam identificados e recuperados os casos mais similares;

Pré-condições: A invocação de um agente knowledge RBC.

Iniciado por: Agente knowledge RBC.

Fluxo principal:

- repassa o novo caso para objetos da classe rreasoning.

Pós-condições:

- Um objeto da classe rreasoning é invocado;
- O agente knowledge RBC fica aguardando a recuperação de casos similares.

CASO DE USO: Recupera caso da base

Objetivo: Solicita casos da base MCB, para calcular a similaridade com o novo caso;

Pré-condições: Uma instância da classe representation é invocada por uma instância da classe reasoning para recuperar casos da base MCB;

Iniciado por: Instância da classe reasoning;

Fluxo principal:

- Uma instância da classe reasoning envia um pedido para que sejam recuperados casos da base MCB, a fim de que possa ser calculado para cada um, o grau de similaridade, com o novo caso;
- Fica aguardando casos da base MCB;

Pós-condições:

- Uma instância da classe representation
- Uma instância da classe reasoning fica aguardando resposta.

CASO DE USO: Retorna caso da base

Objetivo: Repassar casos da base MCB, para calcular a similaridade com o novo caso;

Pré-condições: Uma instância da classe representation tenha sido invocada por uma instância da classe reasoning;

Iniciado por: Instância da classe representation;

Fluxo principal:

- Uma instância da classe representation envia os casos da base MCB, a fim de que seja calculado para cada um, o grau de similaridade, com o novo caso.

Pós-condições: A instância da classe reasoning, originadora da solicitação, é atendida.

CASO DE USO: Calcula similaridade

Objetivo: Calcular a similaridade entre os casos da base MCB e o novo caso em questão;

Pré-condições: O recebimento do novo caso, repassado pelo agente knowlegde RBC; e o recebimento de casos da base MCB, repassados por uma instância da classe representation.

Iniciado por: Instância da classe reasoning;

Fluxo principal:

- O agente knowledge RBC repassa o novo caso para objeto da classe reasoning;
- Uma instância da classe representation repassa os casos da base MCB;
- Calcula o grau de similaridade entre o novo caso e os casos da base MCB;

Pós-condições: O grau de similaridade calculado para cada caso da base;

CASO DE USO: Envia casos mais similares

Objetivo: Enviar os casos mais similares com o novo a fim de que possam ser usados pela aplicação de gerência;

Pré-condições: O grau de similaridade para todos os casos da base MCB tenha sido calculado.

Iniciado por: Instância da classe reasoning;

Fluxo principal:

- Após o calcular o grau de similaridade para todos os casos da base MCB, identifica os mais similares;
- Envia os casos mais similares;

Pós-condições: Recuperado os casos mais similares;

CASO DE USO: Repassa casos mais similares

Objetivo: Repassar os casos mais similares com o novo caso em questão;

Pré-condições: Tenham sido recuperados os casos mais similares com o novo caso em questão.

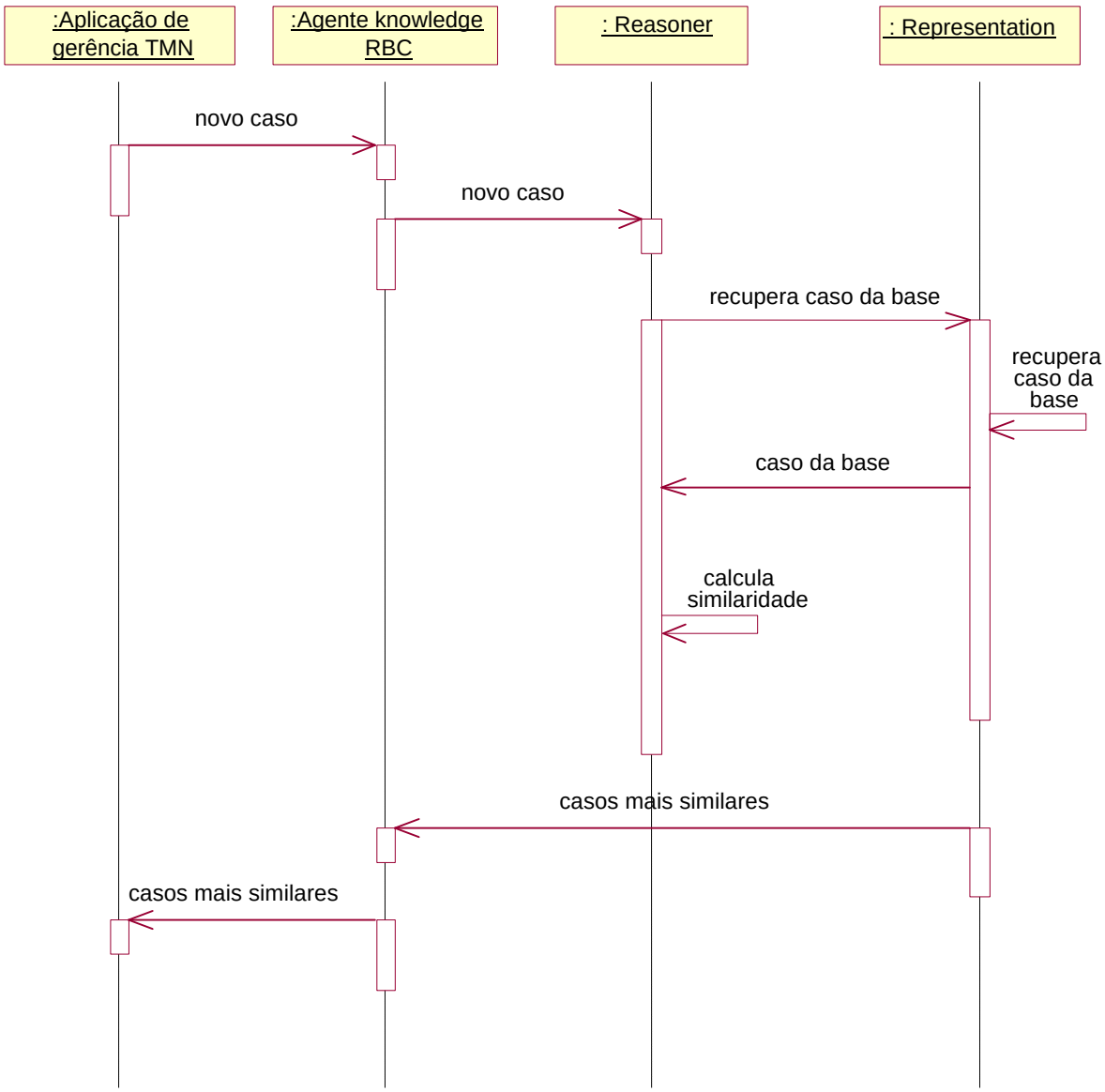
Iniciado por: Agente de knowledge RBC;

Fluxo principal:

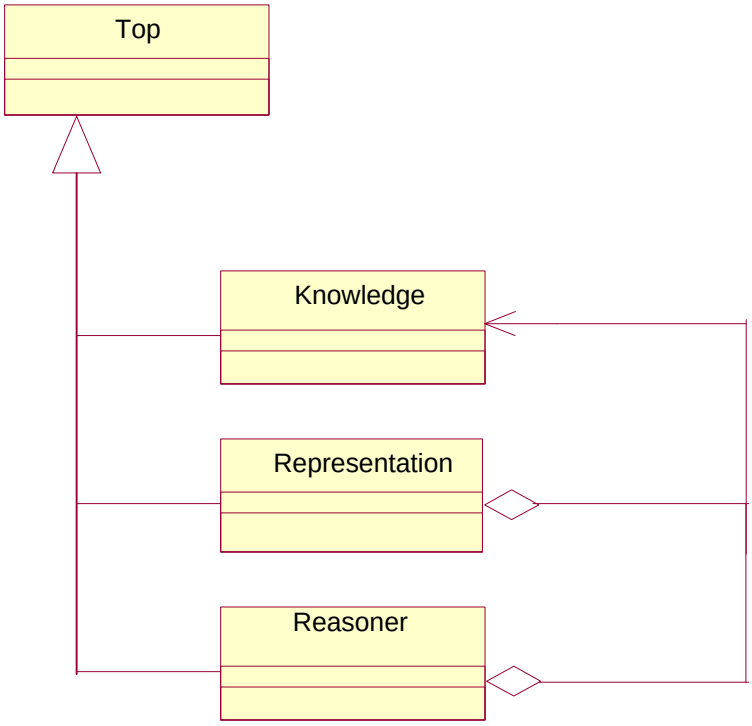
- Repassar os casos mais similares com o novo caso em questão

Pós-condições: Busca realizada; Agente knowledge RBC fica aguardando nova consulta.

A.1.3 Diagrama de Seqüência

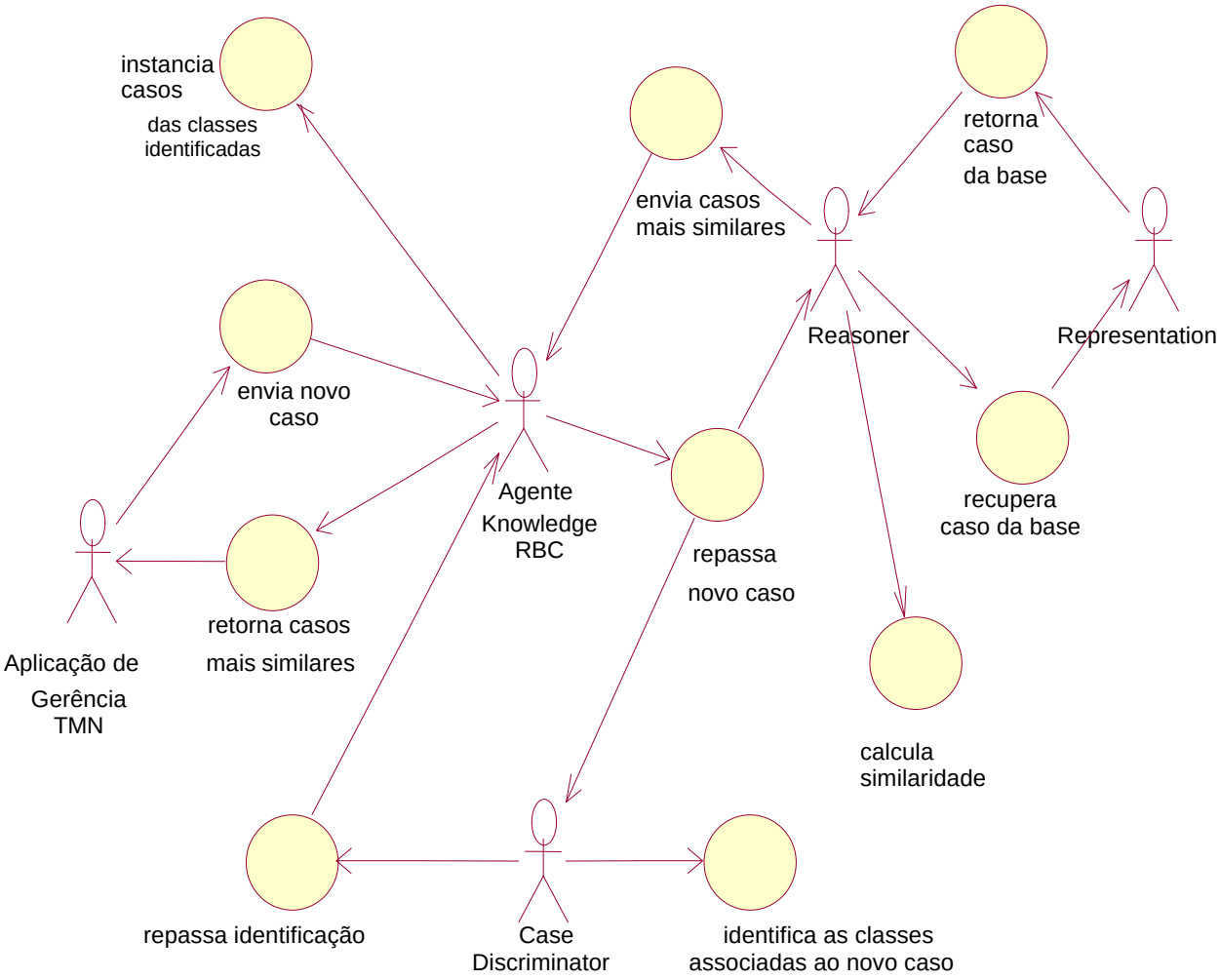


A.1.4 Diagrama de Classes



A.2 APLICAÇÃO AVANÇADA

A.2.1 Diagrama de Casos de Uso



A.2.2 Descrição Textual dos Casos de Uso

CASO DE USO: Envia novo caso

Objetivo: Invocar um agente knowledge RBC, para dar suporte a um profissional de gerência de redes e serviços de telecomunicações, através de uma aplicação de gerência TMN.

Pré-condições: Ocorrência de algum problema ou situação, no ambiente de redes e serviços de telecomunicações, onde o profissional de gerência não dispõe do conhecimento necessário para decidir qual ação tomar ou solução adotar.

Iniciado por: Processo de aplicação gerente TMN.

Fluxo principal:

- A aplicação gerente envia o novo caso a ser solucionado;
- O agente knowlege RBC é invocado para recuperar casos similares a este, em sua base de casos – MCB.

Pós-condições: Agente knowledge RBC é invocado.

CASO DE USO: Repassa novo caso

Objetivo: Repassar o novo caso a fim de que sejam identificados e recuperados os casos mais similares;

Pré-condições: A invocação de um agente knowledge RBC.

Iniciado por: Agente knowledge RBC.

Fluxo principal:

- O agente knowledge RBC repassa o novo caso para objetos da classe reasoning e da classe case discriminator;

Pós-condições:

- Um objeto da classe case discriminator é invocado;
- Um objeto da classe reasoning é invocado;
- O agente knowledge RBC fica aguardando a recuperação de casos similares.

CASO DE USO: Identifica classes associadas ao novo caso

Objetivo: Identificar as classes a que pertencem o novo caso, a fim de que possam ser instanciadas dinamicamente na base de casos. Conforme as propriedades do novo caso, discrimina a classe dos casos similares e qual a classe do método de raciocínio para se estabelecer o grau de similaridade com o novo caso.

Pré-condições: Recebimento de um novo caso.

Iniciado por: Instância da classe case discriminator.

Fluxo principal:

- Um agente knowledge RBC repassa um novo caso para uma instância da classe case discriminator;
- Baseado em suas asserções e nas propriedades do novo caso, o objeto case discriminator identifica as subclasses da classe representation e da classe reasoning as quais o novo caso se enquadra.

Pós-condições: identificação efetuada.

CASO DE USO: Repassa identificação

Objetivo: Repassar a identificação das classes associadas ao novo caso.

Pré-condições: Identificação efetuada.

Iniciado por: Instância da classe case discriminator.

Fluxo principal:

- Instância da classe case discriminator repassa a identificação das classes associadas ao novo caso.

Pós-condições:

- O agente knowledge RBC é invocado.

CASO DE USO: Instancia casos das classes associadas

Objetivo: Instanciar dinamicamente os casos das classes associadas ao novo caso;

Pré-condições: As classes associadas tenham sido identificadas;

Iniciado por: Agente knowledge RBC;

Fluxo principal:

- Agente knowledge RBC crias as instâncias das classes associadas, ou seja, cria os casos na base de casos a fim de que possam ser comparados, similarmente, com o novo caso;
- Somente são instanciados na base de casos MCB, os casos associados;

Pós-condições: Instâncias de classes, associadas ao novo caso, criadas dinamicamente na base de casos MCB.

CASO DE USO: Recupera caso da base

Objetivo: Solicita casos da base MCB, para calcular a similaridade com o novo caso;

Pré-condições: Uma instância da classe representation é invocada por uma instância da classe reasoning para recuperar casos da base MCB;

Iniciado por: Instância da classe reasoning;

Fluxo principal:

- Uma instância da classe reasoning envia um pedido para que sejam recuperados casos da base MCB, a fim de que possa ser calculado para cada um, o grau de similaridade, com o novo caso;
- Fica aguardando casos da base MCB;

Pós-condições:

- Uma instância da classe representation é invocada;
- Uma instância da classe reasoning fica aguardando resposta.

CASO DE USO: Retorna caso da base

Objetivo: Repassar casos da base MCB, para calcular a similaridade com o novo caso;

Pré-condições: Uma instância da classe representation tenha sido invocada por uma instância da classe reasoning;

Iniciado por: Instância da classe representation;

Fluxo principal:

- Uma instância da classe representation envia os casos da base MCB, a fim de que seja calculado para cada um, o grau de similaridade, com o novo caso.

Pós-condições: A instância da classe reasoning, originadora da solicitação, é atendida.

CASO DE USO: Calcula similaridade

Objetivo: Calcular a similaridade entre os casos da base MCB e o novo caso em questão;

Pré-condições: O recebimento do novo caso, repassado pelo agente knowlegde RBC; e o recebimento de casos da base MCB, repassados por uma instância da classe representation.

Iniciado por: Instância da classe reasoning;

Fluxo principal:

- O agente knowledge RBC repassa o novo caso para objeto da classe reasoning;
- Uma instância da classe representation repassa os casos da base MCB;
- Calcula o grau de similaridade entre o novo caso e os casos da base MCB;

Pós-condições: O grau de similaridade calculado para cada caso da base;

CASO DE USO: Envia casos mais similares

Objetivo: Enviar os casos mais similares com o novo a fim de que possam ser usados pela aplicação de gerência;

Pré-condições: O grau de similaridade dos casos da base MCB tenha sido calculado.

Iniciado por: Instância da classe reasoning;

Fluxo principal:

- Após o calcular o grau de similaridade para todos os casos da base MCB, identifica os mais similares;
- Envia os casos mais similares;

Pós-condições: Recuperado os casos mais similares;

CASO DE USO: Repassa casos mais similares

Objetivo: Repassar os casos mais similares com o novo caso em questão;

Pré-condições: Tenham sido recuperados os casos mais similares com o novo caso em questão.

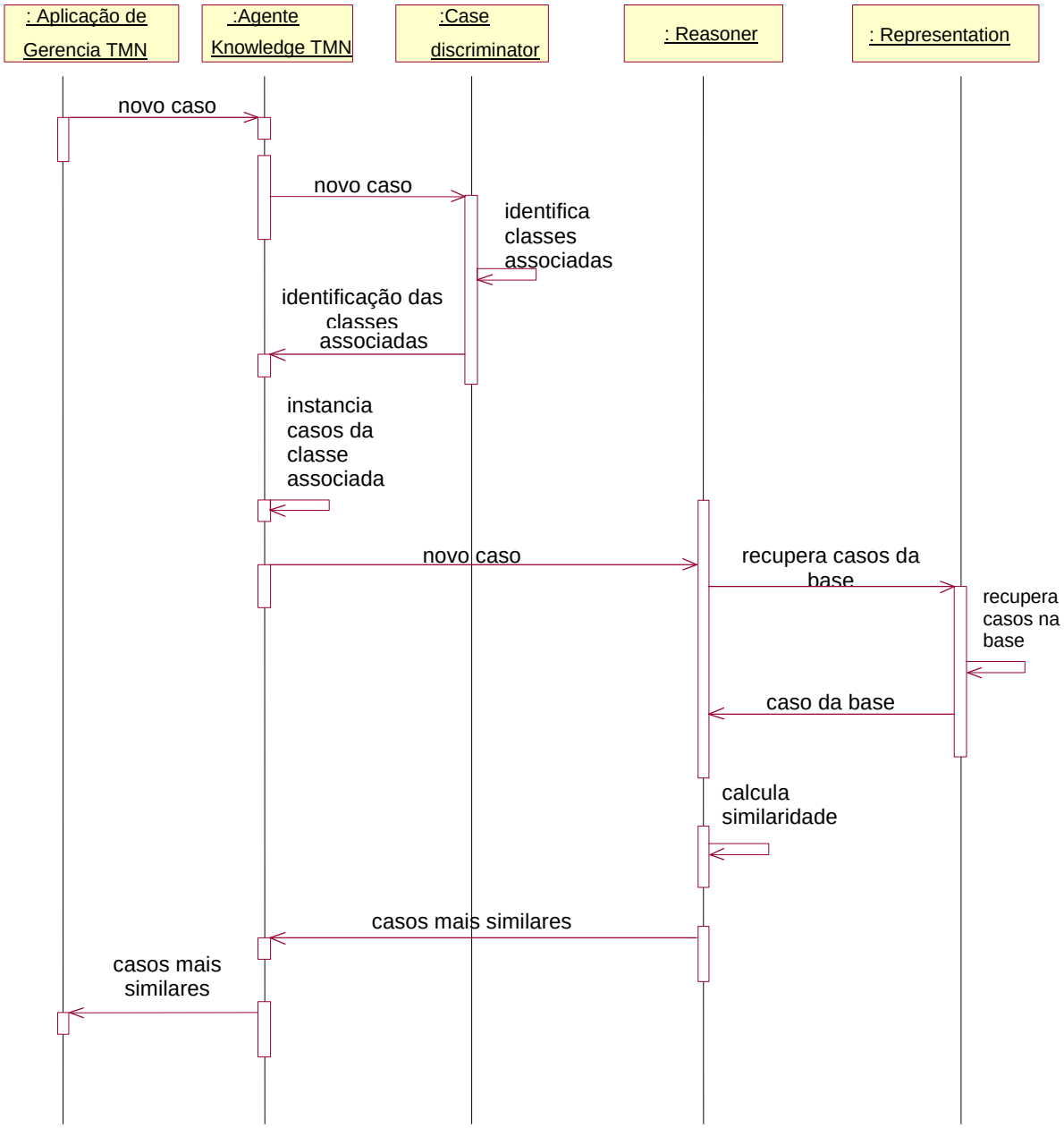
Iniciado por: Agente de knowledge RBC;

Fluxo principal:

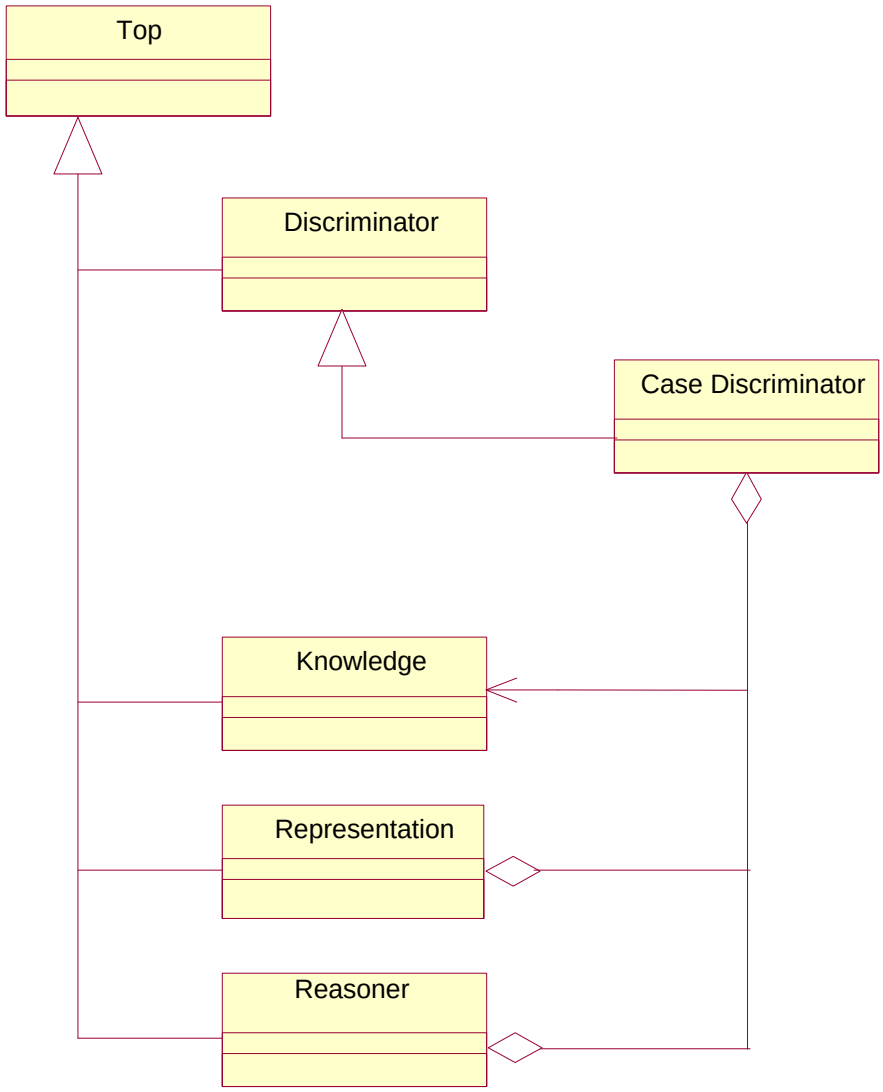
- Repassar os casos mais similares com o novo caso em questão

Pós-condições: Busca realizada; Agente knowledge RBC fica aguardando nova consulta.

A.2.3 Diagrama de Seqüência



A.2.4 Diagrama de Classes



ANEXO.B – NOTAÇÃO ASN.1

B.1 DEFINIÇÕES ASN.1

ObjectIdentifier ::= OBJECT IDENTIFIER
AditionalInformation ::= CHARACTER STRING
GenericIdentifier ::= GraphicString
IdentifierSet ::= SET OF GraphicString
Peso ::= REAL {0 ..1}
Indexes ::= SET OF SEQUENCE {AttributeId, Peso}

B.2 TEMPLATES DOS ATRIBUTOS PROPOSTOS PELO MODELO

Alguns módulos de definições de tipos ASN.1 especificados em outras recomendações são também utilizados para a definição dos atributos introduzidos por este modelo, através do uso de construções IMPORT[X.208]. A seguir são apresentad
templates para os atributos criados a partir do modelo proposto.

IMPORTS

AttributeId FROM CMIP {joint-iso-ccitt ms(9) cmip(1) modules(0) protocol(3)}

smi2AttributeID ObjectIdentifier, AditionalInformation, GenericIdentifier, IdentifierSet {joint-iso-ccitt ms(9) smi(3) part2(2) attribute(7)}

caseCauseProblem ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined IdentifierSet;

MATCHES FOR EQUALITY;

BEHAVIOUR caseCauseProblemBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo define a causa do problema,. Por exemplo, o conjunto de alarmes que por ventura foram gerados emf função do problema causado.";;

REGISTERED AS {representationAttribute1};

caseContextId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIcIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR caseContextIdBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o contexto do problema.";;

REGISTERED AS {representationAttribute2};

caseContextType ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR caseContextTypeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o tipo do contexto do problema.";;

REGISTERED AS {representationAttribute3};

caseDeveloper ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined AditionalInformation;

MATCHES FOR EQUALITY;

BEHAVIOUR caseDeveloperBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o responsável pelo desenvolvimento da base de casos";;

REGISTERED AS {representationAttribute4};

caseEffectProblem ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined AditionalInformation;

MATCHES FOR EQUALITY;

BEHAVIOUR caseEffectProblemBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica os efeitos causados pelo problema.";;

REGISTERED AS {representationAttribute5};

caseId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR caseIdBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica a base de casos.";;

REGISTERED AS {representationAttribute6};

caseObjetcProblem ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR caseObjetcProblemBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o objeto do problema. Por exemplo, se é um determinado equipamento, uma determinada aplicação, etc.";;

REGISTERED AS {representationAttribute7};

caseOutcomeAssessment ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined AditionalInformation;

MATCHES FOR EQUALITY;

BEHAVIOUR caseOutcomeAssessmentBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica os resultados esperados com a soluçãoempregada.";;

REGISTERED AS {representationAttribute8};

caseProblemId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR caseProblemIdBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o problema.";;

REGISTERED AS {representationAttribute9};

caseProblemType ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR caseProblemTypeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o tipo do problema.";;

REGISTERED AS {representationAttribute10};

caseSimilarityRelevantIndexes ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined Indexes;

MATCHES FOR EQUALITY, ORDERING;;

BEHAVIOUR caseSimilarityRelevantIndexesBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica os atributos do caso e seus respectivos graus de relevância utilizados pela função recuperação no cálculo de similaridade.";;

REGISTERED AS {representationAttribute11};

caseSimilarityGoal ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined Indexes;

MATCHES FOR EQUALITY, ORDERING;;

BEHAVIOUR caseSimilarityGoalBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica para qual objetivo é orientado o conjunto de índices.";;

REGISTERED AS {representationAttribute12};

caseSimilarityId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR caseSimilarityIdBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica qual o método de similaridade aplicável.";;

REGISTERED AS {representationAttribute13};

caseSolutionId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR caseSolutionIdBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica a solução aplicada.";;

REGISTERED AS {representationAttribute14};

caseSolutionType ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR caseSolutionTypeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o tipo de solução aplicada.";;

REGISTERED AS {representationAttribute15};

caseTitle ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR caseTitleBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica um rótulo para a base de casos.";;

REGISTERED AS {representationAttribute16};

caseType ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR caseTypeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o tipo da base de casos.";;

REGISTERED AS {representationAttribute17};

REGISTERED AS {representationAttribute20};

reasoningId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR reasoningIdBehaviour BEHAVIOUR

DEFINED AS

“O valor deste atributo identifica o método de raciocínio agregado àArquitetura TMN.”;;

REGISTERED AS {representationAttribute21};

REGISTERED AS {representationAttribute22};

reasoningType ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR reasoningTypeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o tipo de método de raciocínio agregado àArquitetura TMN.";;

REGISTERED AS {representationAttribute23};

representationId ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;

BEHAVIOUR representationIdBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica a representação do conhecimento agregado àArquitetura

REGISTERED AS {representationAttribute24};

representationTitle ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR representationTitleBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica um rótulo para a representação do conhecimento agregada

REGISTERED AS {representationAttribute25};

representationType ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR representationTypeBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o tipo de representação conhe

REGISTERED AS {representationAttribute26};

supportReasoning ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR supportReasoningBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica o método de raciocínio suportado pela aplicação

REGISTERED AS {representationAttribute27};

supportRepresentation ATTRIBUTE

WITH ATTRIBUTE SINTAX ASN.1 Defined GenericIdentifier;

MATCHES FOR EQUALITY;;

BEHAVIOUR supportRepresentationBehaviour BEHAVIOUR

DEFINED AS

"O valor deste atributo identifica a representação do conhecimento suportada por esta

REGISTERED AS {representationAttribute28};