FABBIANO FIORIN FERRARI

Uso do LMP para Descoberta Automática e Gerenciamento de Enlaces em Redes OTN

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. Anilton Salles Garcia

VITÓRIA

2011

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Ferrari, Fabbiano Fiorin, 1985-

F375u Uso do LMP para descoberta automática e gerenciamento de enlaces em redes OTN / Fabbiano Fiorin Ferrari. – 2011.

107 f. : il.

Orientador: Anilton Salles Garcia.

Dissertação (Mestrado em Engenharia Elétrica) – Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Simulação (Computadores). 2. Telecomunicações. 3. Redes ópticas de transporte. I. Garcia, Anilton Salles. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 621.3

FABBIANO FIORIN FERRARI

Uso do LMP para Descoberta Automática e Gerenciamento de Enlaces em Redes OTN

Dissertação submetida ao programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para a obtenção do Grau de Mestre em Engenharia Elétrica.

Aprovada em 06 de Setembro de 2011.

COMISSÃO EXAMINADORA

Prof. Dr. Anilton Salles Garcia – Orientador
Universidade Federal do Espírito Santo

Prof. Dr. Antônio Manoel Ferreira Frasson
Universidade Federal do Espírito Santo

Prof. Dr. Magnos Martinello
Universidade Federal do Espírito Santo

DEDICATÓRIA

A meus pais e a toda minha família, por todo o apoio.

AGRADECIMENTOS

Agradeço a Deus por guiar e iluminar meu caminho, e aos meus pais, Roque e Lourdes, por serem a inspiração para alcançar meus ideais.

À minha irmã, Danielle, pelo incentivo e à minha namorada, Mariana, pela excelente companhia e motivação.

Ao professor Anilton, pela orientação e apoio, durante anos de trabalho, que foram fundamentais na realização desta dissertação.

Aos amigos Pedro Paulo e Rodrigo Stange e a todos os membros do projeto que participo, pela ajuda durante o trabalho.

Aos membros da banca, Antônio Manoel Frasson e Magnos Martinello, pelo tempo dedicado à avaliação deste trabalho.

A toda equipe da PADTEC S. A., empresa parceira do projeto de pesquisa que colaborou com o estudo e o desenvolvimento deste trabalho, oferecendo a oportunidade de conhecer na prática as tecnologias estudadas.

Ao Programa de Pós-Graduação em Engenharia Elétrica, PPGEE, e a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES, pela oportunidade e suporte financeiro para realização deste trabalho.

E a todos os familiares e amigos pelo carinho e pensamento positivo.

RESUMO

Para atender a crescente demanda por serviços de telecomunicações, com aplicações exigindo cada vez mais das redes de núcleo, as Redes Ópticas de Transporte (*OTN – Optical Transport Network*), baseadas em multiplexação por comprimento de onda (WDM - *Wavelength-Division Multiplexing*) e no SONET/SDH (*Synchronous Optical NETwork/ Synchronous Digital Hierarchy*), surgem para prover serviços de transporte óptico mais eficiente. A OTN consolida-se como a nova geração de redes ópticas, cabendo à ITU-T (*International Telecommunication Union – Telecommunication Section*) fornecer a padronização para os fabricantes de equipamentos e soluções de redes de transporte.

Uma vez que essas novas tecnologias de redes oferecem maiores possibilidades de tráfego, proteção, restauração, comutação e qualidade de serviço para múltiplos clientes, há um aumento na complexidade da rede como um todo e no modo de gerenciamento desta. Neste cenário, com o objetivo de aprimorar o gerenciamento dessas redes, busca-se a automatização das suas funcionalidades.

Nesta dissertação é apresentada uma proposta de arquitetura para o plano de gerência das redes OTN visando integração das funcionalidades de gerência e o plano de transporte. São desenvolvidos mecanismos para a descoberta automática da topologia e recursos da rede, baseando-se nos procedimentos descritos na recomendação ITU-T G.7714 e na IETF RFC4204 (LMP – *Link Management Protocol*). A arquitetura e os procedimentos implementados são validados através de simulação no *software* OMNeT++, onde um *framework* de simulação de redes OTN está em desenvolvimento pelo grupo de pesquisa do Laboratório de Telecomunicações (LabTel) na UFES com o objetivo de analisar, testar e validar comportamentos e funcionalidades das redes OTN.

Palavras-chave: OTN, G.7714, LMP, Simulação, OMNeT++.

ABSTRACT

In order to meet the growing demand for telecommunications services, with applications increasingly demanding of core networks, the Optical Transport Networks (OTN), based on Wavelength-Division Multiplexing (WDM) and in Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH), appear to provide more efficient optical transport services. The OTN has been consolidating as a new generation of optical networks and the ITU-T (International Telecommunication Union – Telecommunication Section) provides standardization for equipment manufacturers and transport network solutions.

Once these new networking technologies offer greater possibilities of traffic, protection, restoration, switching and quality of service for multiple clients, there is an increase of network and management complexity. In this context, in order to improve the management of these networks, the automation of their functions is desired.

This dissertation proposes an architecture for the management plane of OTN to achieve the integration of management features and the transport plane. Mechanisms are developed for the automatic discovery of network topology and resources, based on the procedures described at ITU-T Recommendation G.7714 and IETF RFC4204 (LMP - Link Management Protocol). The architecture and the procedures implemented are validated by simulation, using OMNeT++, where an OTN simulation framework is being developed by the Telecommunications Laboratory (LabTel) research group at UFES, in order to analyze, test and validate behavior and features of OTN networks.

Keywords: OTN, G.7714, LMP, Simulation, OMNeT++.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS/ACRÔNIMOS

AF – *Atomic Function*

AI – *Adapted Information*

API – *Application Programming Interface*

ASON – *Architecture for the Automatically Switched Optical Network*

ATM – *Asynchronous Transfer Mode*

DA – *Discovery Agent*

DCN – *Data Communications Network*

DTH – *Digital Transport Hierarchy*

DWDM – *Dense Wavelength-Division Multiplexing*

FEC – *Forward Error Correction*

GCC – *General Communications Channel*

GFP – *Generic Framing Procedure*

GMPLS – *Generalized Multi-Protocol Label Switching*

ICMP – *Internet Control Message Protocol*

IGMP – *Internet Group Management Protocol*

IETF – *Internet Engineering Task Force*

IP – *Internet Protocol*

ITU-T – *International Telecommunication Union – Telecommunication Section*

LMP – *Link Management Protocol*

MI – *Management Information*

MIB – *Management Information Base*

MP – *Management Point*

NE – *Network Element*

NED – *Network Description*

OADM – *Optical Add/Drop Multiplexers*

OAM&P – *Operation, Administration, Maintenance and Provisioning*

OCh – *Optical Channel*

ODU – *Optical Channel Data Unit*

O-E-O – *Óptico-Elétrico-Óptico*

OMNeT++ - *Objective Modular Network Testbed*

OMS – *Optical Multiplex Section*

ONE – *Optical Network Element*

OPU – *Optical Channel Payload Unit*

OSC – *Optical Supervisory Channel*

OTH – *Optical Transport Hierarchy*

OTN – *Optical Transport Network*

OTS – *Optical Transmission Section*

OTU – *Optical Channel Transport Unit*

OXC – *Optical Cross-connect*

PPP – *Point-to-Point Protocol*

RFC – *Request for Comments*

ROADM – *Reconfigurable Optical Add/Drop Multiplexers*

SAPI – *Source Access Point Identifier*

SDH – *Synchronous Digital Hierarchy*

Sk – *Sink*

So – *Source*

SONET – *Synchronous Optical NETwork*

TCM – *Tandem Connection Monitoring*

TCP – *Termination Connection Point*

TDM – *Time-Division Multiplexing*

TI – *Tecnologia de Informação*

TTI – *Trail Trace Identifier*

WDM – *Wavelength-Division Multiplexing*

SUMÁRIO

1 INTRODUÇÃO

Esse capítulo apresenta uma introdução sobre o trabalho, contendo: contextualização e motivação, justificativa, objetivos e resultados esperados, metodologia, trabalhos relacionados e as principais contribuições. A estrutura da dissertação também é exibida.

1.1 CONTEXTUALIZAÇÃO E MOTIVAÇÃO

Devido ao crescimento da demanda por serviços de telecomunicações, causado principalmente pelos serviços da internet e o avanço dos serviços de telefonia e televisão digitais, surgiram tecnologias de redes ópticas para transporte com altas taxas de transmissão.

A primeira tecnologia de redes ópticas a ser amplamente utilizada, composta pelas redes SONET (*Synchronous Optical Network*) e SDH (*Synchronous Digital Hierarchy*), utilizava a tecnologia de multiplexação TDM (*Time Division Multiplexing*), e foram originalmente projetadas para serviços ponto a ponto, carregando um sinal óptico por fibra. Com o crescimento da utilização de serviços de dados e vídeo, resultou-se em alguns desafios técnicos para essa tecnologia. Nesse momento, houve a necessidade de uma nova tecnologia, voltada ao tráfego de dados, que oferecesse suporte para requerimentos emergentes como aprovisionamento rápido e automático, re-roteamento e restauração óptica, suporte a múltiplos clientes e de diferentes tipos, e inter-operação das redes IP (*Internet Protocol*) e ópticas.

Nesse contexto, surgiu uma nova tecnologia de redes ópticas de transporte: OTN (*Optical Transport Networks*). Essa tecnologia é baseada em SONET/SDH, herdando muitas de suas funcionalidades, porém, uma das principais diferenças entre elas é a tecnologia de multiplexação utilizada. A OTN oferece suporte ao WDM (*Wavelength Division Multiplexing)*, que oferece suporte à transmissão de dados simultaneamente em múltiplos comprimentos de onda.

Uma vez que essas novas tecnologias de redes de transporte oferecem maiores possibilidades de tráfego, proteção, restauração, comutação e qualidade de serviço para múltiplos clientes, há um aumento na complexidade da rede como um todo e no modo de gerenciamento dessa.

O GMPLS (*Generalized Multi-protocol Label Switching*) surge como um conjunto de protocolos para implementação do plano de controle, que visa controlar as funcionalidades de proteção, restauração e comutação visando a melhor qualidade de serviço. A descoberta e gerenciamento dos recursos existentes na rede forma a base de dados de enlaces e capacidades que é utilizada pelos planos de controle e gerência da rede.

A simulação dos protocolos de descoberta automática (Iniewski, McCrosky, & Minoli, Network Infraestructure and Architecture, 2008) e gerenciamento de enlaces (Lang, 2005) permite avaliar os aspectos de implementação dessas funcionalidades aplicadas às redes OTN e integradas com os demais protocolos e processos dos planos de gerência e controle da rede.

1.2 JUSTIFICATIVA

Devido a crescente utilização de soluções voltadas às tecnologias OTN e WDM, com o plano de controle GMPLS, em conjunto com os vários projetos relacionados à simulação dessas tecnologias, realizados no LabTel (Laboratório de Telecomunicações) na UFES, verifica-se a necessidade de oferecer uma base de dados com informações dos recursos existentes na rede para a continuidade do trabalho nos planos de gerência e controle da rede.

A descoberta de topologia é uma funcionalidade fundamental em redes, sendo uma primitiva básica para diversas funcionalidades de gerenciamento e controle na rede. Particularmente em redes OTN, o desafio de descobrir a topologia e recursos suportados é ainda maior pelo fato dos elementos de rede (NEs) permitir topologias arbitrárias que dependem da camada em questão da arquitetura das redes OTN.

Os trabalhos encontrados na literatura, como em (Perelló, Escalona, Spadaro, Comellas, & Junyent, 2007), (Zhou & Chi, 2004) e (Papadimitriou, Poppe, & Rousseau, 2004) fazem referência ao LMP como protocolo para efetuar o gerenciamento de enlaces na rede e descoberta de conexões.

O processo de implementação de protocolos e funcionalidades descritas pelos órgãos de normatização nem sempre é simples. A tarefa de integração desses protocolos e funcionalidades com os demais procedimentos dos planos de transporte, gerência e controle se mostra um estudo complexo e, muitas vezes, surge a necessidade de propor novas arquiteturas e métodos. Para isso é utilizado simulação para analisar e validar a implementação dessas novas funcionalidades nas redes OTN.

Esta dissertação é uma continuidade de outros trabalhos desenvolvidos e em desenvolvimento do grupo de pesquisa, portanto, é natural o desenvolvimento das novas funcionalidades no simulador de eventos discretos OMNeT++ (*Objective Modular Network Testbed*), *software* livre e gratuito para uso acadêmico, que tem sido usado pelos trabalhos do grupo.

1.3 OBJETIVOS E RESULTADOS ESPERADOS

Esta dissertação tem como objetivo geral estudar e desenvolver uma solução para descoberta e gerenciamento dos enlaces em uma rede OTN, provendo as informações necessárias para outros processos dos planos de gerência e controle da rede.

Para alcançar este objetivo geral, os seguintes objetivos específicos são estabelecidos: (1) desenvolver uma proposta de arquitetura de gerência para comunicação entre os elementos da rede, tratando mensagens de descoberta e informações de alarmes; (2) aprimorar e integrar o trabalho desenvolvido em (Ferrari, 2009), baseado na recomendação (ITU-T, G.7714, 2005), ao plano de transporte e gerência da rede OTN; (3) utilizar o Protocolo de Gerenciamento de Enlaces (LMP) do GMPLS para obter e trocar informações de capacidades dos

enlaces entre os nós da rede; (4) implementar os protocolos mencionados e integrar ao *framework* de simulação de redes OTN no OMNeT++.

Como resultado, espera-se obter os dados de topologia, os enlaces e suas capacidades de modo automático, criando grafos que representem a topologia da rede e uma base de dados com informações dos enlaces existentes entre os nós.

1.4 METODOLOGIA

Para chegar ao objetivo da dissertação, a seguinte metodologia é utilizada:

- Estudo das recomendações ITU-T e RFCs (*Request for Comments*) do IETF (*Internet Engineering Task Force*) sobre as tecnologias OTN e GMPLS, respectivamente;
- Estudo de trabalhos relacionados ao gerenciamento de recursos em redes OTN;
- Desenvolvimento de uma proposta de arquitetura de gerência, atividade realizada em conjunto com o autor no trabalho (Tessinari, 2011);
- Integração da funcionalidade de descoberta automática no *framework* de simulação OTN, seguindo os mecanismos descritos na recomendação (ITU-T, G.7714.1, 2010);
- Implementação do LMP para estabelecimento de canais de controle na rede e troca de capacidades entre os nós;
- Validar a arquitetura e os métodos implementados através de simulação de redes com topologias distintas.

1.5 TRABALHOS RELACIONADOS

Esta dissertação está inserida no contexto de um conjunto de trabalhos desenvolvidos pelo grupo de pesquisa do LabTel na UFES. Esses trabalhos visam a

criação de um *framework* de simulação de redes OTN no simulador de redes OMNeT++.

Os trabalhos desenvolvidos focam nas funcionalidades dos planos de transporte, controle e gerência da rede e são abordados na Seção 2.3. Destacam-se os trabalhos: (Ferrari, 2009), (Favoreto, 2009), (Tessinari, 2009), (Frigini, 2010) e (Tessinari, 2011). Outras funcionalidades continuam em desenvolvimento pelo grupo de pesquisa citado.

1.6 PRINCIPAIS CONTRIBUIÇÕES

As principais contribuições trazidas por esta dissertação são:

- Proposta de arquitetura para o plano de gerência;
- Aprimoramento e integração da funcionalidade de descoberta automática;
- Análise dos principais aspectos da implementação dos procedimentos essenciais do LMP aplicados à rede OTN;
- Protótipo de simulação com funcionalidades de identificação de enlaces e suas capacidades nas redes OTN configuradas.

1.7 ORGANIZAÇÃO DO TEXTO

A dissertação apresenta em seu Capítulo 2 a fundamentação teórica no que se refere às redes ópticas de transporte. São discutidos a evolução, principais recomendações que definem as redes OTN e sua arquitetura em camadas. Também são apresentados os trabalhos relacionados com breves descrições de cada.

No Capítulo 3 é apresentado o protocolo de gerenciamento de enlaces do LMP, seus principais procedimentos e máquinas de estados. Ao final do Capítulo, também são apresentados alguns aspectos referentes à aplicação do LMP nas redes OTN.

O Capítulo 4 aborda a arquitetura do plano de gerência, os principais aspectos relacionados com a integração da funcionalidade de descoberta automática aos planos de transporte e gerência e as questões envolvidas no desenvolvimento do módulo do LMP.

No Capítulo 5 são abordados o simulador OMNeT++ e os principais elementos e módulos implementados.

O Capítulo 6 apresenta os testes executados com o objetivo de validação dos elementos implementados e seus respectivos resultados, mostrados na forma de grafos e tabelas de informações de cada nó, geradas na simulação.

No Capítulo 7 são discutidas as conclusões desta dissertação e apresentadas as sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo apresenta-se a OTN (*Optical Transport Network*), que representa uma evolução das redes ópticas de transporte, desenvolvida com o objetivo de combinar os benefícios da tecnologia SONET/SDH (*Synchronous Optical Network / Synchronous Digital Hierarchy*) com a tecnologia de multiplexação DWDM (*Dense Wavelength-Division Multiplexing*). São destacados os principais aprimoramentos em relação às outras tecnologias, as principais recomendações ITU-T que normatizam a OTN e sua estrutura de camadas que descrevem seu funcionamento. Por fim, é feita uma revisão bibliográfica a respeito dos trabalhos que auxiliam e servem como base para o desenvolvimento desta dissertação.

2.1 CONTEXTUALIZAÇÃO

Segundo (Ramaswami & Sivarajan, 2002), quando se fala sobre redes ópticas, normalmente se refere às duas gerações de redes ópticas tradicionais. Na primeira geração, as redes ópticas eram essencialmente usadas para transmissão e aumento da capacidade. A fibra óptica é um meio que provê baixa taxa de erro de *bits* e alta capacidade de transmissão em comparação com outras tecnologias. Todas as operações de comutação e outras funções inteligentes das redes dessa geração são feitas eletronicamente, a partir de uma conversão O-E-O em cada elemento de rede. Exemplos dessa primeira geração são o SONET, tecnologia usada principalmente na América do Norte, e o SDH, similar ao SONET e utilizado no Brasil, Europa e Ásia.

Alguns limites no que se refere ao crescimento de largura de banda surgiram. Com a multiplexação FDM (*Frequency Division Multiplexing*) utilizada na primeira geração de redes ópticas, aumentava-se a complexidade dos sistemas eletrônicos para a transmissão a altas taxas e, no mesmo instante, não aproveitava a capacidade óptica que a fibra provê. Nesse contexto surgiu a WDM (*Wavelength-Division*

Multiplexing), que consiste em transmitir dados simultaneamente em diferentes comprimentos de onda em uma mesma fibra óptica.

Com o avanço da tecnologia, funções mais complexas foram atribuídas às redes ópticas como comutação e roteamento na camada óptica da rede. Elementos como *Optical Add/Drop Multiplexers* (OADM's), *Reconfigurable Add/Drop Multiplexers* (ROADM's) e *Optical CrossConnects* (OXC's) surgiram para adicionar essas funcionalidades na camada óptica. Além de roteamento e comutação, funções de monitoramento e gerenciamento foram adicionadas nos níveis ópticos da rede. Essas novas funcionalidades caracterizam a segunda geração de redes ópticas.

2.2 REDES ÓPTICAS DE TRANSPORTE

Segundo (Iniewski, McCrosky, & Minoli, 2008), a OTN é uma tecnologia desenvolvida para ser a evolução das redes de núcleo. Ela apresenta os principais benefícios das redes SDH/SONET agregando a tecnologia WDM (*Wavelength-Division Multiplexing*) que permite a multiplexação de comprimentos de onda em uma fibra.

Uma das principais vantagens da tecnologia OTN é a possibilidade de incorporar algumas das funcionalidades de comutação e roteamento na parte óptica da rede, evitando a limitação elétrica para altas taxas de dados.

As redes OTN são compostas por um conjunto de ONEs (*Optical Network Elements*) interconectados, que são capazes de prover funcionalidades de transporte como: multiplexação, roteamento, gerenciamento, supervisão, e sobrevivência de canais ópticos, que carregam os sinais clientes.

Em (ECI Telecom, 2008) destacam-se os principais aprimoramentos trazidos com a OTN:

- Escalabilidade aprimorada: a OTN apresenta um esquema de multiplexação que pode transportar dados nativamente a taxas de 1.25Gbps, 2.5Gbps,

10Gbps, 40Gbps e 100Gbps (em fase de padronização) com uma quantidade menor de *overhead* se comparado à tecnologias anteriores;

- Transporte transparente do sinal cliente: sinais SDH/SONET são encapsulados diretamente dentro da OTN, enquanto que outros tráfegos de dados utilizam o GFP (*Generic Framing Procedure*). A OTN pode transportar de forma transparente vários tipos de dados, sem que haja terminação do sinal cliente em cada elemento de rede, tornando possível o transporte sem alterações no formato, taxa de *bits* e *clock* intrínsecos do sinal;
- Mecanismo aprimorado de *Forward Error Correction* (FEC): a OTN possui um mecanismo de FEC aprimorado, com um campo maior e específico, permitindo alcançar distâncias maiores de transmissão sem necessidade de regeneração do sinal, uma vez que o algoritmo é capaz de corrigir uma quantidade maior de erros de *bit*;
- Mais níveis de *Tandem Connection Monitoring* (TCM): Redes OTN provêm suporte a até seis níveis de TCM independentes, tornando possível o monitoramento de vários segmentos de caminhos em múltiplos domínios administrativos;
- Operação, Administração, Manutenção e Aprovisionamento (OAM&P): a rede OTN provê funções de operação, administração, manutenção e aprovisionamento, adaptando os mecanismos já existentes no SONET/SDH para as camadas elétricas da rede OTN.

As principais recomendações que definem a OTN e sua arquitetura de camadas são apresentadas nas seções a seguir.

2.2.1 Recomendações OTN

As várias características referentes à arquitetura, *frame*, gerenciamento e controle da OTN são definidas a partir de um conjunto de recomendações da ITU-T. A seguir são apresentadas as principais recomendações.

G.805 *(Generic Funtional Architecture of Transport)* – Como recomendação base para redes de transporte, a G.805 descreve a arquitetura funcional genérica de uma

rede de transporte do ponto de vista da capacidade de transferência da informação, isto é, a arquitetura funcional e estrutural das redes de transporte é descrita independente da tecnologia utilizada. (ITU-T, G.805, 2000).

G.872 (*Architecture of Optical Transport Network*) – Essa recomendação descreve a arquitetura funcional das redes ópticas de transporte usando as metodologias de modelagem descritas em (ITU-T, G.805, 2000). Nessa recomendação, a rede OTN é estruturada em camadas e, para cada camada, há descrição de suas funcionalidades, do tipo de informação que nela trafega e das associações entre as camadas do tipo cliente/servidor. (ITU-T, G.872, 2001).

G.709 *(Interfaces for the Optical Transport Network (OTN))* – A recomendação G.709 define as interfaces da rede OTN em termos da hierarquia de transporte óptico e digital, especificando a estrutura dos frames; a taxa de bits, os formatos para mapeamento dos sinais clientes, e os esquemas de multiplexação dos sinais ópticos e digitais. (ITU-T, G.709, 2009).

G.798 *(Characteristics of Optical Transport Network Hierarchy Equipment Functional Blocks)* – Apresenta um conceito amplo das características dos blocos funcionais dos equipamentos da OTN. É baseada nos requisitos apresentados nas recomendações ITU-T G.872 e ITU-T G.709. Essa recomendação descreve os blocos funcionais que compõem as funções de adaptação e de terminação de trilha das camadas OTN. (ITU-T, G.798, 2010).

G.874 *(Management aspects of the optical transport network)* – Essa recomendação aborda os aspectos do Gerenciamento das Redes Ópticas de Transporte. São apresentadas a arquitetura, os equipamentos funcionais e os requerimentos da gerência OTN. Nessa recomendação, encontra-se a lista dos parâmetros referentes às disciplinas da gerência: Falha, Configuração e Desempenho, para as Redes Ópticas de Transporte. (ITU-T, G.874, 2008).

G.8080 *(Architecture for the Automatically Switched Optical Network (ASON))* – Nessa recomendação é apresentada uma arquitetura de referência para o plano de controle de uma Rede Óptica de Transporte Comutada Automaticamente, descrevendo um conjunto de componentes do plano de controle para manipular os

recursos de uma rede de transporte provendo funcionalidades para criar, manter e terminar conexões. (ITU-T, G.8080, 2006).

G.7714 *(Generalized automatic Discovery for transport entities)* – Essa recomendação descreve o processo de descoberta automática para entidades de transporte, seus sub-processos e as interações básicas de um modo independente de protocolos. (ITU-T, G.7714, 2005).

G.7714.1 (*Protocol for automatic Discovery in SDH and OTN networks*) – Essa recomendação descreve os métodos, procedimentos e mecanismos de transporte para a descoberta de adjacências de camada descrita na recomendação ITU-T G.7714. (ITU-T, G.7714.1, 2010).

2.2.2 Arquitetura de Camadas OTN

Em (ITU-T, G.872, 2001) define-se a arquitetura de uma rede OTN, apresentando três camadas: OCh (*Optical Channel*), OMS (*Optical Multiplex Section*) e OTS (*Optical Transport Section*). As três juntas formam a Hierarquia Óptica de Transporte (OTH – *Optical Transport Hierarchy*). Em função da necessidade de tratar os sinais digitais, provendo multiplexação digital e monitoramento de caminhos, a OCh engloba três subcamadas formando a Hierarquia Digital de Transporte (DTH – *Digital Transport Hierarchy*). A DTH é composta por: OPU (*Optical Channel Payload Unit*), ODU (*Optical Channel Data Unit*) e OTU (*Optical Channel Transport Unit*), que a partir desse ponto são tratadas como camadas. Essa estrutura de camadas é observada na Figura 2.1.

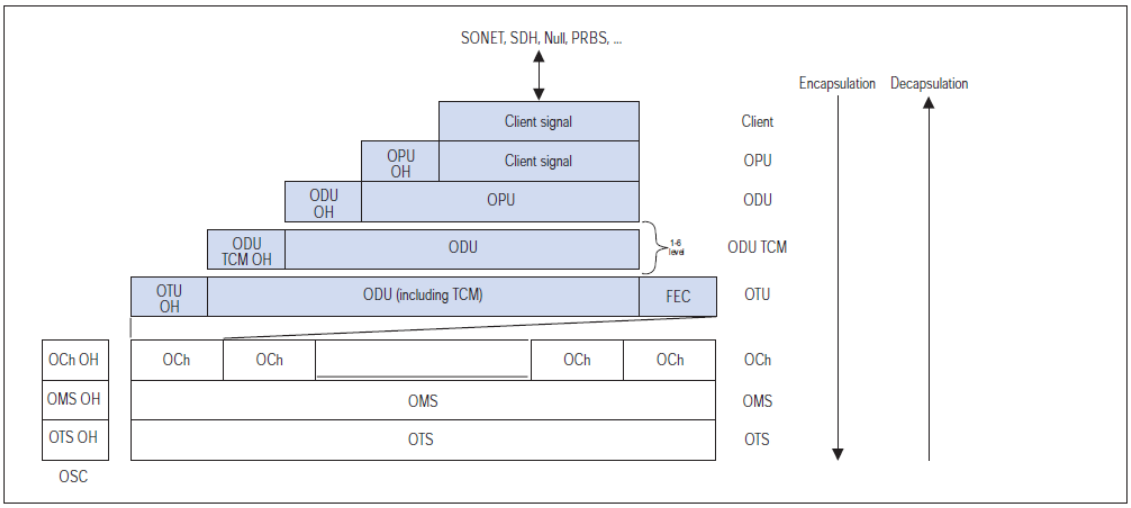

Figura 2.1 – Estrutura de camadas OTN. (EXFO, 2006).

A camada OCh é responsável por fornecer um caminho óptico para transportar o sinal cliente pela rede OTN. Esse caminho, caracterizado por um comprimento de onda, está entre duas terminações ópticas, uma na fonte realizando a conversão do sinal elétrico para óptico e outra no final da trilha realizando a conversão do sinal óptico para elétrico.

A camada OMS é responsável por multiplexar/demultiplexar diversos comprimentos de onda, cada um transportando um canal óptico em uma fibra.

A camada OTS é a responsável por gerar o canal de supervisão e transmiti-lo juntamente com o sinal multiplexado proveniente da camada OMS. Esse canal de supervisão é usado para transmitir as informações de cabeçalho das três camadas ópticas de um modo separada dos canais de dados. Ao contrário dos demais comprimentos de onda dos canais de dados, o canal de supervisão é processado em cada nó da rede.

A camada OPU é responsável por realizar o tratamento do sinal digital do cliente, encapsulando e efetuando as operações de justificação.

A camada ODU é responsável de prover as funcionalidades de transporte de um caminho digital para o cliente. É a camada que realiza a multiplexação TDM do sinal cliente, oferecendo proteção, supervisão de caminho fim a fim, monitoração de *Tandem Connection* e supervisão da qualidade do sinal através do BIP (*Bit Interleaved Parity*).

A camada OTU provê alinhamento de quadros e adiciona o código de correção de erros – FEC (*Forward Error Correction*). Cada OTU, por sua vez, é mapeado em um OCh.

2.3 TRABALHOS RELACIONADOS

Esta dissertação utiliza como base os trabalhos anteriores e em desenvolvimento pelo grupo de pesquisa na UFES. Destacam-se os trabalhos (Favoreto, 2009), (Tessinari, 2009), (Ferrari, 2009), (Ferrari, Frasson, & Garcia, 2010), (Frigini, 2010) e (Tessinari, 2011).

Em (Favoreto, 2009) são encontrados os primeiros trabalhos de implementação de um simulador de redes OTN. O principal objetivo é relacionar as recomendações do ITU-T para redes OTN com as RFCs (*Request for Comments*) do IETF para o plano de controle GMPLS. Nesse trabalho é apresentada a primeira implementação, simplificada, das camadas da hierarquia óptica da OTN. O foco desse trabalho é a representação dos recursos do plano de transporte no plano de controle e as principais extensões para a aplicação do GMPS na OTN.

O desenvolvimento de uma versão mais abrangente das camadas da hierarquia óptica da OTN ocorre em (Tessinari, 2009). Nesse trabalho são modeladas as funções de adaptação e terminação de acordo com a recomendação (ITU-T, G.798, 2010), e modelados alguns equipamentos ópticos.

Em (Frigini, 2010) é modelada a hierarquia digital da OTN. Esse trabalho completa as camadas do plano de transporte em relação à (Tessinari, 2009). Esses trabalhos ainda não comtemplam os aspectos de comunicação com gerência ou tratamento de falhas na rede.

Em (Ferrari, 2009) e (Ferrari, Frasson, & Garcia, 2010) é desenvolvida a primeira versão da funcionalidade de descoberta automática baseada na recomendação (ITU-T, G.7714, 2005). Nesses trabalhos são implementadas as trocas de mensagens utilizando identificadores simples e não integrado à rede OTN desenvolvida no simulador.

Em (Tessinari, 2011) é realizado um trabalho de integração dos trabalhos descritos nesta seção. Uma arquitetura de gerência é proposta para integração do plano de gerência e descoberta automática ao plano de transporte, elaborada em conjunto com esta dissertação.

3 O PROTOCOLO DE GERENCIAMENTO DE ENLACES – LMP

Devido ao crescimento das redes e ao uso do Plano de Controle para alocar recursos e prover proteção e restauração de modo dinâmico na rede, o gerenciamento dos enlaces e dos recursos existentes na rede se torna um procedimento essencial. Entre um par de nós é possível existir diversos enlaces de dados multiplexados por TDM ou WDM. Com propósitos de escalabilidade, um conjunto de enlaces de dados pode ser combinado em um único TE *Link* (*Traffic Engineering Link*).

Neste capítulo é apresentado o LMP (*Link Management Protocol*) (Lang, 2005) que é executado entre pares de nós com o propósito de gerenciar os *TE Links* existentes entre eles. O LMP é um protocolo inserido no contexto do GMPLS (*Generalized Multiprotocol Label Switching*) e dentre suas funções está a de gerenciar um canal de comunicação para dados de controle e gerência, gerenciar os recursos de enlaces de dados, verificação dos enlaces de dados e isolamento de falhas.

3.1 PROCEDIMENTOS PRINCIPAIS

O LMP é composto por dois procedimentos principais: o de gerenciamento do canal de controle e de correlação de propriedade de enlace. Esses procedimentos são descritos nas seções a seguir.

3.1.1 Gerência do Canal de Controle

A comunicação entre nós da rede, utilizada para troca de informações do plano de controle, é realizada através do canal de controle. Este é definido em (Lang, 2005) como sendo um par de interfaces mutuamente acessíveis que é usada para habilitar a comunicação entre nós para roteamento, sinalização e gerenciamento de enlace.

Esse procedimento do LMP visa estabelecer e manter o canal de controle na rede. O estabelecimento do canal de controle ocorre através da troca de parâmetros de configuração que são acordados entre o par de nós. Para a manutenção do canal, mensagens *Hello* precisam ser trocadas entre os nós para verificação de disponibilidade do canal.

O LMP usa três mensagens para configurar os canais de controle individualmente: *Config, ConfigAck e ConfigNack*.

Para iniciar um canal de controle, uma mensagem *Config* precisa ser enviada ao nó remoto e em resposta uma mensagem *ConfigAck* precisa ser transmitida do nó remoto ao nó local. A mensagem *Config* contém o identificador do canal de controle (CC_Id), o identificador de nó (*Node_Id*), o identificador de mensagem (*Message_Id*) e o objeto de configuração (*CONFIG*).

A mensagem *ConfigAck* é usada para confirmar o recebimento da mensagem *Config* e concordar com todos os parâmetros negociáveis e não negociáveis no objeto de configuração recebido. Já a mensagem *ConfigNack* é usada para confirmar o recebimento da mensagem *Config*, indicando que um ou mais parâmetros não negociável é inaceitável ou propor parâmetros alternativos para os valores negociáveis. Se um nó receber uma mensagem *ConfigNack* com parâmetros de configuração aceitáveis, ele deve retransmitir a mensagem *Config* com os novos valores propostos.

Para monitorar a disponibilidade do canal de controle são trocadas mensagens *Hello* contendo números de sequência de mensagens transmitidas e recebidas. Desse modo, pode-se verificar se o estado do canal de controle está em condições normais, em estado de degradação ou inativo.

3.1.2 Correlação de Propriedade de Enlace

As propriedades de enlaces são informações que: (1) identificam o enlace, ou seja, a identificação dos pontos que terminam o enlace, (2) caracterizam o sinal transmitido

nesse enlace, (3) acordam as capacidades de multiplexação e (4) apresentam as funcionalidades suportadas pelos nós que terminam o enlace.

A correlação dessas propriedades é a troca de informações entre os nós que terminam o enlace, com a finalidade de obter os dados referentes aos recursos e capacidades que ambos suportam, e, assim, as capacidades do enlace em si.

A correlação de propriedades de *TE Links* é realizada através da troca das mensagens *LinkSummary, LinkSummaryAck* e *LinkSummaryNack*.

Cada *TE Link* possui um identificador de enlace (*Link_Id*) que é atribuído a cada terminação do enlace*.* Da mesma forma acontece com os enlaces de dados, onde cada terminação do enlace recebe um identificador (*Interface_Id*) .

As mensagens *LinkSummary* são usadas para verificar a consistência de informação em ambos os lados do *TE Link* ou enlace de dados. Essas apresentam um objeto *TE_LINK* e um ou mais objetos *DATA_LINK.* O objeto *TE_LINK* identifica o *TE Link* e o *Link_Id* remoto. O objeto *DATA_LINK* é usado para identificar os enlaces de dados que estão contidos no *TE Link*, ele contém o *Interface_Id* remoto e local e pode conter sub-objetos para descrever as propriedades dos enlaces de dados, ou seja, as capacidades suportadas nesse enlace.

A mensagem *LinkSummaryAck* é usada para concordar com o mapeamento de *Interface_Ids* e as definições de propriedades de enlace. Por outro lado, a mensagem *LinkSummaryNack* é usada no caso do mapeamento de *Interface_Ids* não estar correto ou quando não há um acordo com as definições de propriedade de enlace. Caso a discordância seja em um objeto negociável, deverão ser incluídos na mensagem os parâmetros aceitáveis. O nó que originou a mensagem de *LinkSummary*, ao receber uma *LinkSummaryNack*, deverá responder novamente com uma *LinkSummary* contendo os parâmetros dentro dos limites aceitáveis informados na *LinkSummaryNack*.

3.2 PROCEDIMENTOS OPCIONAIS

O LMP apresenta dois procedimentos considerados opcionais: um procedimento para verificação de enlaces de dados e um para gerenciamento de falhas. Esses são descritos nas seções a seguir.

3.2.1 Verificação de Conectividade de Enlace

Esse procedimento é usado para verificação da conectividade física dos enlaces de dados e associações entre os identificadores de interfaces, é realizado no estabelecimento do TE *Link* e, periodicamente, para os enlaces de dados não alocados.

Uma vez que um canal de controle é estabelecido entre dois nós, esses podem iniciar a verificação de seus enlaces de dados. Para isso, o nó local envia uma mensagem de *BeginVerify* através do canal de controle, contendo informações de quais enlaces de dados serão verificados e os parâmetros envolvidos na verificação.

Se o nó remoto está preparado para a verificação, este deverá responder com uma mensagem *BeginVerifyAck* para o nó de origem, incluindo um identificador (*VerifyID*) único que será utilizado nas mensagens de testes dos enlaces de dados para diferenciar de outros procedimentos de testes em andamento. Nesse instante, o nó que originou o procedimento inicia o envio de mensagens *Test* através dos enlaces de dados. Esse envio pode ocorrer através de diferentes mecanismos e depende da tecnologia sendo utilizada, podendo ser transmitida pelo cabeçalho, enviando um pacote IP, etc.

Quando o nó remoto recebe as mensagens *Test* para cada enlace de dados, ele responde com mensagens *TestStatusSuccess* ou *TestStatusFailure* indicando o sucesso ou não da verificação de determinado enlace de dados. Essas mensagens também devem ser respondidas com a mensagem *TestStatusAck*. Uma vez terminados os testes, são trocadas mensagens de *EndVerify* e *EndVerifyAck*, e

ambos os nós podem efetuar a correlação e conhecer o mapeamento de interfaces entre os dois nós.

3.2.2 Gerenciamento de Falhas

O mecanismo de gerenciamento de falhas é um procedimento opcional oferecido pelo LMP que permite gerenciar falhas pela rápida notificação do status de um ou mais enlaces de dados de um TE *Link.* O escopo desse procedimento é interno ao TE *Link*, e por isso pode ser negociado entre as partes pela mensagem *LinkSummary.* O objetivo desse procedimento é isolar rapidamente falhas nos enlaces de dados que formam um TE *Link.*

Em algumas situações, quando há uma falha no enlace entre dois nós, o alarme é propagado como se todos os demais nós tenham detectado as falhas, sem antes localizá-la. Para evitar múltiplos alarmes para a mesma falha, o LMP comunica a falha pela mensagem *ChannelStatus.* A mensagem pode indicar que um enlace de dados falhou, vários enlaces de dados falharam ou até mesmo que o TE *Link* inteiro falhou. Assim a correlação da falha é feita localmente em cada nó tão logo a notificação do evento de falha seja recebida.

3.3 MÁQUINAS DE ESTADOS

O LMP tem seu funcionamento caracterizado por três máquinas de estados que descrevem seus elementos principais: canal de controle, *TE Link* e Enlace de Dados. Para cada máquina de estados são definidos seus estados e eventos que causam as transições de estados. Cada estado possui certo comportamento até que um determinado evento ocorra. Na seção seguinte é apresentada a máquina de estados do *TE Link*. Para informações específicas deve-se consultar (Lang, 2005).

As máquinas de estados do Canal de Controle e do Enlace de Dados são apresentadas no ANEXO I – Máquina de Estados do LMP - Canal de Controle e no ANEXO II – Máquina de Estados do LMP - Enlace de Dados, respectivamente.

3.3.1 FSM do *TE Link*

Cada *TE Link* pode estar em um dos estados definidos a seguir. Os seguintes estados são definidos em (Lang, 2005):

- *Down*: Não há nenhum enlace de dados alocado para o *TE Link* em questão;
- *Init*: Os enlaces de dados foram alocados para o *TE Link* em questão, deve-se iniciar o envio periódico da mensagem *LinkSummary* ao seu vizinho;
- *Up*: É o estado de operação normal do *TE Link*. Ao menos um canal de controle precisa estar ativo entre os dois nós. As mensagens *LinkSummary* são periodicamente enviadas ao vizinho;
- *Degraded*: Nesse estado, todos os canais de controle estão inativos, mas o *TE Link* ainda continua com enlaces de dados que estão alocados para tráfego de usuários.

A Figura 3.1 mostra o funcionamento da máquina de estados do *TE Link*, indicando os eventos que fazem a transição dos estados.

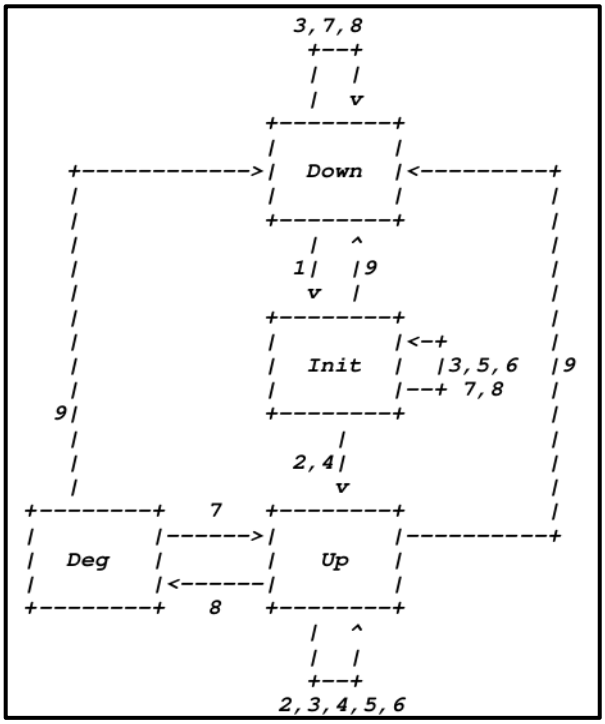

Figura 3.1 - FSM do *TE Link*.

A operação da máquina de estados do TE *Link* é definida em função dos estados e dos eventos. Os eventos do TE *Link* são gerados por rotinas de processamento de

pacotes, pela FSM do canal de controle associado aos enlaces de dados. Os possíveis eventos são descritos a seguir:

1. *evDCUp:* Evento que indica que um ou mais enlaces de dados foram atribuídos ao TE *Link;*
2. *evSumAck:* Evento que indica que a mensagem *LinkSummary* foi recebida e confirmada positivamente;
3. *evSumNack:* Evento que indica que a mensagem *LinkSummary* foi recebida e confirmada negativamente;
4. *evRcvAck:* Evento que indica que a mensagem de *LinkSummaryAck* foi recebida confirmando a configuração do *TE Link;*
5. *evRcvNack:* Evento que indica que uma mensagem *LinkSummaryNack* Foi recebida;
6. *evSumRet:* Evento que indica que o tempo de retransmissão foi atingido e uma mensagem *LinkSummary* será retransmitida;
7. *evCCUp:* Evento que indica que o primeiro canal de controle ativo foi estabelecido;
8. *evCCDown:* Evento que indica que o ultimo canal de controle ativo foi desativado;
9. *evDCDown:* Evento que indica que o ultimo enlace de dados foi removido.

3.4 O LMP APLICADO À OTN

A OTN sofreu novos progressos com as últimas recomendações do ITU-T. Novos quadros ODU foram adicionados (ex. ODU0, ODU4, ODU2e e ODUflex) e uma nova granularidade de *Tributary Slot* (TS) (1,25Gbps) foi introduzida. Com essa evolução, surgem problemas de compatibilidades que devem ser considerados. Equipamentos que trabalham com TS de 1,25Gbps podem combinar TS específicos para poderem trabalhar com equipamentos que suportam apenas 2,5Gbps.

Sinais de alta ordem (HO ODU) podem suportar diferentes sinais de baixa ordem (LO ODU). Um equipamento em determinado nó da rede que termina uma HO ODU pode não suportar os mesmos tipos de LO ODU de outro nó da rede. Logo, esse

enlace de HO ODU não pode ser usado para transportar certos tipos de sinas LO ODU.

Do ponto de vista do canal de controle, é necessário descobrir o tipo de TS suportado por cada terminação e os tipos de LO ODU suportado por uma HO ODU entre dois nós para então, ter as informações necessárias para a alocação de recursos de modo correto.

Em (Zhang, et al., 2011) são definidos novos objetos de mensagens e descrito o procedimento para a troca de informações referentes ao suporte de LO ODU e tipos de TS por cada nó. Esse *draft* do IETF é utilizado nesta dissertação para a implementação do protocolo LMP no contexto da rede OTN.

3.5 CONSIDERAÇÕES FINAIS

Neste Capítulo é apresentado de forma breve o protocolo de gerenciamento de enlaces do GMPLS: o LMP. São descritos os procedimentos (Seções 3.1 e 3.2) e estados operacionais (Seção 3.3) dos principais elementos gerenciáveis (Canal de Controle, *TE Link* e o Enlace de Dados).

Na Seção 3.4 é abordado o uso do LMP em redes OTN de acordo com (Zhang, et al., 2011). Este *draft* do IETF apresenta uma extensão ao LMP para oferecer suporte às questões descritas nesta seção.

4 DESENVOLVIMENTO

Ao se implementar comportamentos de equipamentos e protocolos descritos em recomendações da ITU-T e RFCs da IETF deve-se tomar decisões em alguns pontos que não são abordados ou especificados pelas normas. Os trabalhos relacionados à implementação e simulação de redes OTN envolve os planos de transporte, controle e gerência da rede. Portanto, deve-se modelar uma arquitetura para integração dos planos.

Este capítulo tem como objetivo apresentar a proposta de arquitetura para o plano de gerência desenvolvida juntamente com o trabalho (Tessinari, 2011), apresentar as melhorias e integração realizadas na funcionalidade de Descoberta Automática apresentada inicialmente em (Ferrari, 2009) e (Ferrari, Frasson, & Garcia, 2010), e, por fim, apresentar as principais questões envolvidas na proposta da criação de um módulo para gerenciamento de enlaces na rede OTN baseado na implementação do protocolo LMP, descrito no Capítulo 3.

Nas seções seguintes são apresentadas as arquiteturas utilizadas, a funcionalidade de descoberta automática e sua integração com outros planos da rede OTN e o gerenciamento de enlaces na rede OTN.

4.1 PLANO DE GERÊNCIA

Inicialmente, verificou-se a necessidade de elaboração de uma proposta de arquitetura para integração entre os planos de gerência e transporte da rede. Essa necessidade é devida às recomendações não apresentarem de forma detalhada uma proposta de arquitetura de gerência a ser utilizada com o plano de transporte OTN, e devido à integração das funcionalidades desenvolvidas. Na Figura 4.1 a seguir observa-se a comunicação da gerência com os planos de controle e de transporte proposta em conjunto com (Tessinari, 2011). Observa-se a necessidade de propor uma arquitetura que possibilite a comunicação entre os elementos de rede do plano de transporte e os controladores. Portanto, propõe-se o Gerente de NE

(GNE) para cada ONE e os Gerentes Locais para os equipamentos interligados em cada par de fibras. Estes são detalhados a seguir.

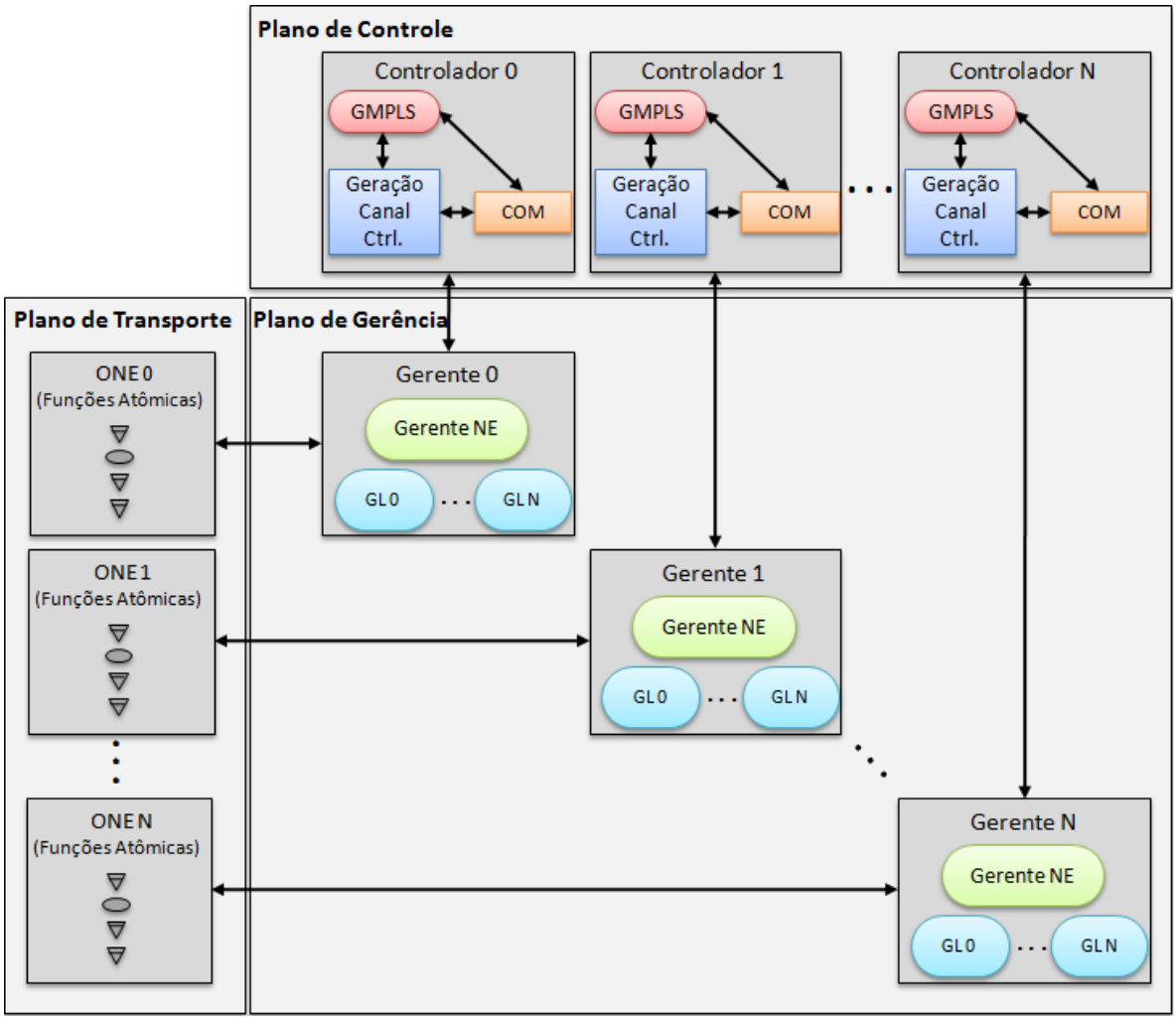

Figura 4.1 - Arquitetura proposta, realçando o relacionamento entre os diferentes planos (Tessinari, 2011).

Esta proposta leva em consideração a abordagem de muitos fabricantes de equipamentos de redes OTN que utilizam uma placa ou dispositivo em seu ONE com a função de supervisionar as demais placas. Essa placa é responsável pela verificação do estado operacional das demais, pela troca de dados referente ao funcionamento das mesmas para estatísticas, e pelo gerenciamento dos alarmes e informações importantes que devem ser reportadas para a Gerência da rede.

A arquitetura proposta é mais bem observada na Figura 4.2. A figura mostra os componentes da gerência e suas ligações (em vermelho) com os AF (*Atomic Functions*), que representam as adaptações e terminações das camadas, e com o

bloco de Descoberta Automática. As AFs representam as funções atômicas contidas nas camadas OTN, composta pelas funções de adaptação e terminação destas. A DCN representa a rede de gerência que interliga os nós que compõem a rede OTN.

Nas próximas seções são abordados os componentes da arquitetura de gerência e o componente de Descoberta Automática.

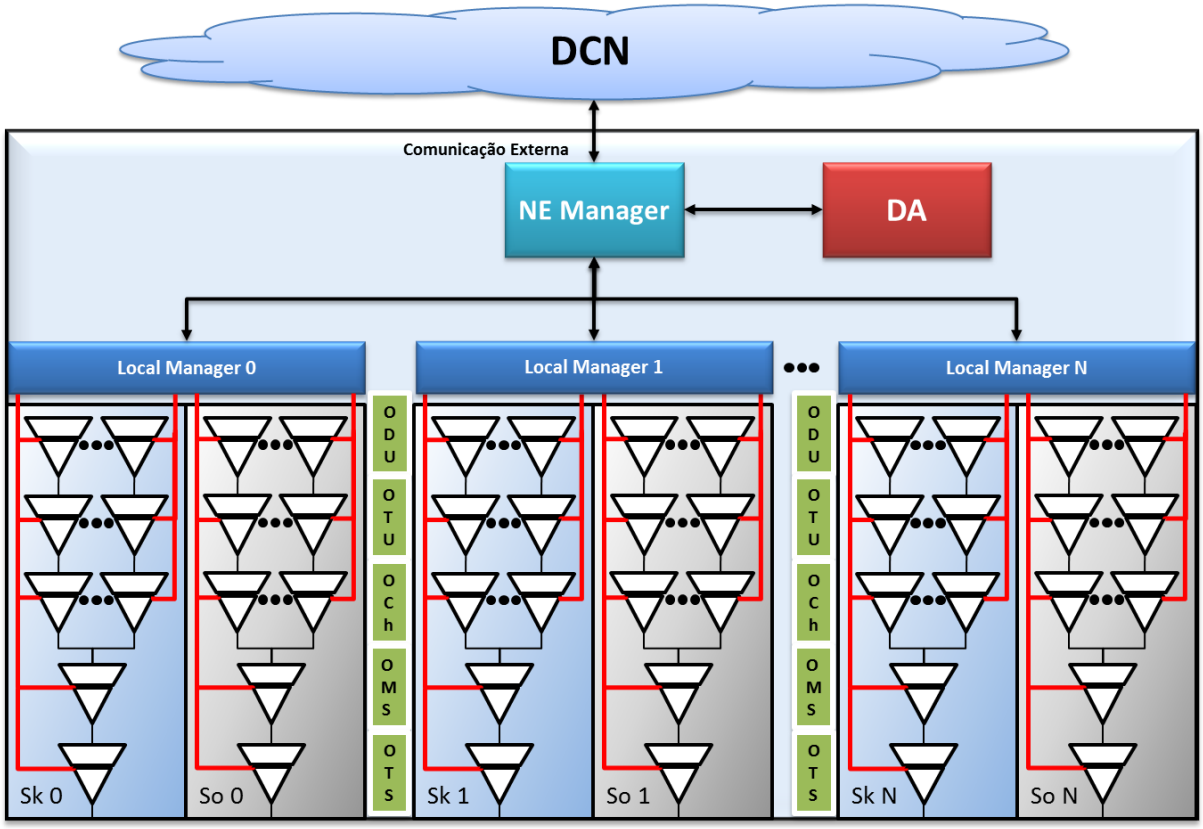

Figura 4.2 – Proposta de arquitetura para plano de gerência.

Na arquitetura proposta,Figura 4.2, podem-se destacar dois componentes fundamentais: o Gerente Local (*Local Manager*) e o Gerente de NE (*NE Manager*).

As funcionalidades dos blocos gerenciais desta arquitetura também abrangem de modo simplificado o modelo proposto na recomendação ITU-T G.874. As correspondências desta arquitetura com a referida recomendação pode ser encontrada em (Tessinari, 2011).

Para cada par de fibras (*Source e Sink*) no ONE (*Optical Network Element*), existe um Gerente Local. Este exerce as funcionalidades da gerência de falhas e gerência

de configuração dos componentes de rede. As demais gerências (contabilidade, desempenho e segurança) não fazem parte do escopo deste trabalho.

As principais funções do bloco Gerente Local são:

- Trocar informações de identificação com todos os componentes funcionais de Adaptação e Terminação (AFs) conectadas a ele, mantendo o inventário de componentes;
- Comunicar-se com os componentes de rede e trocar informações de gerenciamento;
- Gerenciar o estado atual de todos os alarmes dos componentes;
- Reportar eventuais mudanças de alarmes ao Gerente de NE;
- Enviar informações que devem ser inseridas nos cabeçalhos das camadas OTN, como TTI (*Trail Trace Identifier*).

O componente Gerente de NE é o componente responsável por agrupar todas as informações recolhidas pelos Gerentes Locais e efetuar a comunicação com qualquer componente externo. O Gerente de NE é único no ONE e seu identificador é utilizado como identificador do ONE, sendo único em toda a rede.

Suas principais funções são:

- Manter o inventário de Gerentes Locais existentes no NE;
- Efetuar a comunicação com outros componentes do plano de gerência via DCN;
- Efetuar a comunicação com o plano de controle para troca de informações importantes entre os planos;
- Efetuar a comunicação com o componente de Descoberta Automática, transmitindo mensagens e configurando o cabeçalho das camadas;
- Geração de *logs* de eventuais mudanças identificadas pelos Gerentes Locais do ONE.

Através do uso em conjunto dos Gerentes Local e de NE é possível manter informações sobre falhas e configurar os componentes que compõem a rede OTN. Como cada Gerente Local possui em seu escopo apenas um par de fibras, ele é

responsável por todas as camadas até esse par de fibras sabendo, por exemplo, em qual fibra cada cliente da hierarquia digital está conectada, informação importante para o gerenciamento de falhas na rede.

4.2 INTEGRAÇÃO DA DESCOBERTA AUTOMÁTICA

Desde os trabalhos (Ferrari, Projeto de Graduação - Simulação da Funcionalidade de Descoberta Automática Aplicada às Redes OTN, 2009) e (Ferrari, Frasson, & Garcia, 2010) foram efetuadas modificações do modelo da Descoberta Automática para integração com os planos de gerência e transporte da OTN. As principais alterações são abordadas nas seções a seguir.

4.2.1 Modularização dos Processos da ITU-T G.7714

O módulo de Descoberta Automática, primeiramente desenvolvido em (Ferrari, 2009), foi modularizado e sofreu alterações para simplificação e melhor integração com a gerência OTN. A arquitetura do DA (*Discovery Agent*) é observada na Figura 4.3, contendo seus dois módulos principais: DT (*Discovery Trigger*) e LAD (*Layer Adjacency Discovery*). Esses módulos executam os processos de mesmo nome descritos na recomendação G.7714 (ITU-T, G.7714, 2005).

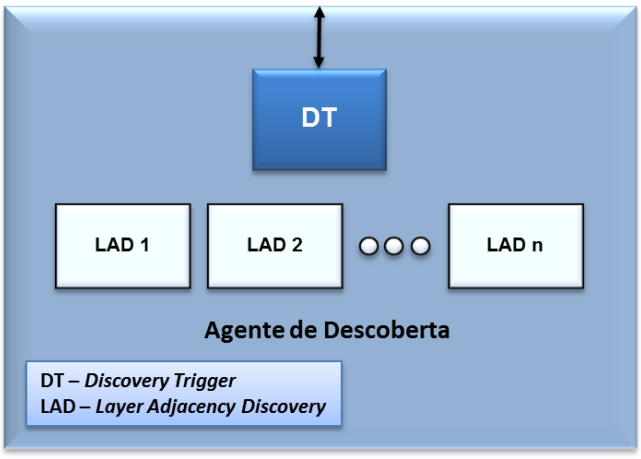

Figura 4.3 – Arquitetura do Agente de Descoberta.

O componente DA na arquitetura de gerência, Figura 4.2, tem como função principal executar o protocolo descrito nas recomendações G.7714 (ITU-T, G.7714, 2005) e G.7714.1 (ITU-T, G.7714.1, 2010) e explicado em (Ferrari, 2009). Cada ONE possui um componente DA que é responsável por identificar os pontos de terminação das camadas ODU e OTU na rede OTN. Utilizando-se de uma combinação de identificadores do NE e TCP (*Termination Connection Point*) o DA cria uma base de dados de conexões que será utilizada pelo plano de gerência e controle da rede.

O componente DT desta arquitetura coordena os componentes LAD da simulação e se comunica com o Gerente de NE. Para cada terminação de camada ODU ou OTU no NE o DT cria um processo LAD correspondente e direciona as devidas mensagens entre o LAD e a gerência. O DT contém a base de informações de todas as conexões descobertas e o estado atual do procedimento de descoberta em cada TCP. Essas informações são reportadas ao Gerente de NE e posteriormente ao plano de controle, onde serão utilizadas pelo LMP (ver Seção 4.3.1). O DT também gera um arquivo com as informações dos enlaces descobertos e seu estado atual (Enlace Correto, com Erro ou Perdido).

O componente LAD é o responsável pela descoberta em cada TCP do NE. Este componente é baseado nas máquinas de estados contidas na recomendação G.7714.1 (ITU-T, G.7714.1, 2010). Seu funcionamento consiste basicamente em enviar mensagens contendo identificadores locais pelo canal de dados, utilizando o cabeçalho da camada, e receber a resposta do NE remoto, contendo os identificadores do ponto que recebeu a mensagem original, através da rede de gerência (DCN – *Data Communication Network*). Com esse processo executado por ambos os NEs, é possível realizar a correlação dos dados e identificar possíveis erros de conexão.

4.2.2 Uso de IP e criação da Rede DCN

Outra modificação importante na simulação é o uso de endereços IP para identificar todos os NE´s na rede e a implementação de uma rede de gerência IP – DCN. Em (Ferrari, 2009) e (Ferrari, Frasson, & Garcia, 2010) a Descoberta Automática utiliza

identificadores inteiros para obter as conexões. Contudo, as mensagens de respostas, que deveriam trafegar por uma rede de gerência ou controle de modo *out-of-band*, são enviadas de forma direta entre os blocos de simulação.

A implementação de endereçamento IP e a criação da rede DCN (*Data Communication Network)* se justifica pela sua utilização nas redes atuais de diversos fabricantes e na própria recomendação, que cita a rede DCN utilizando IP. A implementação torna a simulação um modelo mais fiel ao que é praticado no mercado, possibilitando testes e análises mais reais. Para utilizar endereços IP na rede de simulação são utilizados os módulos do INET (Varga & Acadêmica, INET Framework main page, 2011). Através deste, cada NE passou a ter um endereço IP, o qual é utilizado pelo DA para envio das mensagens de identificação. Nesta dissertação, a mensagem de resposta é encapsulada em UDP e roteada através da rede DCN de acordo com o especificado nas recomendações ITU-T.

A rede DCN implementada nesta dissertação consiste em uma rede IP entre os nós da rede que utiliza os canais de serviço de cada fibra para transportar as mensagens. Essa abordagem atende os requisitos desta dissertação, porém, a DCN também pode ser estabelecida por um meio físico separado do plano de dados.

4.2.3 Integração com os Planos de Gerência e Transporte da OTN

Na recomendação G.7714.1 (ITU-T, G.7714.1, 2010) define-se o método de transporte da mensagem de descoberta na rede OTN. Os seguintes mecanismos podem ser utilizados:

- OTUk layer: O *byte* da seção de monitoramento SM (*Section Monitoring*) e o GCC0 (*General Communication Channel*) podem ser usados para suportar a descoberta das adjacências da OTUk. Especificamente, o campo SAPI (*Source Access Point Identifier*) contido no campo SM é usado para carregar a mensagem de descoberta;
- ODUk layer: O *byte* do caminho de monitoramento PM (*Path Monitoring*) e o GCC-1 e GCC-2 pode ser usado para suportar o descoberta das adjacências

da ODUk. Especificamente, o campo SAPI contido no campo PM é usado para carregar a mensagem de descoberta.

Os campos do SAPI estão inseridos no TTI (*Trail Trace Identifier*), que carregam informações de identificação da trilha. O campo do SAPI dispõe de 16 *bytes* que podem ser utilizados para o transporte da mensagem de descoberta pela trilha.

Com a integração da descoberta ao plano de dados da rede OTN, uma codificação da mensagem de descoberta para inserção no cabeçalho das terminações de camadas a serem descobertas se torna necessária. Nesta dissertação, o componente DT do DA, exerce a função de codificar/decodificar as mensagens de descoberta e encaminhá-la ao Gerente de NE. A mensagem de descoberta, composta basicamente pelo IP do ONE e identificador do TCP em questão, é transformada em um conjunto de caracteres que serão inseridos no campo SAPI da terminação correspondente. O caractere inicial (“+”) distingue a mensagem de descoberta do uso padrão do campo SAPI, desse modo, a terminação pode identificar quando se trata de uma mensagem de descoberta e não disparar qualquer tipo de alarme por falha nesse campo. Ao identificar essa mensagem de descoberta, a terminação a reporta para a gerência (Gerente de NE) que encaminhará a mensagem até o DA.

Nesta dissertação, além do uso especificado na recomendação (ITU-T, G.7714.1, 2010) para as camadas ODU/OTU da rede OTN, utiliza-se desse procedimento para efetuar a descoberta automática dos enlaces de fibra da camada OTS. A camada OTS é caracterizada por processar o canal de serviço óptico, isso é feito em todos os ONEs da rede, e seu cabeçalho possui campos de TTI semelhantes ao das camadas ODU/OTU. Através desses campos, são transmitidas mensagens de descoberta para identificar enlaces de fibra, ou seja, da camada OTS.

O Gerente de NE exerce sua funcionalidade de configurar os campos de cabeçalho dos componentes da rede, para inserir a mensagem de descoberta no campo SAPI das terminações de camadas que estão em procedimento de descoberta.

O Gerente de NE também possui a capacidade de habilitar ou desabilitar o procedimento de descoberta, sendo responsável por identificar as terminações de camadas ODU e OTU no ONE e ativar o componente DA para cada terminação.

Outra funcionalidade do Gerente de NE é a comunicação de gerência com outros ONEs via DCN. As mensagens do procedimento de descoberta que precisam ser entregues a outro ONE são transmitidas através da DCN para o endereço IP de destino, descoberto pela mensagem vinda do canal de dados.

4.3 LMP EM REDES OTN

O LMP, como já descrito no Capítulo 3, é um componente do GMPLS para gerenciamento dos enlaces de dados da rede e do canal de controle. Nesta dissertação desenvolveu-se um módulo de gerenciamento de enlaces para aplicação nas redes OTN. O objetivo desse módulo é descobrir e acordar as informações referentes à capacidade dos enlaces de dados ODU. Essas informações são utilizadas pelo plano de controle para alocação de enlaces de dados. As principais funcionalidades do módulo do LMP aplicado à rede OTN são descritas nas próximas seções.

4.3.1 Utilização de Dados da Descoberta Automática

Para a execução dos procedimentos do LMP é necessário que cada nó conheça o endereço IP de seu nó vizinho. O procedimento de verificação de conectividade de enlace do LMP corresponde ao processo de descoberta das conexões dos enlaces de dados, porém, esse processo só é executado após o estabelecimento do canal de controle e, como descrito no Capítulo 3, esse processo envia uma mensagem ao seu vizinho para indicar a intenção de iniciar um teste nos enlaces de dados.

Nesta dissertação é utilizado o procedimento de descoberta baseado na recomendação (ITU-T, G.7714.1, 2010). Este procedimento é executado sem qualquer conhecimento anterior de endereço de nós vizinhos. Após a execução do

procedimento de Descoberta Automática o DA do ONE descobre todas as suas conexões de dados ODU e OTU e, portanto, descobre o IP de seus nós vizinhos na rede OTN.

A Descoberta Automática é executada assim que a rede entra em operação e suas informações de descoberta são encaminhadas através do Gerente de NE ao plano de controle da rede, mais especificamente para o módulo do LMP. Para cada TCP descoberto o DA passa as informações do ponto local e remoto para o LMP, um exemplo das informações passadas de cada ponto pode ser observada na Tabela 4.1. O LMP, ao receber as informações descobertas a respeito de seus vizinhos, inicia sua execução estabelecendo o canal de controle com cada vizinho descoberto.

Tabela 4.1 – Exemplo de dados transferidos do DA para o LMP.

NE Local	TCP Local	NE Remoto	TCP Remoto	Camada
10.0.0.1	1	10.0.0.2	1	ODU
10.0.0.1	3	10.0.0.3	5	ODU

A integração entre o procedimento de Descoberta e o LMP evita qualquer configuração manual para identificação de nós vizinhos na rede. Desse modo, o processo de Descoberta complementa o LMP para uma maior automatização dos processos.

4.3.2 Estabelecimento e Manutenção do Canal de Controle

Para o funcionamento dos protocolos do GMPLS é necessária a existência de um canal de comunicação IP entre os nós da rede. Esse canal é estabelecido e supervisionado pelo LMP que configura e mantém o canal monitorado para detecção de qualquer falha ou indisponibilidade desse canal.

Conforme descrito na seção anterior, o LMP utiliza as informações da Descoberta Automática para obter os endereços IP de seus vizinhos e, assim, estabelecer o canal de controle. O estabelecimento ocorre de acordo com o processo descrito na

Seção 3.1.1. São configurados e acordados valores para temporização de mensagens *Hello* e inicializados os números de sequência das mensagens. Ao final de processo, se ocorrido com sucesso, o estado do canal é colocado como ativo e o LMP e demais protocolos podem utilizar o canal para trocar mensagens de controle da rede.

4.3.3 Criação dos *TE Links*

O *Traffic Engineering* (TE) *Link* é definido como a entidade que representa um recurso ou agrupamento de recursos para propósitos de roteamento. Como o objetivo principal desta dissertação é fornecer aos demais processos do plano de controle e gerência informações sobre as conexões existentes e suas capacidades, não foi dado enfoque à estratégia para agrupamento de enlaces de dados em *TE Links*. No entanto, esse trabalho deve ser realizado de acordo com a aplicação do plano de controle GMPLS na rede OTN. De acordo com o modo de alocação de recursos pode-se escolher a melhor estratégia para agrupamento dos enlaces em *TE Links*.

Nesta dissertação é utilizada a versão atual do plano de transporte desenvolvida pelos demais pesquisadores do Labtel, principalmente nos trabalhos (Tessinari, 2009) e (Frigini, 2010). Porém, essa versão ainda não dispõe de uma implementação da funcionalidade de multiplexação em nível de ODU, portanto, o plano de controle não está ativo para reservar/alocar enlaces de dados nessa camada.

Devido a essas considerações, nesta dissertação o *TE Link* é criado como o agrupamento dos enlaces de dados ODU descobertos para cada vizinho, ou seja, existe um *TE Link* para cada vizinho, contendo um ou mais enlaces de dados ODU.

Referente à comutação de comprimentos de onda na camada OCh, nesta dissertação são utilizadas as informações de descoberta automática de OTS para gerar os dados de enlaces OCh. Ao identificar um enlace de fibra óptica (OTS), o LMP cria um *TE Link* referente a essa fibra, e cadastra enlaces de dados de acordo

com a quantidade de comprimentos de onda que o nó oferece suporte. Desse modo, cada fibra corresponde à um *TE Link*, que por sua vez representa o conjunto de comprimentos de onda (enlaces de dados) suportados por aquela terminação.

4.3.4 Descoberta de Capacidade entre Adjacências LMP

A descoberta de capacidade dos enlaces de dados é muito importante tanto para os demais procedimentos do plano de controle quanto para o plano de gerência. Através da base de dados de enlaces criada a partir do LMP, e manipulada pelos outros protocolos para alocação de enlaces, pode-se verificar a disponibilidade de enlaces e banda entre os nós. Para o plano de gerência, é possível obter os dados de conexões através da Descoberta Automática e as informações referentes à capacidade desses enlaces através da troca de informações providas pelo LMP. Para os protocolos do plano de controle são disponibilizados os dados necessários para a escolha do melhor enlace a ser alocado de acordo com suas características de granularidade e capacidade de multiplexação.

Para obter essas informações de capacidades o LMP executa o procedimento descrito na Seção 3.1.2. Esse procedimento é implementado nesta dissertação.

A aplicação do LMP nas redes OTN foi baseada no *draft* recente do IETF (Zhang, et al., 2011). Para trocar os dados de capacidade para a rede OTN, criaram-se novos objetos que compõem as mensagens de *LinkSummary*. Esses objetos, definidos em (Zhang, et al., 2011), contém as informações de identificação da ODU de alta ordem (ODU1, ODU2, ODU3 ou ODU4) e é enviada na mensagem através do campo HOODUk, do modo apresentando na Tabela 4.2.

Tabela 4.2 – Identificação dos Sinais ODU na mensagem *LinkSummary*.

HOODUk	Tipo de Sinal
0	Reservado
1	ODU1
2	ODU2
3	ODU3
4	ODU4
5-15	Reservado

4.3.4.1 Informações sobre Granularidade

A granularidade suportada de uma terminação ODU de alta ordem precisa ser identificada entre os pontos finais de um enlace. Caso uma terminação suporte *Tributary Slot (TS)* de 1,25 Gbps, pode-se combinar os TS e trabalhar com terminações que suportam TS de 2,5 Gbps. Logo, as terminações precisam acordar os tipos de TS que podem ser utilizados por determinado enlace ODU. Essas informações se tornam importantes no momento que o plano de controle reserva os recursos de TS corretos para o enlace.

A Tabela 4.3 mostra como é identificado o tipo de TS de uma terminação.

Tabela 4.3 – Identificação da granularidade suportada pela terminação ODU.

TS	Tipo de TS
0	1,25 Gbps
1	2,5 Gbps
2-3	Reservado

4.3.4.2 Informações sobre Capacidade de Multiplexação

A capacidade de multiplexação da terminação ODU de alta ordem (HO – *High Order*) é outra informação que é transmitida dentro do objeto para acordar a capacidade de um enlace de dados. Os tipos de sinais de baixa ordem (LO – *Low Order*) que são suportados são transmitidos dentro do objeto através de *Flags* que são configuradas como ‘1’ quando:

- *Flag* A: LO ODU0 é suportada;
- *Flag* B: LO ODU1 é suportada;
- *Flag* C: LO ODU2 é suportada;
- *Flag* D: LO ODU3 é suportada;
- *Flag* E: LO ODU4 é suportada;
- *Flag* F: LO ODU2e é suportada;
- *Flag* G: LO ODUflex é suportada.

Quando a *Flag* corresponde ao próprio sinal de alta ordem, esta é colocada como ‘1’, para indicar um mapeamento direto, sem multiplexação. Ambos os nós verificam as capacidades em comum para então disponibilizar os sinais suportados pelo enlace de dados em questão.

4.4 CONSIDERAÇÕES FINAIS

Na Seção 4.1 é apresentada uma proposta de arquitetura de plano de gerência, desenvolvida juntamente com (Tessinari, 2011), indicando as principais funções dos Gerentes e como esses interagem com o plano de transporte e o Agente de Descoberta.

A Seção 4.2 aborda o componente de Descoberta Automática, desenvolvido com base nas recomendações (ITU-T, G.7714, 2005) e (ITU-T, G.7714.1, 2010). São apresentadas as principais melhorias realizadas, a partir do trabalho em (Ferrari, 2009), para a integração do Agente de Descoberta com os planos de transporte e gerência da rede OTN.

Na Seção 4.3 são apresentadas as principais questões envolvidas no desenvolvimento de uma solução para gerenciamento de enlaces para redes OTN baseada no LMP, assim como a integração com o DA e os seus procedimentos principais de gerenciamento de canal de controle e correlação de propriedade de enlace.

No Capítulo 5 são discutidos os detalhes de implementação das arquiteturas e procedimentos aqui apresentados.

5 IMPLEMENTAÇÃO

Neste capítulo são apresentados os principais aspectos referentes à implementação dos componentes da arquitetura proposta para o plano de gerência, da Descoberta Automática integrada à rede OTN e da proposta de aplicação do LMP para efetuar o gerenciamento de enlaces da rede.

5.1 O SIMULADOR OMNET++

Nesta dissertação utilizou-se o simulador OMNeT++. Este simulador vem sendo utilizado nos diversos trabalhos desenvolvidos pelo grupo de pesquisa do LabTel citados nesta dissertação, e vem atendendo ao seu propósito, portanto, a escolha dessa ferramenta se mostra a mais adequada. O simulador OMNeT++ apresenta todas as ferramentas necessárias para alcançar os objetivos desta dissertação, possibilitando a implementação e testes dos protocolos nas redes OTN simuladas e apresentando os métodos necessários para validação das funcionalidades desenvolvidas. Uma comparação com outras ferramentas de simulação e a justificativa pela escolha do OMNeT++ pode ser encontrada em (Favoreto, 2009).

Segundo (Varga), o OMNET++ é um *framework* de simulação de redes de eventos discretos, orientado a objetos e modular. É desenvolvido em C++, com uma *Integrated Development Enviroment* (IDE) no Eclipse e um ambiente gráfico de execução. Por possuir uma arquitetura genérica, o OMNeT++ vendo sendo utilizado em várias áreas como: sistemas distribuídos, validação de arquiteturas de *hardware*, modelagem de redes de comunicação, modelagem de protocolos, entre outros.

A arquitetura do OMNET++ consiste em uma hierarquia de módulos que se comunicam através da troca de mensagens. Os módulos atômicos são chamados de *módulos simples* e quando agrupados formam os *módulos compostos*. Cada módulo simples tem seu comportamento descrito na linguagem C++, utilizando as classes de simulação da biblioteca provida pelo OMNET++. Entende-se por comportamento as ações realizadas pelo módulo ao ser inicializado e ao receber uma mensagem.

As mensagens trocadas entre os módulos podem representar *frames* ou pacotes em uma rede de computadores, tarefas numa fila ou qualquer outro tipo de entidades móveis. Mensagens podem conter estruturas de dados complexas e serem enviadas para outros módulos, através de portas. As mensagens também podem ser enviadas para o próprio módulo que as criaram, funcionando como sinalização interna do módulo. Para cada arquivo de mensagens (.msg) definido no OMNeT++, são geradas as classes correspondentes e seus arquivos .cc e .h.

Portas são as interfaces de entrada e saída dos módulos. As portas são interligadas por canais de conexões. Esses canais possuem parâmetros como *propagation delay* (atraso na propagação entre canais), *bit error rate* (probabilidade de erro de bit) e *data rate* (taxa de transmissão em bits/s).

As topologias das redes, os módulos compostos e seus componentes (portas, submódulos e parâmetros) são descritos pela linguagem NED (*Network Description Language*). Essa linguagem foi criada especificamente para o simulador OMNeT++ e, a partir dessa linguagem são definidas as estruturas dos modelos de simulação.

5.2 DESCOBERTA AUTOMÁTICA E GERENTE DO NE

No *framework* de simulação criou-se o componente *Gerencia*, um módulo composto que agrupa os módulos simples do Agente de Descoberta (cDA), do Gerente de NE (*cNEManager*) e os módulos dos Gerentes Locais (*cLocalManager*). A Figura 5.1 mostra os componentes básicos da *Gerencia*. Esse módulo é conectado ao plano de transporte através das conexões de cada *cLocalManager*. Este, por sua vez, foi implementado no trabalho de (Tessinari, 2011).

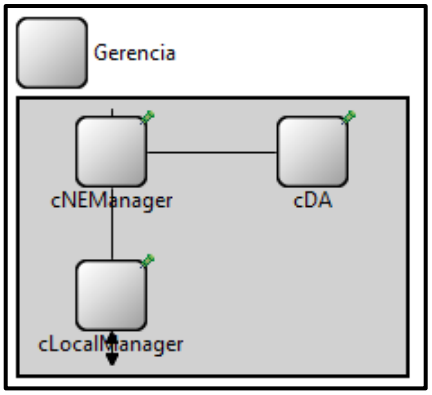

Figura 5.1 – Módulo Composto da Gerência de um NE.

Na Figura 5.2 é apresentada a estrutura dos arquivos relacionados ao plano de gerência na simulação.

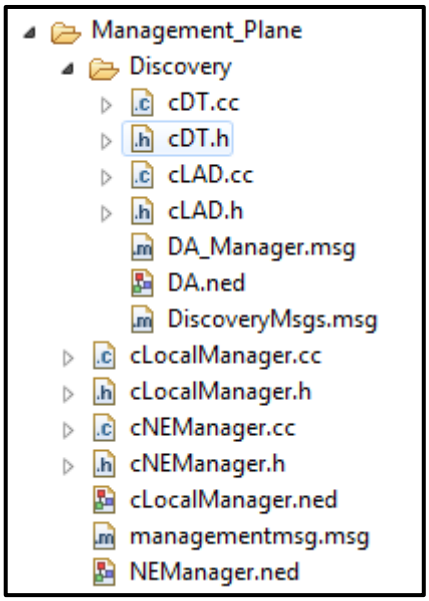

Figura 5.2 – Arquivos relacionados com o plano de e gerência.

Nas próximas seções são abordadas a implementação do Gerente de NE, sua comunicação com os Gerentes Locais e as mensagens envolvidas, a atualização e integração do Agente de Descoberta (DA) aos planos de transporte e gerência.

5.2.1 Módulo Gerente do NE (*cNEManager*)

O Gerente de NE (GNE) é responsável pela comunicação com os Gerentes Locais, o DA e a rede de gerência (DCN). O GNE possui uma estrutura de armazenamento, na qual são guardados os dados de todos os TCPs do NE. Para cada par de TCPs,

é identificada a camada, os identificadores do lado *source* e *sink*, o identificador do Gerente Local responsável por aquele TCP e o parâmetro de ativação da descoberta automática.

Ao iniciar a simulação, cada Gerente Local reporta ao GNE os TCPs identificados. O GNE armazena os TCPs e ativa o DA para as terminações das camadas ODU, OTU e OTS. Quando o Gerente Local identifica alguma mudança de alarmes ou campos do cabeçalho, o mesmo envia os sinais alterados ao GNE juntamente com os identificadores do Gerente Local e dos TCPs onde ocorreram as alterações. Por sua vez, o GNE atualiza um *log* de mudanças na rede e, dependendo da informação recebida, o GNE a reencaminha para outros elementos, por exemplo, a mensagem de descoberta vinda pelo campo SAPI do cabeçalho é encaminhada ao DA para processamento da mesma. Na Figura 5.3 pode ser observado o agrupamento dos logs gerados por cada NE da rede. Para cada alteração verificada são indicados: o tempo, camada, identificador do NE e do TCP, o campo no qual ocorreu a mudança e o valor desse campo.

```
LOG de alteracoes de MI recebidas pelo NE_Manager 8.
@TIME: 0.0035 ### Layer: (1) OTS / NE: 8 / TCP: 6 / Field: cBDI / Change to: TRUE
@TIME: 0.052471001 ### Layer: (2) OMS / NE: 8 / TCP: 7 / Field: cBDI_P / Change to: TRUE

LOG de alteracoes de MI recebidas pelo NE_Manager 46.
@TIME: 0.0001 ### Layer: (1) OTS / NE: 46 / TCP: 0 / Field: cLOS / Change to: TRUE

LOG de alteracoes de MI recebidas pelo NE_Manager 60.


LOG de alteracoes de MI recebidas pelo NE_Manager 85.
@TIME: 0.0003 ### Layer: (2) OMS / NE: 85 / TCP: 1 / Field: cLOS_P / Change to: TRUE

LOG de alteracoes de MI recebidas pelo NE_Manager 99.


LOG de alteracoes de MI recebidas pelo NE_Manager 102.
@TIME: 0.051471001 ### Layer: (3) OCH / NE: 102 / TCP: 1 / Field: cLOS_P / Change to: TRUE
@TIME: 0.049471001 ### Layer: (5) OTU / NE: 102 / TCP: 3 / Field: cSSF / Change to: TRUE
@TIME: 0.049471001 ### Layer: (7) ODU / NE: 102 / TCP: 5 / Field: cSSF / Change to: TRUE
```

Figura 5.3 – *Log* gerado pelo Gerente de NE.

Além de receber alterações detectadas pelo Gerente Local, o GNE pode enviar mensagens de configuração, como a mensagem contendo as informações de TTI a serem inseridas no cabeçalho em determinado TCP. Através dessa configuração que são inseridas as mensagens de descoberta no campo SAPI.

O GNE é utilizado também como interface para a rede de gerência DCN, ele encaminha as mensagens de resposta de descoberta, proveniente do DA, para outros NEs na rede de modo *out-of-band*.

No GNE estão implementados os métodos para escrita de *log* (*WriteLog()*), envio de alteração de SAPI (*SendChangeSAPI()*), atualização da estrutura de dados (*UpdateTCPInfo()*), envio de informações para o DA (*TCPdiscovery(), ReceivedDiscMsgtoDA()*) e verificação de perda de sinais (*VerifyLOSS()*).

5.2.1.1 Principais Mensagens da Gerência

São definidas mensagens para a troca de informações entre os módulos de gerência citados. As mensagens específicas entre as funções atômicas (adaptação e terminação de trilha) e o Gerente Local não são abordadas nesta dissertação, mas podem ser encontradas em (Tessinari, 2011).

Na Tabela 5.1 são apresentadas as principais mensagens implementadas e suas funções.

Tabela 5.1 – Mensagens do GNE.

MENSAGEM	DESCRIÇÃO
TCP_ID_Information	Mensagem enviada pelo TCP, no início da simulação, para indicar sua presença à Gerência.
Manager_Information	Mensagem utilizada entre o GNE e o Gerente Local, para reportar os TCPs, identificando seus respectivos Gerentes Locais.
ChangeField	Reporta para o GNE qualquer alteração detectada nos alarmes das terminações ou configura os campos de cabeçalho (ex.: SAPI) nas terminações. Essa mensagem é dividida nos tipos CHAR e BOOL, dependendo do campo do cabeçalho em questão.

DA_Information	Mensagem enviada pelo GNE para passar os identificadores do TCP e iniciar a descoberta automática para determinado TCP. Também possui um parâmetro para indicar perda de conexão no enlace do respectivo TCP.
Discovery_Message	Mensagem contendo um campo de 16 caracteres utilizado para preencher o campo SAPI das camadas ODU, OTU e OTS.

5.2.2 Módulo de Descoberta Automática (*cDA*)

O DA é o componente da simulação responsável pela descoberta automática das terminações. Como apresentado na Seção 4.2, o DA é composto pelos processos DT e LAD.

O DT é o componente que realiza a interface entre a gerência e os processos LAD, e apresenta, em sua implementação, uma estrutura de dados para armazenamento de todas as informações de identificação trocadas para cada TCP local, além do estado do enlace descoberto.

Dentre os principais métodos do DT, pode-se destacar:

- *writeLog()* e *writeOutput()* – Métodos que criam os arquivos de *log* de TCP descobertos e o arquivo de saída contendo todos os enlaces descobertos pelo DA em questão. Esses arquivos servem como saída de dados do simulador, podendo ser processados para geração de grafos e relatórios referentes às conexões existentes da rede.
- *UpdateTCPInfo() e UpdateLinkStatus()* – O DT atualiza sua tabela de dados ao receber mensagens da gerência, referentes a um novo TCP identificado, ou dos processos LAD, referentes à descoberta de um determinado enlace.

- *discMsg2char()* e *char2discMsg()* – Métodos que realizam a codificação/decodificação das mensagens de descoberta que são inseridas ou extraídas do campo SAPI das terminações.
- *SendDiscMsgtoNE()* – Método que cria e envia uma mensagem de configuração de SAPI a um determinado TCP.

A implementação do LAD é baseada na máquina de estados descrita na recomendação (ITU-T, G.7714, 2005) e explicada com maiores detalhes em (Ferrari, 2009).

Na Figura 5.4 são observados os estados e eventos definidos de acordo com a recomendação (ITU-T, G.7714, 2005).

```
// States of the FSM
enum LADState {
    INIT, S_IDLE, S_AZ, S_ZA, S_AZZA
};

enum LADTxState {
    INITtx, S_IDLEtx
};

// Events of the FSM
enum LADEvent {
    StartLADInstance, RxDiscMsgMatchedZ,
    RxDiscMsgUnMatchedZ, RxDiscAckMatchedZ,
    RxDiscAckUnMatchedZ, FAIL, StopLADInstance,
    StartLADTxInstance, Timeout, StopLADTxInstance
};
```

Figura 5.4 – Estados e eventos do LAD implementados.

Na Figura 5.5 são mostrados os métodos implementados do LAD. Destacam-se os métodos da máquina de estados (FSM_LAD e FSM_LADTx) que implementam todo o mecanismo de transição de estados de acordo com os eventos ocorridos. Ao ocorrer um evento específico, um conjunto de ações é executado. Essas ações são observadas na Figura 5.5 e têm relação direta com as ações descritas na recomendação (ITU-T, G.7714, 2005).

```
protected:
// The following redefined virtual function holds the algorithm.
     void initialize();
     void handleMessage(cMessage *msg);
    FSMs
     void FSM_LAD(LADEvent Event,discovery *msg);
     void FSM_LADTx(LADEvent Event);
    Actions
     void acStartLADTxInstance();
     void acTerminateLADTxInstance();
     void acSetObservedZIdentifier(discovery *msg);
     void acUnsetObservedZIdentifier();
     void acNotifyDTMiswire(discovery *msg);
     void acNotifyDTLinkFound(discovery *msg);
     void acNotifyDTLinkLost();
     void acTxDiscAck();
     void acStartTxTimer();
     void acRestartTxTimer();
     void acTerminateTxTimer();
     void acTxDiscMsg();
```

Figura 5.5 – Métodos do LAD implementados.

5.2.2.1 Principais Mensagens da Descoberta Automática

Na Figura 5.6 é observada a sequencia da troca de mensagens entre os DA de um par de nós (A e B). Para cada TCP, o DA envia uma mensagem de descoberta contendo os identificadores locais pelo cabeçalho do enlace de dados. O nó remoto identifica a mensagem e a repassa para o DA, que envia uma mensagem de resposta contendo os identificadores locais e remotos do enlace. Esta mensagem é enviada através da rede DCN, utilizando o IP do nó A contido na mensagem de descoberta recebida. Esse procedimento é realizado nas duas direções e, depois de finalizado, os dados são correlacionados para verificação de possíveis erros de conexões. Essa verificação acontece comparando os dados recebidos pela mensagem de descoberta e de resposta, possibilitando identificar erros na identificação das terminações dos enlaces. Finalmente, o DA armazena as informações do enlace descoberto.

As mensagens de descoberta e sua resposta são criadas pelo processo LAD e repassadas ao DT. A mensagem de resposta não sofre alterações e é enviada ao Gerente de NE para ser transmitida pela DCN. A mensagem de descoberta é

codificada pelo DT para encaminhar ao Gerente de NE e configurar o SAPI do TCP correspondente.

As demais mensagens de inicialização e finalização do processo, e mensagens para reportar um enlace descoberta ao DT também são implementadas de acordo com a recomendação (ITU-T, G.7714, 2005). Essas mensagens são internas ao DA e são trocadas somente entre o DT e o LAD.

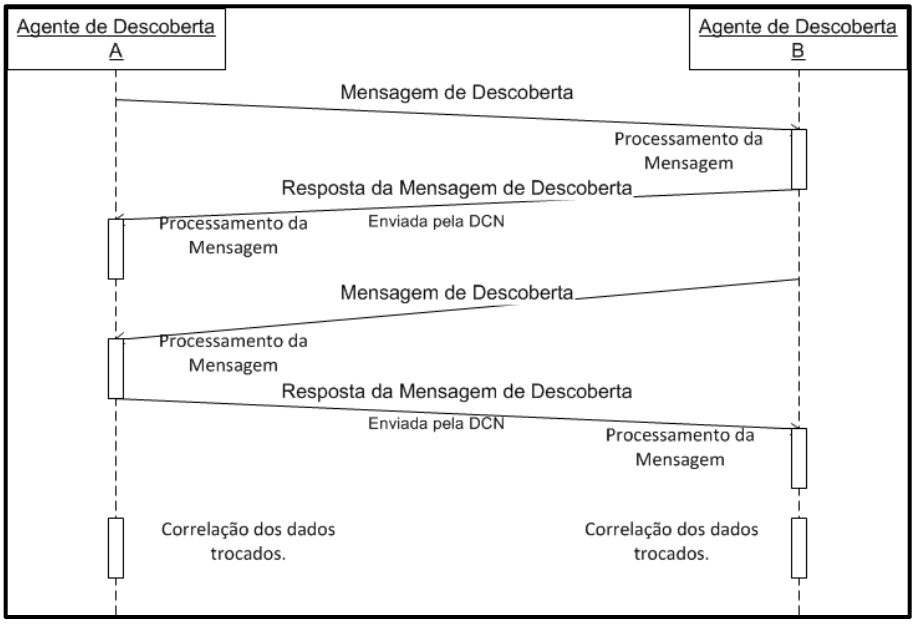

Figura 5.6 – Diagrama de Sequência do *Discovery*.

As mensagens implementadas e uma breve explicação de suas funções são vistas na Tabela 5.2.

Tabela 5.2 – Mensagens internas do DA no Modelo de Simulação.

MENSAGEM	DESCRIÇÃO
LADDiscMsg	Mensagem transmitida pelo plano de dados entre os TCPs cuja conexão será descoberta.
LADDiscAckMsg	Mensagem transmitida através DCN entre os Das, em resposta à Mensagem de Descoberta.
DTLADStart	Mensagem para iniciar o envio de mensagens de descoberta pelo LAD.
DTLADStop	Mensagem para interromper o processo LAD.
LADDTMiswire	Mensagem do LAD ao DT para reportar enlace com erro.
LADDTLinkDisc	Mensagem do LAD ao DT para reportar enlace descoberto.
LADDTLinkLost	Mensagem do LAD ao DT para reportar enlace perdido.

As mensagens entre o GNE e o DA foram abordadas na Seção 5.2.1.1, Tabela 5.1.

5.3 GERENCIAMENTO DE ENLACES – LMP

Nesta seção é apresentada a implementação dos dois procedimentos principais (Seção 3.1) do LMP aplicados na rede OTN. Apesar do *framework* de simulação de redes OTN, desenvolvido no grupo de pesquisa, ainda não possuir funcionalidades como a multiplexação ODU, o LMP foi implementado para efetuar o estabelecimento do canal de controle e a troca de capacidades dos nós no que se refere à camada digital. Para a hierarquia óptica o LMP troca informações referentes aos comprimentos de onda suportados por cada terminação óptica. Para trabalhos futuros, outros processos do plano de controle podem alocar ou desalocar os recursos nas camadas ODU e OCh. Na Figura 5.7 são vistos os arquivos que contêm a implementação do LMP.

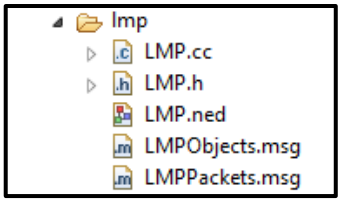

Figura 5.7 – Arquivos da implementação do LMP.

5.3.1 Estruturas de Armazenamento e Métodos

O arquivo LMP.h define a classe do LMP. Nesse arquivo estão contidas as estruturas onde são armazenados os dados referentes às entidades envolvidas e os métodos implementados. Também são definidos os diversos parâmetros e *flags* que são inseridos nas mensagens do LMP.

Como discutido na Seção 3.3, o LMP apresenta três máquinas de estados: do canal de controle, do *TE Link* e do Enlace de Dados. A Figura 5.8 mostra o trecho do arquivo LMP.h em que são declarados os estados de cada máquina e os eventos no contexto do LMP. O estado atual de cada componente é armazenado na sua respectiva estrutura de dados enquanto os eventos são utilizados como parâmetros para o método da máquina de estados de acordo com a mensagem recebida ou algum evento externo.

```
enum ControlChannelState {
    ccDown, ccConfSnd, ccConfRcv, ccActive, ccUp, ccGoingDown
};

enum DataLinkState {
    dtDown, dtTest, dtPasvTest, dtUpFree, dtUpAlloc
};

enum TELinkState {
    teDown, teInit, teUp, teDegraded
};

enum Event {
    evBringUpA, evBringUpB, evCCDn, evConfDone, evConfErr, evNewConfOK,
    evNewConfErr, evContenWin, evContenLostA, evContenLostB, evAdminDown,
    evNbrGoesDn, evHelloRcvd, evHoldTimerA, evHoldTimerB, evSeqNumErr,
    evReconfig, evConfRet, evHelloRet, evDownTimer, evStartTst,
    evStartPsv, evTestOKA, evTestOKB, evTestRcv, evTestFail, evPsvTestFail,
    evLnkAlloc, evLnkDealloc, evTestRet, evSummaryFail, evLocalizeFail, evdcDown,
    evDCUp, evSumAck, evSumNack, evRcvAck, evRcvNack, evSumRet, evCCUp, evCCDown, evDCDown
};
```

Figura 5.8 – Estados e Eventos do LMP definidos.

De acordo com o estudo realizado do protocolo e das informações essenciais para a troca das mensagens do LMP, são criadas estruturas de dados para a representação de cada entidade no contexto do LMP, observadas na Figura 5.9. A estrutura *PeerInfo_t* contém os dados relacionados a um nó vizinho, a *ControlChannel_t* armazena os dados relacionados ao canal de controle, a *TeLink_t* contém informações de determinado *TE Link* e o *DataLink_t* os dados referentes ao enlace de dados.

Cada enlace de dados contém o identificador do *TE Link* no qual está contido. Cada estrutura apresenta, além de parâmetros importantes de seu funcionamento, as mensagens que são utilizadas na temporização de reenvio de mensagens ou detecção de algum problema (ex.: tempo de espera da mensagem *Hello* para indicar queda do canal de controle).

```
struct PeerInfo_t {
    int remoteNodeID;
    int peerMsgID;
    IPAddress remoteNodeIP;
    int AdminStatus;
};
typedef std::vector<PeerInfo_t> PeerVector;

struct ControlChannel_t {
    int ccID;
    int ccMsgID;
    int ccRemoteCCID;
    IPAddress ccRemoteNodeIP;
    double ccHelloInterval;
    double ccHelloIntervalMin;
    double ccHelloIntervalMax;
    double ccHelloDeadInterval;
    double ccHelloDeadIntervalMin;
    double ccHelloDeadIntervalMax;
    ControlChannelState ccState;
    cMessage *hellotimeout;
    cMessage *hellodeadtimeout;
    cMessage *configtimeout;
    cMessage *downtimer;
    int count_config_retry_limit;
    int TxSeqNum;
    int RcvSeqNum;
    unsigned char msgFlag;
    cMessage *cc_up;
    cMessage *cc_down;
};
typedef std::vector<ControlChannel_t> CCVector;

struct DataLink_t {
    int dlTeLinkID;
    int dlLocalInterfaceID;
    int dlRemoteInterfaceID;
    int dlwavelength;
    int dlSwitchType;
    int dlEncType;
    double dlMinimum_Reservable_Bandwidth;
    double dlMaximum_Reservable_Bandwidth;
    int dlHOODTUk;
    int dlTS;
    unsigned char dlLOODUFlags;
    unsigned char dlLOODUFlagsAgree;
    DataLinkState dlState;
    cMessage *testtimeout;
};
typedef std::vector<DataLink_t> DataLinkVector;

struct TeLink_t {
    int teID;
    IPAddress teRemoteNodeIP;
    IPAddress teLocalLinkID;
    IPAddress teRemoteLinkID;
    bool teVerification;
    bool teFaultManagement;
    bool teDwdm;
    TELinkState teState;
    int teVerifyReceiveId;
    int teVerifyId;
    int layer;
    cMessage *summarytimeout;
    cMessage *beginverifytimeout;
    cMessage *endverifytimeout;
    int count_beginverify_retry_limit;
    int count_endverify_retry_limit;
    int count_summary_retry_limit;
    cMessage *verifycomplete;
};
typedef std::vector<TeLink_t> TeLinkVector;
```

Figura 5.9 – Estruturas de armazenamento de informações do LMP.

Os procedimentos do LMP se caracterizam pela troca de mensagens que neles ocorrem. Conforme a mensagem que é recebida ou enviada e as informações nela contidas, se distinguem os eventos. Esses eventos são responsáveis pela transição de estados de cada componente. Como observado na Figura 5.10, as máquinas de estados nativas do LMP (Lang, 2005) foram implementadas através de métodos que utilizam como parâmetros de entrada o evento ocorrido e o índice do vetor de estruturas correspondente a cada componente.

```
// State Machines
void CC_FSM(Event ccEvent, int index);
void DataLink_FSMActive(Event dtEvent,int index);
void DataLink_FSMPassive(Event dtEvent,int index);
void TELink_FSM(Event teEvent, int index);
```

Figura 5.10 – Máquinas de Estados do LMP.

As principais ações de cada máquina de estados são o envio e o recebimento de mensagens. Essas ações foram separadas em métodos que possuem as funções de enviar ou processar cada mensagem descrita.

Os métodos que criam e enviam as mensagens são observados na Figura 5.11. Esses métodos se utilizam dos dados armazenados referentes aos componentes em questão, das informações de mensagens recebidas e parâmetros que indicam estados de funcionamento do canal de controle e do protocolo.

```
// Parameter Negotiation Messages (12.3 RFC 4204)
void sendConfig(ControlChannel_t x, unsigned char msgFlag);
void sendConfigAck(ControlChannel_t x, unsigned char msgFlag, LMPConfig *msg);
void sendConfigNack(ControlChannel_t x, unsigned char msgFlag, LMPConfig *msg);
void sendHello(ControlChannel_t x, unsigned char msgFlag);
// Link Verification Messages (12.5 RFC 4204)
void sendBeginVerify(TeLink_t x, unsigned char msgFlag, unsigned char beginVerifyFlag, int nDataLink);
void sendBeginVerifyAck(TeLink_t x, unsigned char msgFlag, unsigned char verifyResponseFlag, LMPBeginVerify *msg);
void sendBeginVerifyNack(TeLink_t x, IPAddress dest, unsigned char msgFlag, unsigned char BeginVerifyerro, LMPBeginVerify *msg);
void sendEndVerify(int index, IPAddress dest, unsigned char msgFlag);
void sendEndVerifyAck(int index, IPAddress dest, unsigned char msgFlag, LMPEndVerify *msg);
void sendTestMsg(int TEindex, int DTindex, unsigned char msgFlag);
void sendTestStatusSuccess(int TEindex, int DTindex, IPAddress dest, unsigned char msgFlag);
void sendTestStatusFailure(int index, IPAddress dest, unsigned char msgFlag);
void sendTestStatusAck(int index, IPAddress dest, unsigned char msgFlag, LMPMessage *msg);
// Link Summary Messages (12.6 RFC4204)
void sendLinkSummary(TeLink_t x, unsigned char msgFlag);
void sendLinkSummaryAck(unsigned char msgFlag, LMPLinkSummary *msg);
void sendLinkSummaryNack(unsigned char msgFlag, unsigned char link_sum_error, LMPLinkSummary *msg, vector<int> DataLinkIndexProblem);
// Fault Management Messages (12.7 RFC4204)
// NOT USED
void sendChannelStatus();
void sendChannelStatusAck();
void sendChannelStatusRequest();
void sendChannelStatusResponse();
```

Figura 5.11 – Métodos de envio de mensagens do LMP.

Ao receber uma mensagem, é chamado o método de processamento correspondente. Para cada mensagem definida no LMP um método de processamento da mensagem foi implementado, como pode ser visto na lista de métodos da Figura 5.12. A mensagem recebida é passada como único parâmetro desses métodos.

As *Self Messages* são mensagens internas que são usadas para indicar que um *timeout* ocorreu e assim gerar um evento específico. O processamento dessas mensagens geralmente consiste no reenvio de determinada mensagem ou um evento interno ocorrido.

```
// Parameter Negotiation Messages (12.3 RFC 4204)
void processConfig(LMPConfig *msg);
void processConfigAck(LMPConfigAck *msg);
void processConfigNack(LMPConfigNack *msg);
void processHello(LMPHello *msg);
// Link Verification Messages (12.5 RFC 4204)
void processBeginVerify(LMPBeginVerify *msg);
void processBeginVerifyAck(LMPBeginVerifyAck *msg);
void processBeginVerifyNack(LMPBeginVerifyNack *msg);
void processEndVerify(LMPEndVerify *msg);
void processEndVerifyAck(LMPEndVerifyAck *msg);
void processTestMsg(LMPTestMsg *msg);
void processTestStatusSuccess(LMPTestStatusSuccess *msg);
void processTestStatusFailure(LMPTestStatusFailure *msg);
void processTestStatusAck(LMPTestStatusAck *msg);
// Link Summary Messages (12.6 RFC4204)
void processLinkSummary(LMPLinkSummary *msg);
void processLinkSummaryAck(LMPLinkSummaryAck *msg);
void processLinkSummaryNack(LMPLinkSummaryNack *msg);
// Fault Management Messages (12.7 RFC4204)
// NOT USED
void processChannelStatus(LMPChannelStatus *msg);
void processChannelStatusAck(LMPChannelStatusAck *msg);
void processChannelStatusRequest(LMPChannelStatusRequest *msg);
void processChannelStatusResponse(LMPChannelStatusResponse *msg);

// Self-Messages (Timeouts)
void processHelloTimeout(cMessage *msg);
void processHelloDeadTimeout(cMessage *msg);
void processConfigTimeout(cMessage *msg);
void processLinkSummaryTimeout(cMessage *msg);
void processTestTimeout(cMessage *msg);
void processBeginVerifyTimeout(cMessage *msg);
void processEndVerifyTimeout(cMessage *msg);
void processDownTimer(cMessage *msg);
```

Figura 5.12 – Métodos de processamento de mensagens do LMP.

5.3.2 Mensagens

Cada mensagem do LMP é composta por um conjunto de objetos. Esses, por sua vez, podem conter um ou mais sub-objetos. No formato dos objetos existem campos padrões que são utilizados para diferenciar cada objeto, esses campos podem ser observados na Figura 5.13. Os demais objetos implementados são mostrados na Figura 5.14, o detalhamento dos parâmetros contidos em cada é encontrado na RFC4204 (Lang, 2005) e não é descrita nesta dissertação. Esses objetos são utilizados para composição de cada mensagem trocada pelo protocolo do LMP.

```
struct LMPObject
{
    bool N; // NEGOCIABLE?
    int CType;
    int Class;
    int Length;
}
```

Figura 5.13 – Objeto básico do LMP.

```
// ####### OBJECTS of RFC 4204
struct CC_ID extends LMPObject
struct NODE_ID extends LMPObject
struct LINK_ID extends LMPObject
struct INTERFACE_ID extends LMPObject
struct MESSAGE_ID extends LMPObject
struct CONFIG extends LMPObject
struct HELLO extends LMPObject
struct BEGIN_VERIFY extends LMPObject
struct BEGIN_VERIFY_ACK extends LMPObject
struct VERIFY_ID extends LMPObject
struct TE_LINK extends LMPObject
struct DATA_LINK extends LMPObject
struct CHANNEL_STATUS extends LMPObject
struct CHANNEL_STATUS_REQUEST extends LMPObject
struct ERROR_CODE extends LMPObject
```

Figura 5.14 – Objetos implementados do LMP.

Um sub-objeto em especial (Figura 5.15), definido no IETF *draft* (Zhang, et al., 2011), apresenta os parâmetros necessários para a troca de capacidade de enlaces ODU na rede OTN, que são:

- ODTUk – Indica o tipo de ODU do enlace de alta ordem.

- TS – Indica a granularidade do sinal.
- LOODUFlags – Indica quais ODU de baixa ordem podem ser multiplexadas na ODU de alta ordem.

```
struct SubHOODUCapability
{
    int Type;
    int Length = 8;
    int ODTUk;
    int TS;
    unsigned char LOODUFlags;
}
```

Figura 5.15 – Principal objeto para implementação do LMP na OTN.

Nesta dissertação são implementadas todas as mensagens descritas na RFC4204 (Lang, 2005). Essas mensagens são separadas de acordo com o procedimento do qual fazem parte, ou seja, estão divididas em grupos de mensagens relacionadas com cada procedimento descrito no Capítulo 3. Apesar dos procedimentos opcionais não estarem no escopo deste trabalho, suas mensagens foram definidas, deixando uma base para trabalhos futuros. As mensagens são definidas utilizando a classe *packet* que é utilizada no OMNeT++ para representar os pacotes que trafegam na rede. As mensagens do LMP são, posteriormente, encapsuladas em UDP e IP para serem transmitidas através da rede de controle ou gerência.

Na Figura 5.16 podem-se observar as principais mensagens utilizadas nesta dissertação.

```
packet LMPMessage
{
    unsigned char Vers;
    unsigned char Flags;
    int Type;
    int Length;
    IPAddress senderAddress;
    IPAddress destinationAddress;
}

// Parameter Negotiation Messages (12.3 RFC 4204)
packet LMPConfig extends LMPMessage
{
    Type = LMP_MSGT_CONFIG;
    CC_ID localCCId;
    MESSAGE_ID messageID;
    NODE_ID localNodeId;
    CONFIG config;
}
```

```
// Hello MSG (12.4 RFC 4204)
packet LMPHello extends LMPMessage
{
    Type = LMP_MSGT_HELLO;
    CC_ID localCCId;
    CC_ID remoteCCId;
    HELLO helloMsg;
}

// Link Summary Messages (12.6 RFC4204)
packet LMPLinkSummary extends LMPMessage
{
    Type = LMP_MSGT_LINK_SUMMARY;
    MESSAGE_ID messageId;
    TE_LINK teLink;
    DTLinkV dataLinkVector;
}
```

Figura 5.16 - Principais mensagens do LMP.

A mensagem *LMPMessage* é o cabeçalho padrão de todas as demais mensagens do LMP. Seu conteúdo apresenta um parâmetro para indicar a versão do protocolo utilizado, um campo de *Flags* para indicar o modo de funcionamento ou evento específico, o *Type* que indica o tipo da mensagem e seu tamanho *Length*. Os parâmetros que indicam os IPs de origem (*senderAddress*) e destino (*destinationAddress*) não estão descritos em (Lang, 2005), porém, foram adicionados à mensagem nesta implementação. Essa adição é explicada pelo fato de não haver uma interface para esse tipo de comunicação com o módulo do protocolo IP no simulador.

Ainda na Figura 5.16, observa-se as principais mensagens dos procedimentos de gerenciamento do canal de controle (*LMPConfig e LMPHello*) e correlação de propriedade de enlace (*LMPLinkSummary*).

Basicamente, a *LMPConfig* carrega os objetos de identificação local do canal de controle (*CC_ID*), identificação para a mensagem (*MESSAGE_ID*), identificação do nó (*NODE_ID*) e de configuração (*CONFIG*), que contêm os parâmetros que configuram o envio de mensagens *Hello* pelo canal. A mensagem *LMPHello* possui os identificadores do canal de controle local e remoto e o objeto (HELLO) que contêm os números de sequência do protocolo *Hello*.

A mensagem *LMPLinkSummary* contém um identificador da mensagem (*MESSAGE_ID*), identificador do *TE Link* (*TE_LINK*) e um vetor contendo um conjunto de objetos *DATA_LINK*. Esses, por sua vez, contêm os identificadores local e remoto e os sub-objetos que descrevem as propriedades e capacidades do enlace de dados.

O conjunto de todas as mensagens implementadas é observado na Figura 5.17. A implementação dessas segue a descrição apresentada em (Lang, 2005), onde o conteúdo de cada mensagem é detalhado.

```
// Parameter Negotiation Messages (12.3 RFC 4204)
packet LMPConfig extends LMPMessage
packet LMPConfigAck extends LMPMessage
packet LMPConfigNack extends LMPMessage
// Hello MSG (12.4 RFC 4204)
packet LMPHello extends LMPMessage
// Link Verification Messages (12.5 RFC 4204)
packet LMPBeginVerify extends LMPMessage
packet LMPBeginVerifyAck extends LMPMessage
packet LMPBeginVerifyNack extends LMPMessage
packet LMPEndVerify extends LMPMessage
packet LMPEndVerifyAck extends LMPMessage
packet LMPTestMsg extends LMPMessage
packet LMPTestStatusSuccess extends LMPMessage
packet LMPTestStatusFailure extends LMPMessage
packet LMPTestStatusAck extends LMPMessage
// Link Summary Messages (12.6 RFC4204)
packet LMPLinkSummary extends LMPMessage
packet LMPLinkSummaryAck extends LMPMessage
packet LMPLinkSummaryNack extends LMPMessage
// Fault Management Messages (12.7 RFC4204)
packet LMPChannelStatus extends LMPMessage
packet LMPChannelStatusAck extends LMPMessage
packet LMPChannelStatusRequest extends LMPMessage
packet LMPChannelStatusResponse extends LMPMessage
```

Figura 5.17 - Mensagens implementadas do LMP.

5.3.3 Procedimento realizado pelo LMP

O módulo do LMP inicia seus procedimentos com a chegada das informações provenientes da descoberta automática. Ao receber os dados dos enlaces descobertos, o LMP armazena as informações nas tabelas de vizinhos, cria os *TE Links*, agrupando os enlaces de dados descobertos de cada camada, e inicia o procedimento para estabelecer os canais de controle com seus vizinhos. A exemplificação da sequência do procedimento realizado pelo LMP é observada na Figura 5.18.

O NE 0, após receber os dados de enlaces descobertos, inicia o processo de estabelecer os canais de controle através do envio de mensagens *LMPConfig* para seu vizinho. Se os parâmetros são aceitáveis, o nó remoto (NE 1) responde com *LMPConfigAck*, se os parâmetros não são aceitos, acontece outra rodada de negociação.

Aceitos os parâmetros, as mensagens *Hello* começam a ser enviadas por ambos os nós. Após recebimento da primeira mensagem *Hello*, o estado do canal de controle é colocado como ativo para uso.

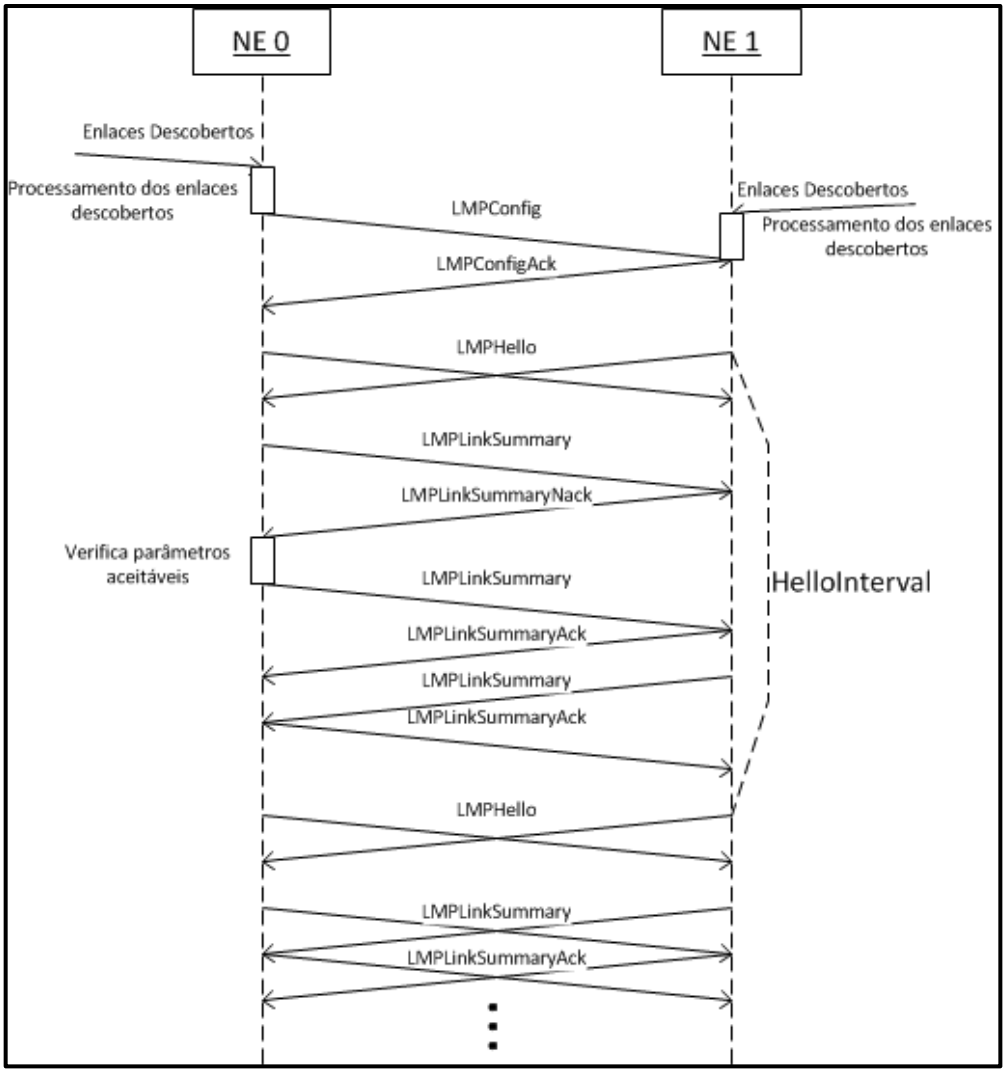

Figura 5.18 - Sequência de mensagens do LMP.

Na sequência, os nós enviam uma mensagem para correlacionar as propriedades dos enlaces (*LMPLinkSummary*). No exemplo da Figura 5.18, os parâmetros não foram aceitos pelo nó remoto (NE 1), o que pode ter acontecido devido esse nó não aceitar o mesmo esquema de multiplexação do nó NE 0. Caso isso ocorra, o nó que originou a mensagem (NE 0) verifica os parâmetros aceitáveis contidos na mensagem *LMPLinkSummaryNack* recebida e, caso aceite tais parâmetros, envia uma nova mensagem *LMPLinkSummary* com esses novos parâmetros acordados por ambos os nós.

As mensagens *LMPHello* e *LMPLinkSummary* continuam sendo enviadas periodicamente e as informações de identificação e capacidade do enlace ficam disponíveis para serem utilizadas pela gerência e controle da rede.

5.4 ROTEADOR IP DO INET

Para a utilização do protocolo IP para identificação de nós nas redes OTN simuladas são utilizados módulos do pacote INET (Varga & Acadêmica, INET Framework main page, 2011), que é um *framework* para o OMNET++ para simulação de redes. No INET são encontrados os módulos necessários para a composição de um roteador IP, que é utilizado como base para a construção do controlador GMPLS referenciado em (Favoreto, 2009) e (Tessinari, 2011).

Um módulo composto (*Router_LMP* - Figura 5.19) é desenvolvido nesta dissertação com a finalidade de efetuar o roteamento de mensagens IP entre os nós da rede e adicionar o módulo do LMP, para, em trabalhos futuros, ser integrado com os módulos referentes ao GMPLS.

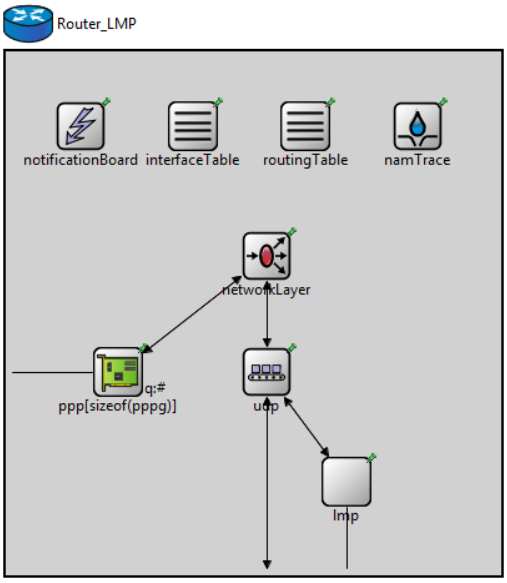

Figura 5.19 - Roteador IP do INET com módulo do LMP.

Além do LMP, este módulo apresenta, como principais elementos:

- *notificationBoard* – Torna possível a comunicação entre outros módulos, por exemplo, notificações de mudança da tabela de roteamento;
- *interfaceTable* – Contém a tabela das interfaces de rede (eth0, wlan0, etc.) do nó local;
- *routingTable* – Contém a tabela de roteamento IPv4;

- *PPP* – Módulo para encapsulamento da mensagem pelo PPP (*Point to Point Protocol*);
- *networkLayer* – Representa os protocolos da camada de rede. No caso desta dissertação, contém os módulos de IPv4, ARP (*Address Resolution Protocol*), ICMP (*Internet Control Message Protocol*) e IGMP (*Internet Group Management Protocol*);
- *UDP* – Encapsula as mensagens no protocolo camada de transporte UDP (*User Datagram Protocol*).

De acordo com a RFC 4204 (Lang, 2005), as mensagens LMP utilizam o protocolo do transporte UDP na porta 701 para serem transmitidas. A outra conexão de aplicação que é observada na Figura 5.19 representa sua ligação com a gerência do ONE. Desse modo, a gerência de cada ONE pode se comunicar através da rede IP, transmitindo, por exemplo, as mensagens de respostas da descoberta automática.

Este módulo composto (*Router_LMP*) é inserido em cada nó e suas conexões formam a rede IP, que pode ser utilizada como a rede de gerência DCN ou a rede de controle da rede. As conexões entre os nós dessa rede IP pode ser um comprimento de onda específico na própria fibra óptica que transporta os dados (canal de serviço) ou uma rede independente.

5.5 SOFTWARE DE GERENCIAMENTO DA DESCOBERTA

Para exercer as funções de geração dos relatórios e grafos através das informações obtidas pelo procedimento de Descoberta Automática, foi criado um programa em JAVA.

Desse modo, o OMNeT++ simula a rede e a troca de mensagens pertinentes à Descoberta Automática, e cada DA gera um arquivo de saída com o estado atual dos enlaces. Esses arquivos são lidos pelo programa e são gerados os relatórios e grafos de acordo com a requisição através da interface do programa.

O programa também permite o monitoramento de mudanças no estado dos enlaces durante a execução da simulação. Os grafos são gerados através de arquivos *.dot* pelo programa *GraphViz* (*Graph Visualization*). Os arquivos *.dot* e a chamada de sistema para geração do grafo é realizada através do próprio programa em JAVA.

A interface do programa (apresentada na Figura 5.20) é simples e, para sua utilização, deve-se entrar com as seguintes informações:

- Indicar a pasta onde são gravados os arquivos pelo DA na simulação;
- Indicar a pasta de saída para salvar os arquivos de relatórios e grafos gerados;
- Caso não seja detectado, deve-se direcionar o executável *dot.exe* do *programa GraphViz* instalado no computador;
- Seleção das camadas a serem abordadas pelos relatórios e grafos;
- Seleção para monitoração de alterações. Nesse caso, o programa fica em execução monitorando alterações nos arquivos e na topologia da rede, sendo atualizada em períodos de tempo indicado pelo usuário;
- Seleção dos arquivos a serem gerados: Relatórios e/ou Grafos.

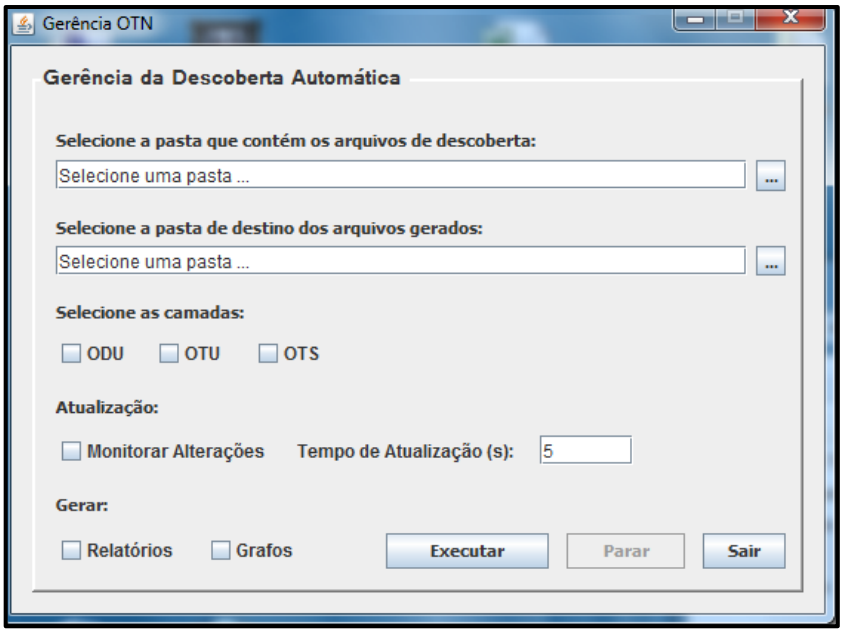

Figura 5.20 - Interface gráfica do programa de exibição de grafos.

O grafo gerado é aberto pelo programa e a Figura 5.21 representa um exemplo prático. Os TCPs na cor vermelha indicam um enlace perdido e na cor verde indicam

TCPs descobertos em estado normal. Esse grafo é atualizado em intervalos de tempo indicados pelo usuário.

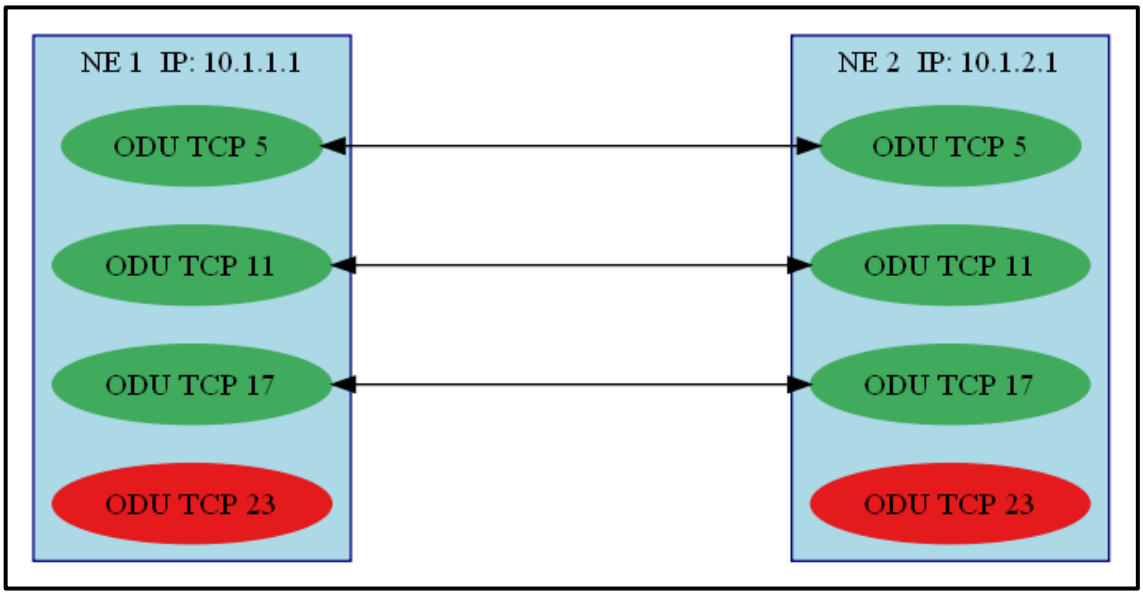

Figura 5.21 - Exemplo de grafo detalhado.

Outro grafo mais simples, é gerado pelo programa na pasta indicada. Esse grafo contém os NEs da rede como nós e indicam a quantidade de enlaces funcionais de determinada camada entre os NEs. Esse grafo pode ser observado na Figura 5.22.

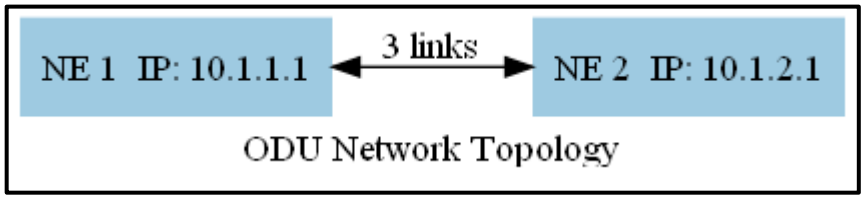

Figura 5.22 - Exemplo de grafo gerado.

Os grafos são gerados em arquivos de figura (*.png*) e salvos na pasta selecionada para saída. Os relatórios também são salvos nessa pasta e sempre atualizados com o último estado observado da rede.

O relatório indica a camada do enlace descoberto, seu estado atual e os identificadores das terminações do enlace. Um exemplo do relatório gerado pelo programa pode ser observado na Figura 5.23.

```
Link ODU
Status: LINKDISC
     DA LOCAL:      ID: 10.1.1.1
     TCP LOCAL:     ID: 5
     DA REMOTO:     ID: 10.1.2.1
     TCP REMOTO:    ID: 5

Link ODU
Status: LINKDISC
     DA LOCAL:      ID: 10.1.1.1
     TCP LOCAL:     ID: 11
     DA REMOTO:     ID: 10.1.2.1
     TCP REMOTO:    ID: 11
```

Figura 5.23 - Exemplo de relatório gerado.

5.5.1 Classes Implementadas

Além das classes referentes à interface gráfica, foram implementadas três classes principais que descrevem o comportamento de elementos fundamentais na simulação, no que se refere à descoberta automática. As três classes estão descritas na Tabela 5.3.

Tabela 5.3 - Classes do programa em Java.

CLASSE	DESCRIÇÃO
ManagerLinkTCP	Classe que referencia o TCP e todos os dados obtidos da Descoberta Automática através dele. Possui a camada do TCP, os identificadores do TCP remoto descoberto e o estado do enlace (Desconhecido, Descoberto, Com ERRO ou Perdido).
ManagerDA	Referencia cada DA na rede. Possui o identificador do NE/DA, o vetor de TCPs contidos no escopo do NE e o arquivo gerado pelo DA correspondente. É função do *ManagerDA* atualizar seu vetor de TCPs e conexões sempre que o arquivo correspondente for alterado pela simulação. O arquivo é percorrido para preenchimento do vetor de TCPs com dados atualizados.

ManagerNetwork	O objeto dessa classe gerencia todos os objetos *ManagerDA* e faz a pesquisa dos arquivos contidos na pasta indicada pelo usuário em busca dos arquivos de descoberta gerados pelos DAs na simulação. Um vetor de DA é mantido nesse objeto, e implementa as funções de geração dos relatórios de enlaces descobertos na rede e os grafos da rede.

5.6 CONSIDERAÇÕES FINAIS

Neste capítulo são descritos os principais aspectos envolvidos na implementação do procedimento de descoberta automática baseado na recomendação (ITU-T, G.7714, 2005) e (ITU-T, G.7714.1, 2010) e do protocolo de gerenciamento de enlaces LMP (Lang, 2005).

É apresentado também a implementação do módulo de Gerência de NE e os módulos que possibilitam a criação da rede IP, externa ao plano de dados.

Na Seção 5.5 é mostrado o programa desenvolvido para criação dos grafos e relatórios a partir dos arquivos provenientes da simulação.

6 **TESTES**

Este capítulo tem como objetivo validar a implementação e integração dos módulos de descoberta automática e do LMP no *framework* de simulação de redes OTN. Nas Seções 6.1 e 6.2 são demonstrados os casos de teste utilizados, os resultados obtidos da descoberta automática e as informações de capacidade descoberta/acordadas pelo LMP. Na Seção 6.3 apresentam-se as considerações finais a respeito dos testes realizados.

Para a validação das funcionalidades desenvolvidas são montadas duas redes no simulador. Essas redes apresentam topologias físicas e lógicas distintas quando observadas de diferentes camadas da rede OTN. A validação é obtida com a verificação da topologia obtida através do procedimento de descoberta (em grafos) e da capacidade e outras propriedades acordadas pelo LMP. Essas informações devem condizer com as configurações da rede simulada.

Para montagem das topologias de rede para os casos de teste, são utilizados módulos compostos que representam alguns equipamentos e, por sua vez, os ONEs (*Optical Network Elements*) das redes OTN. De modo resumido, os equipamentos aqui utilizados são:

- ONE_Host – Responsável por gerar e receber o tráfego OTN. Possui desde as camadas digitais até a última camada óptica;
- ONE_AMP – Amplificador de linha óptico. Apenas amplifica o sinal óptico sem conversão óptico-elétrico;
- ONE_OXC – Esse elemento possui a capacidade de comutação de comprimentos de onda.

Maiores informações sobre a implementação desses equipamentos do plano de transporte da rede OTN encontra-se em (Tessinari, 2009) e (Tessinari, 2011).

6.1 CASO DE TESTE 1

Para o primeiro caso de teste é montada a rede observada na Figura 6.1. Para uma melhor observação, os módulos *router_LMP* são conectados externamente à cada nó da rede, e as conexões entre eles representam a rede IP onde trafegam mensagens de gerência e controle da rede.

Nesta topologia, observam-se quatro *hosts* OTN conectados, por um par de fibras, a um OXC, formando uma topologia física em estrela. Cada *host* trabalha com quatro comprimentos de onda, ou seja, quatro geradores de tráfego contendo todas as camadas digitais e sendo multiplexadas no domínio óptico para a fibra. O OXC está configurado de modo que essa rede represente uma topologia lógica em anel, da seguinte forma:

- *NE 0 (Host_0)* – comprimentos de onda 0 e 1 (azul claro e escuro) – *NE 1 (Host_1);*
- *NE 0 (Host_0)* – comprimentos de onda 2 e 3 (verde claro e escuro) – *NE 2 (Host_2);*
- *NE 1 (Host_1)*– comprimentos de onda 2 e 3 (verde claro e escuro) – *NE 3 (Host_3);*
- *NE 2 (Host_2)*– comprimentos de onda 0 e 1 (azul claro e escuro) – *NE 3 (Host_3).*

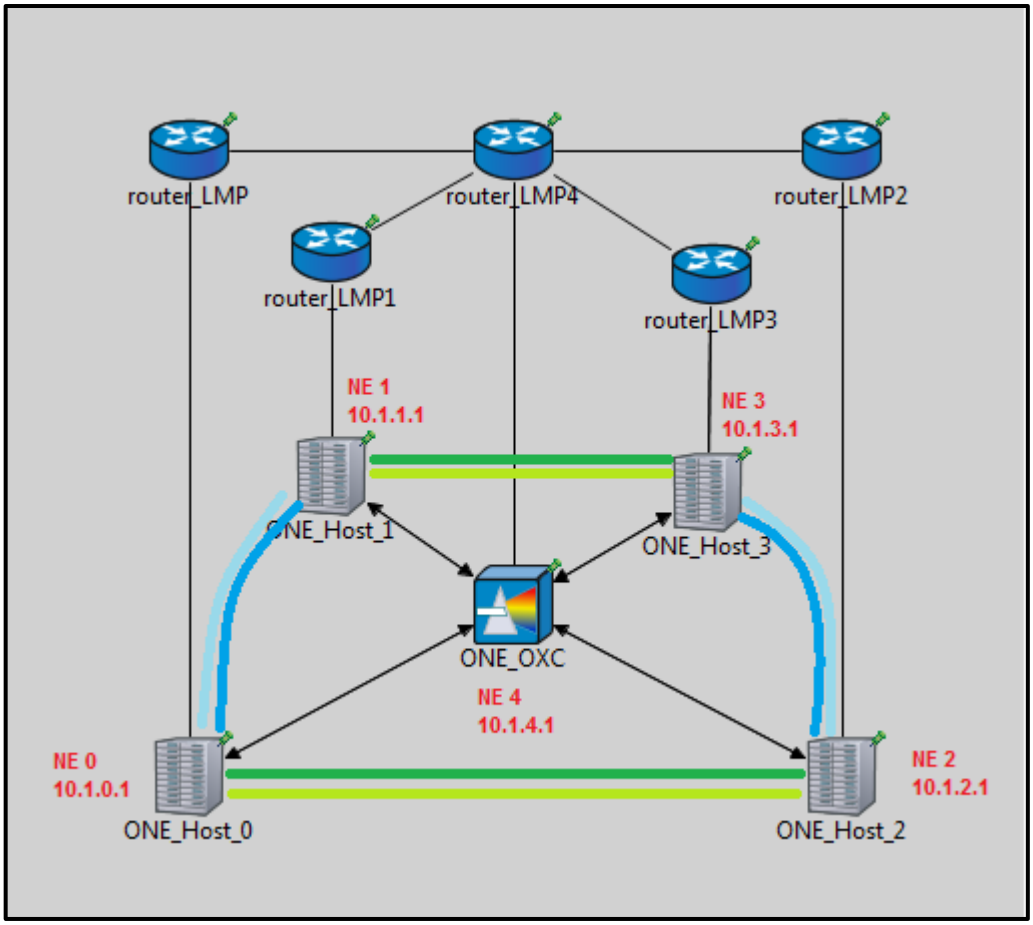

Figura 6.1 - Topologia para teste 1.

Os enlaces ODU entre todos os NEs suportam sinais de alta ordem ODU2 (10Gbps), porém, o NE 0 e o NE 1 estão configurados para suportarem multiplexação de sinais ODU1 (2,5Gbps) de baixa ordem, enquanto os demais não possuem suporte de multiplexação, trabalhando apenas com sinais ODU2. Vale lembrar que essa configuração é realizada para efeito de teste do LMP, visto que a multiplexação de camada digital ainda não é implementada no *framework* de simulação OTN.

6.1.1 Resultados da Descoberta Automática

Ativando o procedimento de descoberta automática, baseada na recomendação (ITU-T, G.7714, 2005), para as camadas ODU, OTU e OTS da rede, obtêm-se os grafos e relatórios contendo as conexões existentes na rede, na visão de cada camada citada. Como os resultados das camadas ODU e OTU são os mesmos, visto

que essas camadas são relacionadas de forma 1:1 devido ao *framework* não possuir multiplexação ODU, vamos apresentar os grafos apenas para a camada ODU.

Na Figura 6.2 são observados os grafos de topologia gerados para as camadas (A) ODU e (B) OTS. Esses grafos apresentam a topologia entre os ONEs, sem detalhes referentes aos TCPs em cada ONE. Suas ligações apresentam a quantidade de enlaces descobertos da camada em questão.

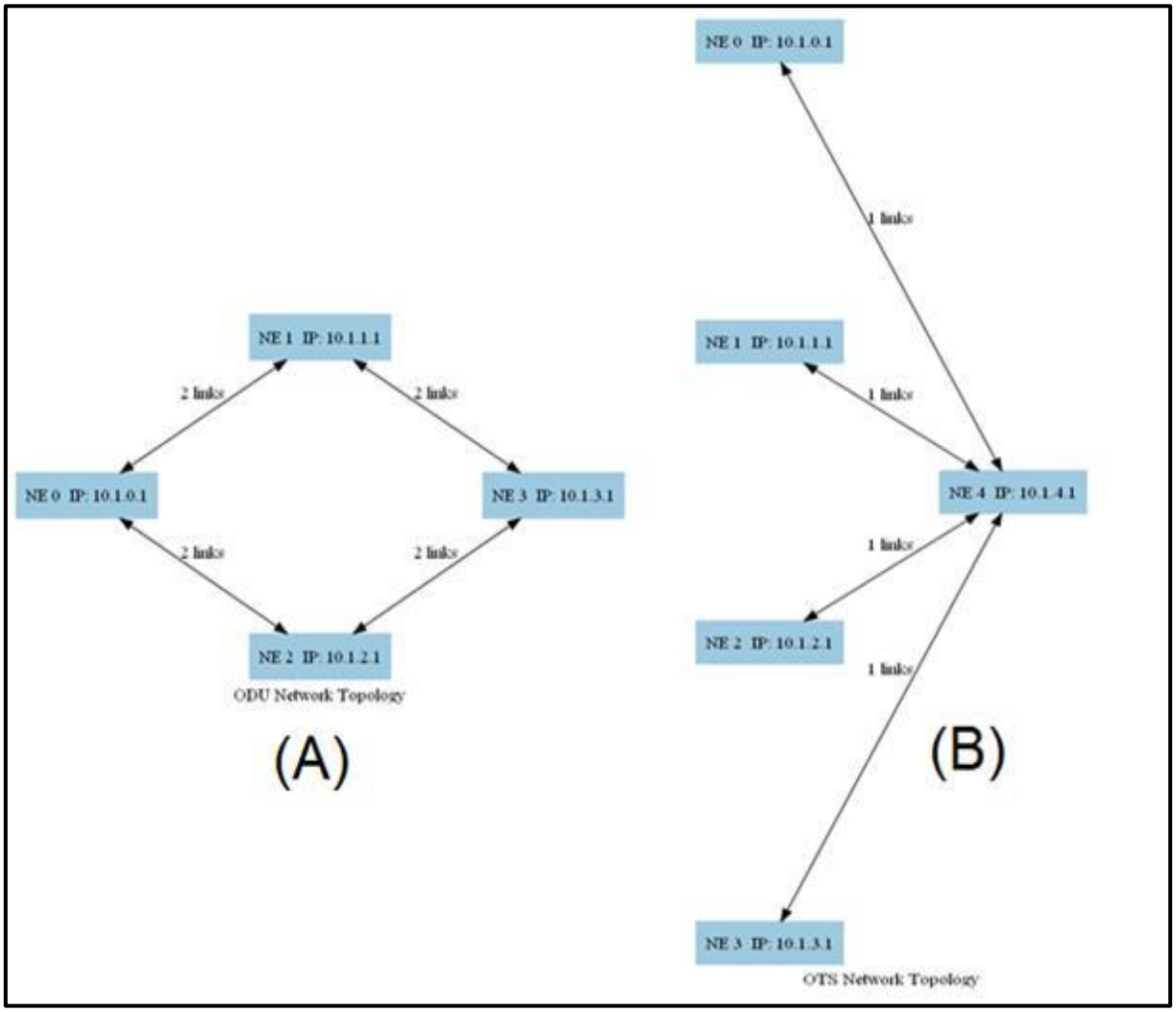

Figura 6.2 - (A) Topologia ODU descoberta. (B) Topologia OTS descoberta.

A topologia OTS mostra os quatro *hosts* conectados ao OXC, que realiza a comutação dos comprimentos de onda. Como a OTS representa a conexão física por fibra óptica, obtêm-se uma topologia em estrela, que representa como os ONEs estão fisicamente ligados.

A topologia ODU mostra os quatro nós conectados em anel. Isso ocorre devido à configuração do OXC, que comuta os comprimentos de onda, não terminando qualquer camada digital. Portanto, na visão da rede pela camada ODU, o nó OXC não apresenta conexões com os demais.

Como pode ser observado na Figura 6.3, cada *host* possui uma terminação bidirecional OTS, detalhada com seus identificadores, pois são conectados por apenas um par de fibras com o OXC. Este, no entanto, possui quatro terminações OTS, uma para cada par de fibras.

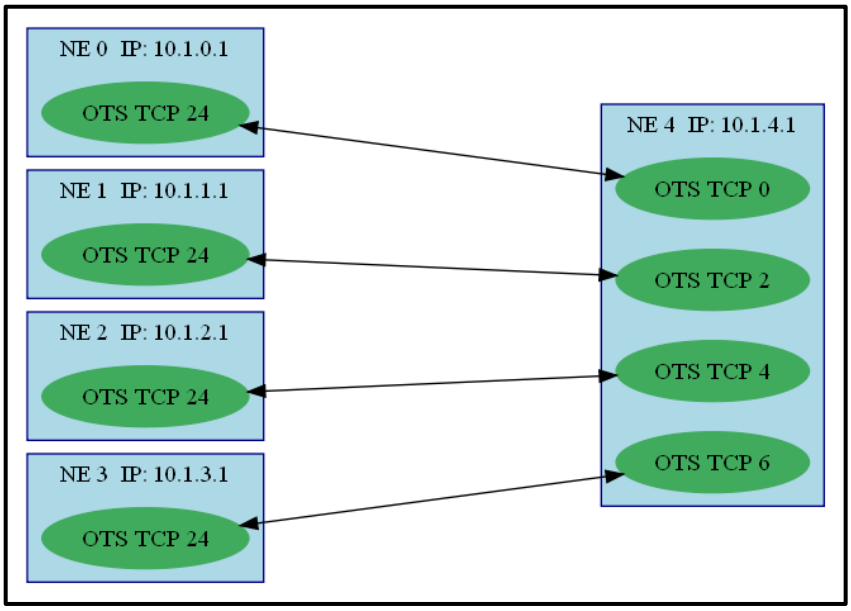

Figura 6.3 - Topologia detalhada OTS para Teste 1.

Na Figura 6.4 é mostrada a topologia em anel da camada ODU mencionada anteriormente. Nesta figura são identificados cada TCP e podem ser observados os dois enlaces existentes entre cada ONE em questão. Os grafos são criados através dos arquivos gerados por cada DA durante a simulação, sendo agrupados pelo programa desenvolvido em Java.

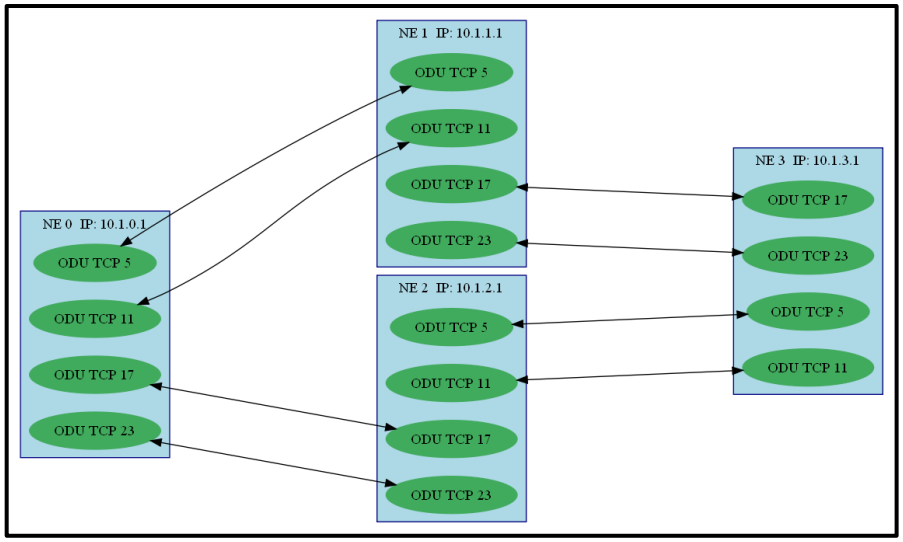

Figura 6.4 - Topologia detalhada ODU para Teste 1.

6.1.2 Resultados do LMP

Para demonstrar os resultados do LMP, são mostrados e explicados a seguir trechos do arquivo de saída gerado pelo módulo do LMP durante a simulação. O módulo LMP de cada nó da rede gera um arquivo contendo as tabelas com dados de seus vizinhos, canais de controle, *TE Links* e enlaces de dados. A seguir, é analisado o arquivo referente ao NE 0 da rede.

Na Figura 6.5, observa-se os vizinhos identificados através das informações obtidas pela descoberta automática e reportadas ao LMP. O NE 0 possui como vizinhos (remoteNodeID) o NE 1, NE 2 e o NE 4 (OXC). O campo *peerMsgID* armazena o identificador da última mensagem LMP enviada, se refere ao *MessageID* descrito no LMP. Por último, o *remoteNodeIP* é o endereço IP do nó vizinho descoberto.

```
%%%%%% PEER INDEX 0 %%%%%% %%%%%% PEER INDEX 1 %%%%%% %%%%%% PEER INDEX 2 %%%%%%
remoteNodeID: 4           remoteNodeID: 1           remoteNodeID: 2
peerMsgID: 19             peerMsgID: 20             peerMsgID: 19
remoteNodeIP: 10.1.4.1    remoteNodeIP: 10.1.1.1    remoteNodeIP: 10.1.2.1
```

Figura 6.5 - Tabelas de vizinhos identificados pelo LMP.

Na Figura 6.6 são apresentados os dados dos canais de controle estabelecidos com cada vizinho. Nessa estrutura de dados estão, inicialmente, contidos:

- *ccID*: Identificador local do canal de controle;
- *ccRemoteCCID*: Identificador do canal de controle no nó remoto;
- *ccRemoteNodeIP*: Endereço IP do nó remoto.

Além desses parâmetros, os três canais de controle possuem parâmetros de configuração de intervalo para envio de mensagens *Hello* (*ccHelloInterval*) e para o intervalo de espera antes de declarar falha no canal (*ccHelloDeadInterval*). Esses parâmetros são acordados entre os nós entre os seus valores mínimo e máximo. Também é identificado o estado do canal de controle, ativo (*ccUp*). Os campos *TxSeqNum e RcvSeqNum* são os números de sequência utilizados nas mensagens *Hello* que são incrementados a cada envio e recebimento respectivamente.

```
%%%%%% CC INDEX 0 %%%%%%      %%%%%% CC INDEX 1 %%%%%%      %%%%%% CC INDEX 2 %%%%%%
ccID: 0                       ccID: 1                       ccID: 2
ccRemoteCCID: 0               ccRemoteCCID: 1               ccRemoteCCID: 2
ccRemoteNodeIP: 10.1.4.1      ccRemoteNodeIP: 10.1.1.1      ccRemoteNodeIP: 10.1.2.1
ccHelloInterval: 0.5          ccHelloInterval: 0.5          ccHelloInterval: 0.5
ccHelloIntervalMin: 0         ccHelloIntervalMin: 0         ccHelloIntervalMin: 0
ccHelloIntervalMax: 2         ccHelloIntervalMax: 2         ccHelloIntervalMax: 2
ccHelloDeadInterval: 3        ccHelloDeadInterval: 3        ccHelloDeadInterval: 3
ccHelloDeadIntervalMin: 2     ccHelloDeadIntervalMin: 2     ccHelloDeadIntervalMin: 2
ccHelloDeadIntervalMax: 4     ccHelloDeadIntervalMax: 4     ccHelloDeadIntervalMax: 4
ccState: ccUp                 ccState: ccUp                 ccState: ccUp
TxSeqNum: 37                  TxSeqNum: 37                  TxSeqNum: 37
RcvSeqNum: 37                 RcvSeqNum: 37                 RcvSeqNum: 37
```

Figura 6.6 - Tabelas de dados dos canais de controle estabelecidos pelo LMP.

Na Figura 6.7 são mostrados os *TE Links* configurados pelo LMP. Sua estrutura de dados contém:

- *teID*: Identificador do *TE Link* no nó local;
- *teRemoteNodeIP*: Endereço IP do nó remoto;
- *teLocalLinkID*: Identificador IP do *TE Link* no nó local;
- *teRemoteLinkID*: Identificador IP do *TE Link* no nó remoto;
- *teVerification*: Suporte ao procedimento de verificação de enlace;
- *teFaultManagement*: Suporte ao procedimento de gerenciamento de falhas;
- *teState:*Estado do *TE Link;*
- *layer*: Camada do respectivo *TE Link;*

Neste caso, observa-se dois *TE Links* compostos por enlaces ODU, ambos em estado funcional (*teUp*), um para o NE1 e outro para o NE 2. Um terceiro *TE Link* é criado para representar o conjunto de comprimentos de onda entre o NE 0 e NE 4 (OXC).

```
%%%%%% TE INDEX 0 %%%%%%%   %%%%%% TE INDEX 1 %%%%%%%   %%%%%% TE INDEX 2 %%%%%%%
teID: 0                     teID: 1                     teID: 2
teRemoteNodeIP: 10.1.1.1    teRemoteNodeIP: 10.1.2.1    teRemoteNodeIP: 10.1.4.1
teLocalLinkID: 10.1.0.2     teLocalLinkID: 10.1.0.3     teLocalLinkID: 10.1.0.4
teRemoteLinkID: 10.1.1.2    teRemoteLinkID: 10.1.2.3    teRemoteLinkID: 10.1.4.2
teVerification: 1           teVerification: 1           teVerification: 1
teFaultManagement: 0        teFaultManagement: 0        teFaultManagement: 0
teState: teUp               teState: teUp               teState: teUp
layer: ODU                  layer: ODU                  layer: OTS
```

Figura 6.7 - TE Links criados pelo LMP.

Na Figura 6.8, são apresentadas as estruturas de dados referentes aos enlaces de dados. Estas são compostas por:

- *dlTeLinkID*: Identificador do *TE Link* ao qual o enlace de dados está inserido;
- *dlLocalInterfaceID*: Interface local do enlace de dados (contexto do *TE Link*);
- *dlRemoteInterfaceID*: Interface remota do enlace de dados;
- *dlMinimum_Reservable_Bandwidth*: Banda mínima que pode ser reservada;
- *dlMaximum_Reservable_Bandwidth*: Banda máxima que pode ser reservada;
- *dlHOODTUk*: Tipo de ODU de alta ordem;
- *dlTS*: Tributário que pode ser alocado;
- *dlLOODUFlags*: Tipo de multiplexação suportado pelo nó;
- *dlLOODUFlagsAgree*: Tipo de multiplexação acordado para o enlace;
- *dlState*:Estado operacional do enlace de dados.

São observados os enlaces de dados ODU, contidos nos *TE Link* 0 e 1, apresentados anteriormente. São mostrados os identificadores de interfaces obtidos pelo procedimento de descoberta automática, a identificação da ODU de alta ordem transportada no enlace (*dlHOODTUk*), a granularidade suportada no enlace (*dlTS*), os sinais de baixa ordem suportados pela terminação local (*dlLOODUFlags*) e os sinais de baixa ordem suportados pelo enlace, ou seja, acordados por ambos os nós

(*dlLOODUFlagsAgree*). Também é mostrado o estado do enlace, neste caso todos em estado funcional e livres (*dtUpFree*).

Ainda na Figura 6.8 observa-se o acordo referente à capacidade de multiplexação dos nós. Como citado anteriormente, o NE 0 e NE 1 possuem capacidade de multiplexação de ODU1, enquanto os demais nós não possuem tal suporte. Logo, para os enlaces ODU entre o NE 0 e NE 1, é acordado o suporte para sinais ODU1 e ODU2, enquanto para o NE 2 (enlaces contidos no *TE Link* 1) apenas sinais ODU2 são suportados. Essas informações são importantes no momento de alocação de recursos na rede.

```
%%%%%% DT INDEX 0 %%%%%%%
dlTeLinkID: 0
dlLocalInterfaceID: 5
dlRemoteInterfaceID: 5
dlMinimum_Reservable_Bandwidth: 2.5
dlMaximum_Reservable_Bandwidth: 10
dlHOODTUk: 2
dlTS: 2.5
dlLOODUFlags: 6 (ODU1)(ODU2)
dlLOODUFlagsAgree: 6     (ODU1)(ODU2)
dlState: dtUpFree

%%%%%% DT INDEX 1 %%%%%%%
dlTeLinkID: 0
dlLocalInterfaceID: 11
dlRemoteInterfaceID: 11
dlMinimum_Reservable_Bandwidth: 2.5
dlMaximum_Reservable_Bandwidth: 10
dlHOODTUk: 2
dlTS: 2.5
dlLOODUFlags: 6 (ODU1)(ODU2)
dlLOODUFlagsAgree: 6     (ODU1)(ODU2)
dlState: dtUpFree

%%%%%% DT INDEX 2 %%%%%%%
dlTeLinkID: 1
dlLocalInterfaceID: 17
dlRemoteInterfaceID: 17
dlMinimum_Reservable_Bandwidth: 10
dlMaximum_Reservable_Bandwidth: 10
dlHOODTUk: 2
dlTS: 2.5
dlLOODUFlags: 6 (ODU1)(ODU2)
dlLOODUFlagsAgree: 4     (ODU2)
dlState: dtUpFree

%%%%%% DT INDEX 3 %%%%%%%
dlTeLinkID: 1
dlLocalInterfaceID: 23
dlRemoteInterfaceID: 23
dlMinimum_Reservable_Bandwidth: 10
dlMaximum_Reservable_Bandwidth: 10
dlHOODTUk: 2
dlTS: 2.5
dlLOODUFlags: 6 (ODU1)(ODU2)
dlLOODUFlagsAgree: 4     (ODU2)
dlState: dtUpFree
```

Figura 6.8 - Enlace de dados contidos no TE Link de ODU.

Na Figura 6.9 são mostrados os enlaces referentes aos comprimentos de onda contidos no *TE Link* 2 (OTS). Essas informações são extraídas dos parâmetros da rede simulada de acordo com a quantidade de comprimentos de onda utilizados em cada NE. Seus identificadores local e remoto são os mesmos, pois, é considerado o comprimento de onda em si, não havendo conversão deste.

```
%%%%%% DT INDEX 4 %%%%%%%    %%%%%% DT INDEX 5 %%%%%%%
dlTeLinkID: 2                dlTeLinkID: 2
dlLocalInterfaceID: 1        dlLocalInterfaceID: 2
dlRemoteInterfaceID: 1       dlRemoteInterfaceID: 2
dlwavelength: 1              dlwavelength: 2
dlState: dtUpFree            dlState: dtUpFree

%%%%%% DT INDEX 6 %%%%%%%    %%%%%% DT INDEX 7 %%%%%%%
dlTeLinkID: 2                dlTeLinkID: 2
dlLocalInterfaceID: 3        dlLocalInterfaceID: 4
dlRemoteInterfaceID: 3       dlRemoteInterfaceID: 4
dlwavelength: 3              dlwavelength: 4
dlState: dtUpFree            dlState: dtUpFree
```

Figura 6.9 - Enlace de dados contidos no TE Link de OTS.

6.2 CASO DE TESTE 2

No segundo caso de teste é utilizada a topologia representada na Figura 6.10. Essa topologia apresenta dois *hosts*, com quatro comprimentos de onda cada, conectados através de quatro OXCs. Os comprimentos de onda 1 e 2 passam pela rota NE1, NE2 e NE4, enquanto os comprimentos de onda 2 e 3 passam pela rota NE1, NE3 e NE4.

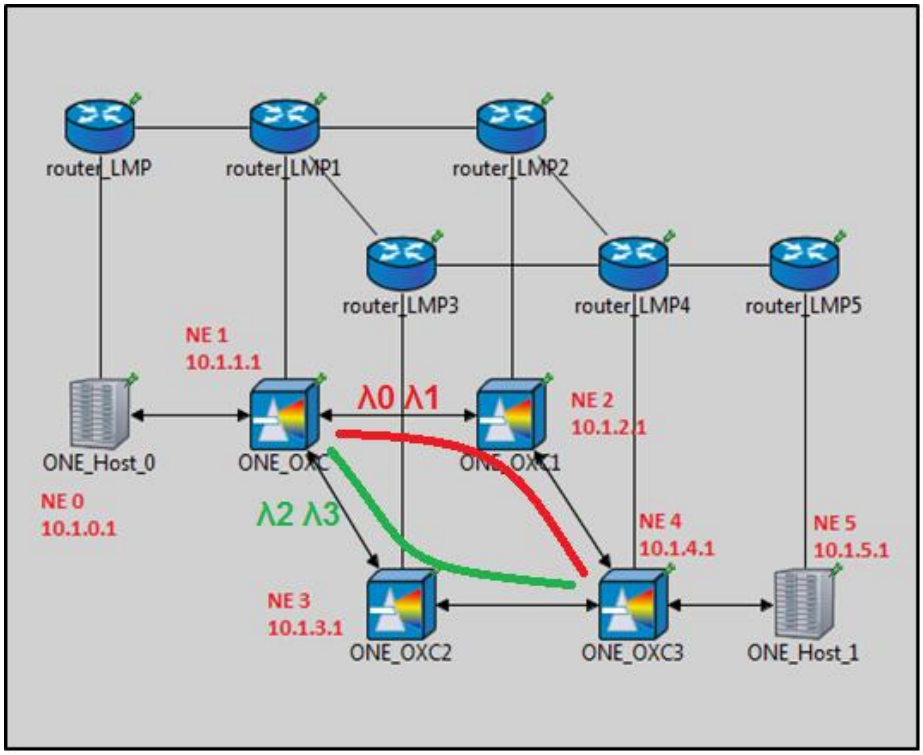

Figura 6.10 - Topologia para o Caso de Teste 2.

O NE0 contém sinais ODU3 (40Gbps) de alta ordem, com capacidade de multiplexação de ODU2 (10Gbps) e ODU1 (2,5Gbps). O NE1 contém sinais ODU3 de alta ordem e suporta multiplexação de ODU2.

6.2.1 Resultados da Descoberta Automática

A Figura 6.11 mostra a topologia em nível de ODU descoberta na rede testada. Apesar dos sinais trafegarem por caminhos diferentes, os OXCs não alteram o sinal em nível de ODU, portanto, a topologia descoberta é composta de quatro enlaces ponto a ponto entre o NE0 e NE5 da rede.

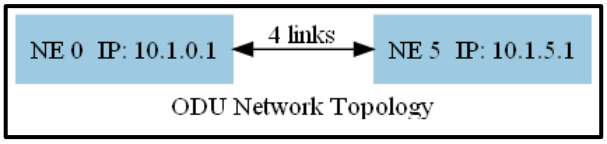

Figura 6.11 - Topologia ODU descoberta para Teste 2.

A Figura 6.12 apresenta, de forma detalhada, o resultado mostrado na Figura 6.11. Observa-se a conexão entre cada terminação ODU identificada entre os dois nós.

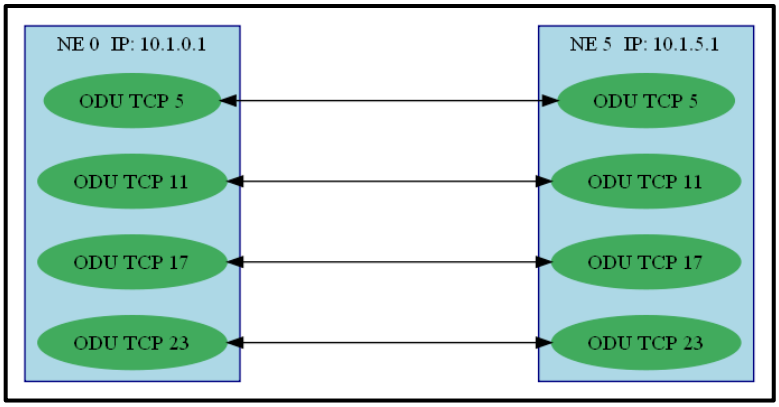

Figura 6.12 - Topologia ODU detalhada para Teste 2.

Na Figura 6.13 é observada a topologia física da rede, ou seja, a topologia em nível de OTS. A topologia em questão apresenta um anel de comutadores de comprimentos de onda (OXCs), sendo dois deles conectados à ONEs que geram/terminam o sinal OTN.

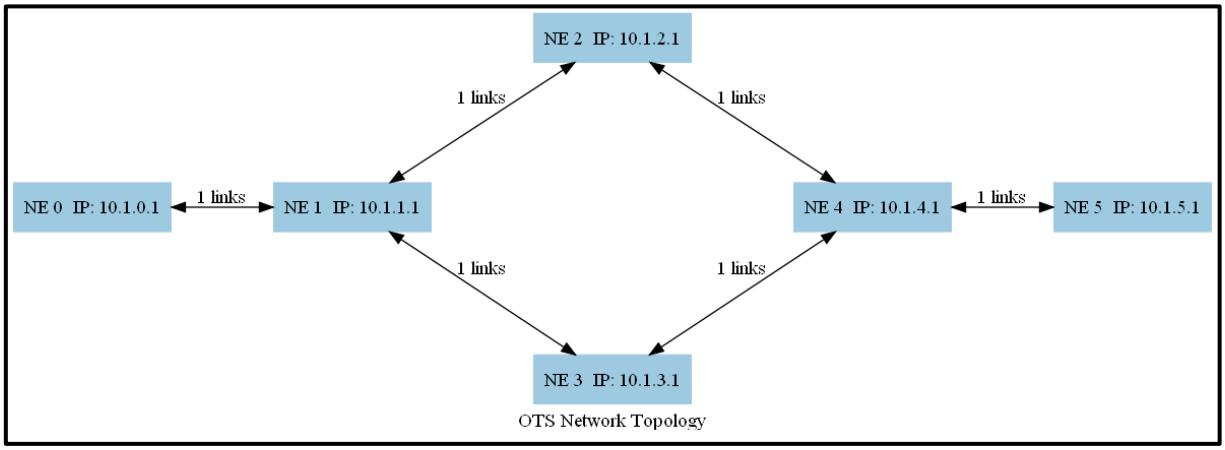

Figura 6.13 - Topologia OTS descoberta para Teste 2.

A Figura 6.14 mostra as conexões entre cada terminação OTS existente nos NEs da rede.

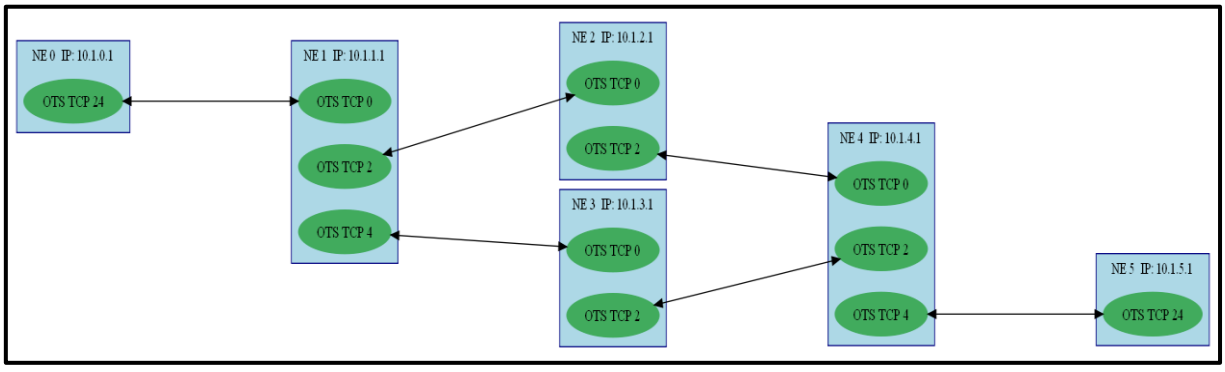

Figura 6.14 - Topologia OTS detalhada para Teste 2.

6.2.2 Resultados do LMP

Na Figura 6.15 são observados os dois *TE Links* criados pelo LMP. O *TE Link* 0 é um enlace ODU entre o NE0 e NE5. O *TE Link* 1 representa o enlace OTS entre os vizinhos diretamente ligados NE0 e NE1.

```
%%%%%% TE INDEX 0 %%%%%%%   %%%%%% TE INDEX 1 %%%%%%%
teID: 0                     teID: 1
teRemoteNodeIP: 10.1.5.1    teRemoteNodeIP: 10.1.1.1
teLocalLinkID: 10.1.0.2     teLocalLinkID: 10.1.0.3
teRemoteLinkID: 10.1.5.2    teRemoteLinkID: 10.1.1.2
teVerification: 1           teVerification: 1
teFaultManagement: 0        teFaultManagement: 0
teState: teUp               teState: teUp
layer: ODU                  layer: OTS
```

Figura 6.15 - Tabela de TE Links do NE0.

Na Figura 6.16 é visto um dos enlaces de dados ODU que compõe o *TE Link* 0. Observa-se a capacidade de multiplexação de ODU1 no NE0, mas não no NE5. Logo, o acordo para o enlace de dados trabalhar apenas com sinais ODU3 e ODU2.

```
%%%%%% DT INDEX 0 %%%%%%%
dlTeLinkID: 0
dlLocalInterfaceID: 5
dlRemoteInterfaceID: 5
dlMinimum_Reservable_Bandwidth: 10
dlMaximum_Reservable_Bandwidth: 40
dlHOODTUk: 3
dlTS: 2.5
dlLOODUFlags: 14     (ODU1)(ODU2)(ODU3)
dlLOODUFlagsAgree: 12    (ODU2)(ODU3)
dlState: dtUpFree
```

Figura 6.16 - Enlace de dados ODU do NE0.

Na Figura 6.17 são observados os *TE Links* criados pelo NE1. Como se trata de um OXC, esse apresenta, apenas, *TE Links* compostos pelos comprimentos de onda entre o NE1 e os NE0, NE2 e NE3.

```
%%%%%% TE INDEX 0 %%%%%%% %%%%%% TE INDEX 1 %%%%%%% %%%%%% TE INDEX 2 %%%%%%%
teID: 0                   teID: 1                   teID: 2
teRemoteNodeIP: 10.1.0.1  teRemoteNodeIP: 10.1.2.1  teRemoteNodeIP: 10.1.3.1
teLocalLinkID: 10.1.1.2   teLocalLinkID: 10.1.1.3   teLocalLinkID: 10.1.1.4
teRemoteLinkID: 10.1.0.3  teRemoteLinkID: 10.1.2.2  teRemoteLinkID: 10.1.3.2
teVerification: 1         teVerification: 1         teVerification: 1
teFaultManagement: 0      teFaultManagement: 0      teFaultManagement: 0
teState: teUp             teState: teUp             teState: teUp
layer: OTS                layer: OTS                layer: OTS
```

Figura 6.17 - TE Links do NE1.

Na Figura 6.18 são mostrados os enlaces de dados referente ao *TE Link* 0 do NE1. Como cada OXC está configurado para trabalhar com quatro comprimentos de onda por fibra, são identificados quatro enlaces de dados entre cada par de OXC na rede testada.

```
%%%%%% DT INDEX 0 %%%%%%%   %%%%%% DT INDEX 1 %%%%%%%
dlTeLinkID: 0               dlTeLinkID: 0
dlLocalInterfaceID: 1       dlLocalInterfaceID: 2
dlRemoteInterfaceID: 1      dlRemoteInterfaceID: 2
dlwavelength: 1             dlwavelength: 2
dlState: dtUpFree           dlState: dtUpFree

%%%%%% DT INDEX 2 %%%%%%%   %%%%%% DT INDEX 3 %%%%%%%
dlTeLinkID: 0               dlTeLinkID: 0
dlLocalInterfaceID: 3       dlLocalInterfaceID: 4
dlRemoteInterfaceID: 3      dlRemoteInterfaceID: 4
dlwavelength: 3             dlwavelength: 4
dlState: dtUpFree           dlState: dtUpFree
```

Figura 6.18 - Enlaces de Dados do TE Link 0 do NE1.

6.3 CONSIDERAÇÕES FINAIS

O objetivo deste capítulo é a validação do procedimento de descoberta automática e do LMP, implementados no simulador de redes OMNeT++. De acordo com os testes realizados, os resultados provenientes da descoberta automática e do protocolo de gerenciamento LMP são os esperados.

Os testes indicaram a correta descoberta de topologia de cada camada, podendo ser mais bem observada através dos grafos gerados, e a troca de informações de

capacidades entre os NEs para, em conjunto, acordarem a capacidade referente aos enlaces de dados entre ambos.

7 CONCLUSÃO

Neste capítulo são discutidas as conclusões do trabalho desenvolvido, apresentando os resultados obtidos e possíveis trabalhos futuros.

7.1 RESULTADOS

O estudo de novas tecnologias e protocolos para sua utilização na prática é uma tarefa, muitas vezes, complexa. Apesar das especificações dos órgãos de padronização como o ITU-T e IETF se mostrarem uma boa base de informações, ao se aplicarem as soluções, surgem questões importantes que serão tratadas de diferentes maneiras em suas implementações.

O objetivo desta dissertação de apresentar e integrar a solução de descoberta automática para redes OTN, com base na recomendação (ITU-T, G.7714, 2005), foi alcançado com sucesso. Foram realizadas diversas modificações com base no trabalho em (Ferrari, 2009) para a integração do protocolo com o plano de transporte da rede OTN. Uma rede DCN utilizando protocolo IP possibilitou a comunicação entre as gerências dos nós da rede.

A elaboração de uma nova arquitetura para gerenciamento dos elementos de rede se mostrou necessária e foi realizada em conjunto ao trabalho desenvolvido em (Tessinari, 2011). Nesta dissertação é desenvolvido o módulo para gerenciamento de NE, realizando sua integração com o processo de descoberta automática e sua comunicação com a rede DCN. Este módulo também é capaz de se comunicar com demais módulos gerentes para configuração de informações de cabeçalho e funcionamento das camadas OTN.

Outro objetivo, a obtenção das informações referentes às capacidades dos enlaces de dados de forma automática, foi alcançado com sucesso utilizando como base os procedimentos do protocolo LMP. A implementação do protocolo seguiu as especificações em (Lang, 2005), sendo que as principais questões observadas

durante sua aplicação à OTN são discutidas na Seção 4.3. A aplicação desse protocolo permite o estabelecimento dos canais de controle e o acordo de capacidade entre os nós, criando a base de informações a ser utilizada pelos demais protocolos do plano de controle GMPLS. Para a adaptação das mensagens do LMP à necessidade das redes OTN, seguiu-se o novo *draft* do IETF (Zhang, et al., 2011), cuja versão final está em desenvolvimento. A integração com a gerência e utilização dos dados obtidos pelo procedimento de descoberta automática são apresentadas na Seção 4.3.1.

A validação da arquitetura de gerência, do procedimento de descoberta automática e do protocolo para gerenciamento dos enlaces foi executada através de redes de testes criadas no simulador OMNeT++. Os resultados obtidos, em forma de grafos de topologias e estruturas de dados obtidas por cada NE, são os esperados de acordo com as configurações impostas nas redes de teste.

7.2 TRABALHOS FUTUROS

Para o contínuo desenvolvimento do *framework* de simulação de redes OTN, o estudo e desenvolvimento de algumas funcionalidades são sugeridos:

- A multiplexação digital, realizada na camada ODU, não é suportada no *framework* atual, sendo uma importante funcionalidade para tornar a simulação mais próxima do real e possibilitando um teste mais eficiente do gerenciamento de enlaces desenvolvido nesta dissertação;
- O desenvolvimento de aplicações de gerência, externas ao simulador, para visualização dos grafos da rede de forma dinâmica e com as informações de capacidades. Vale destacar que o trabalho para comunicação entre o Gerente de NE aqui desenvolvido, com uma aplicação externa com o intuito de observar informações de alarmes ocorridos na rede pode ser encontrado em (Almeida, 2011);
- A criação do plano de controle para a camada digital da rede OTN, que utiliza a base de dados obtidas nesta dissertação e os canais de controle estabelecidos pelo LMP. Esse plano de controle baseado nos protocolos do

GMPLS realiza a alocação de recursos no nível ODU de acordo com as capacidades informadas pelo LMP;

- A implementação dos demais procedimentos opcionais do LMP para verificação dos enlaces de dados e gerenciamento de falhas.

8 REFERÊNCIAS

Almeida, T. M. (2011). *Projeto de Graduação - Interface de comunicação com o plano de gerência OTN no simulador OMNeT++.*

ECI Telecom. (Julho de 2008). White Paper. *Next Generation OTN*.

EXFO. (2006). Application Note. *The G.709 Optical Transport Network - An Overview*. Canadá.

Favoreto, F. P. (2009). *Dissertação de Mestrado - Plano de Controle GMPLS para Redes Ópticas de Transporte.* Universidade Federal do Espírito Santo, Vitória.

Ferrari, F. F. (2009). *Projeto de Graduação - Simulação da Funcionalidade de Descoberta Automática Aplicada às Redes OTN.* Universidade Federal do Espírito Santo, Vitória.

Ferrari, F. F., Frasson, A. M., & Garcia, A. S. (2010). Descoberta Automática em Redes Ópticas de Transporte. *MOMAG2010*.

Frigini, F. N. (2010). *Projeto de Graduação - Modelagem Das Camadas Digitais De Uma Rede Óptica De Transporte No Simulador OMNeT++ De Acordo Com A Recomendação ITU-T G.798.* Universidade Federal do Espírito Santo, Vitória.

Iniewski, K., McCrosky, C., & Minoli, D. (2008). *Network Infrastructure And Architecture.* New Jersey: Wiley Interscience.

ITU-T, T. S. (2000, Março). G.805. *Generic Functional Architecture of Transport*.

ITU-T, T. S. (2001, Novembro). G.872. *Architecture of Optical Transport Networks*.

ITU-T, T. S. (2005, Agosto). G.7714. *Generalized Automatic Discovery for Transport Entities*.

ITU-T, T. S. (2006, Junho). G.8080. *Architecture for the Automatically Switched Optical Network (ASON)*.

ITU-T, T. S. (Março de 2008). G.874. *Management aspects of the optical transport network element*.

ITU-T, T. S. (2009). G.709. *Interfaces for the Optical Transport Network (OTN)*.

ITU-T, T. S. (2010). G.7714.1. *Protocol for automatic discovery in SDH and OTN networks*, 44.

ITU-T, T. S. (2010). G.798. *Characteristics of optical transport network hierarchy equipment functional blocks*.

Lang, J. (Setembro de 2005). IETF RFC4204. *The Link Management Protocol (LMP)*.

Papadimitriou, D., Poppe, F., & Rousseau, B. (2004). Application of the Link Management Protocol to Discovery and Forwarding Adjacencies. *Photonic Network Communications*.

Perelló, J., Escalona, E., Spadaro, S., Comellas, J., & Junyent, G. (Outubro de 2007). Resource Discovery in ASON/GMPLS Transport Networks. *IEEE Communications Magazine*.

Ramaswami, R., & Sivarajan, K. N. (2002). *Optical Networks - A Pratical Perspective.* San Francisco: Morgan Kaufmann.

Tessinari, R. S. (2009). *Projeto de Graduação - Mapeamento de Equipamentos Ópticos e Modelagem de Redes OTN no Simulador OMNeT++ de acordo com a Recomendação ITU-T G.798.* Universidade Federal do Espírito Santo, Vitória.

Tessinari, R. S. (2011). *Dissertação de Mestrado - Integração do Plano de Transporte com os Planos de Controle e de Gerência em Redes OTN: Uma Abordagem Via Simulação.* Vitória.

Varga, A. (s.d.). *Omnet++ Version 4.0 User Manual.* Acesso em 15 de Janeiro de 2009, disponível em Site do Simulador de Eventos Discretos Omnet++: http://omnetpp.org/doc/omnetpp40/manual/usman.html

Varga, A., & Acadêmica, C. (2011). *INET Framework main page*. Acesso em 2011, disponível em Site official do INET.

Zhang, F., Li, D., Ceccarelli, D., Caviglia, D., Zhang, G., Grandi, P., et al. (2011). *IETF draft: Link Management Protocol (LMP) extensions for G.709 Optical Transport Networks.*

Zhou, Z., & Chi, C. (2004). Link Management in Next Generation Optical Networks. *Proceedings of the 3rd International Conference on Networking*.

ANEXO I – MÁQUINA DE ESTADOS DO LMP - CANAL DE CONTROLE

O canal de controle quando em operação se apresenta em algum estado descrito a seguir. Cada estado está associado a certa condição do canal de controle e quase sempre está associado com um tipo específico de mensagem LMP que é periodicamente transmitida.

Os estados definidos para a FSM do canal de controle são:

- *Down*: É o estado inicial do canal de controle. Nesse estado, nenhuma tentativa é feita para estabelecer o canal de controle e nenhuma mensagem é enviada. Os valores iniciais dos parâmetros do canal de controle devem ser estabelecidos.
- *ConfSnd*: É o estado de negociação dos parâmetros. Envia periodicamente mensagens de configuração (*Config*) e aguarda como resposta mensagens *ConfigAck* e *ConfigNack*. A FSM só muda de estado quando todos os parâmetros estão devidamente negociados.
- *ConfRcv*: Estado no qual os parâmetros estão sendo negociados. Ele permanece nesse estado até que todos os parâmetros recebidos do nó remoto sejam aceitos. Quando todos são aceitos, a FSM muda para o estado ativo.
- *Active*: Nesse estado, o nó envia mensagens *Hello* periodicamente e espera por mensagens *Hello* válidas. Quando uma mensagem *Hello* válida é recebida, a FSM pode mudar para o estado Up.
- *Up*: O canal de controle encontra-se no estado operacional. Ele envia e recebe mensagens *Hello* válidas.
- *GoingDown*: O canal pode entrar nesse estado por uma determinação administrativa. Enquanto permanecer nesse estado, o canal ativa o bit de canal *Down* em todas as mensagens enviadas por ele.

Os eventos do canal de controle são gerados por protocolos subjacentes e módulos de *software*, a saber, rotinas de processamento de pacotes e a FSM associada ao TE *Link.* Os eventos possíveis para o canal de controle são:

1. *evBringUp:* É um gatilho externo que indica que a negociação de parâmetros precisa ser iniciada. Dependendo da configuração ele vai ser disparado ou: o envio de uma mensagem *Config* (a) ou um período de espera por uma mensagem *Config* (b).
2. *evCCDn:* Esse evento é gerado quando há uma indicação que o canal de controle não está mais ativo.
3. *evConfigDone:* Evento que indica que a mensagem *ConfigAck* foi recebida e os parâmetros foram acordados.
4. *evConfigErr:* Evento que indica que a mensagem *ConfigNack* foi recebida e que os parâmetros não foram acordados.
5. *evNewConfigOK:* Evento que indica que uma nova mensagem de configuração foi recebida do vizinho e que os parâmetros foram acordados.
6. *evNewConfigErr:* Evento que indica que uma nova mensagem de configuração foi recebida do seu vizinho e foi rejeitada com uma mensagem *ConfigNack.*
7. *evContenWin:* Evento que indica que ao mesmo tempo em que uma mensagem de configuração é enviada ao nó vizinho, uma é recebida do nó vizinho e o nó local ganha a disputa. Como resultado, a mensagem recebida é ignorada.
8. *evContenLost:* Evento que indica uma situação parecida com a anterior, porém, o nó local perde a disputa. Assim ele deve enviar uma mensagem *ConfigAck (a)* ou *ConfigNack (b)* de acordo com a aceitação dos parâmetros.
9. *evAdminDown:* Evento que indica que o administrador solicitou a queda do canal para fins administrativos.
10. *evNbrGoesDn:* Evento que indica que uma mensagem com a *flag ControlChannelDown* foi recebida de um vizinho.
11. *evHelloRcvd:* Evento que indica que um pacote *Hello* com o *RcvNum* esperado foi recebido.

12. *evHoldTimer:* Evento que indica que o intervalo *HelloDeadInterval* foi expirado e a mensagem *Hello* foi recebida. Isso move o canal de controle para o estado de negociação e ele pode: enviar mensagens de configuração periódicas (a) ou esperar receber mensagens de configuração do nó remoto (b).
13. *evSeqNumErr:* Evento que indica que uma mensagem *Hello* foi recebida com um valor inesperado de *SeqNum* e foi descartada.
14. *evReconfig:* Os parâmetros do canal de controle foram reconfigurados e precisam de renegociação.
15. *evConfRet:* Evento que indica que o tempo de retransmissão foi atingido e a mensagem de configuração será retransmitida.
16. *evHelloRet:* Evento que indica que o intervalo *HelloInterval* foi cumprido e uma nova mensagem *Hello* será enviada.
17. *evDownTimer:* Evento que indica que o tempo foi expirado e não foi recebida nenhuma mensagem com a *flag ControlChannelDown* setada.

A figura I.1 a seguir mostra o funcionamento da máquina de estados do canal de controle em forma de um diagrama da transição de estados.

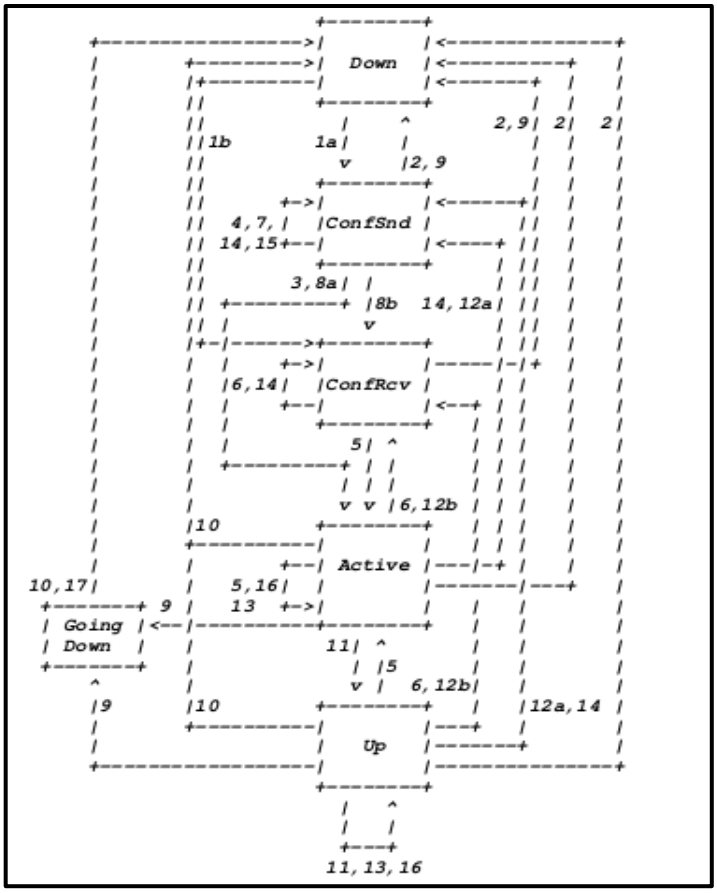

Figura I. 1 - FSM do Canal de Controle.

ANEXO II – MÁQUINA DE ESTADOS DO LMP - ENLACE DE DADOS

A máquina de estados do enlace de dados descreve as operações lógicas e os estados de um enlace de dados com um TE *Link.* O enlace de dados pode estar ou em modo ativo (transmitindo), onde as mensagens de teste são enviadas a partir dele, ou em modo passivo (recebendo) onde as mensagens de teste são recebidas por ele. Para ficar mais claro, são definidas duas FSM, uma para o modo ativo e outra para o modo passivo. Contudo o mesmo conjunto de estados e eventos é definido.

A operação da máquina de estados do enlace de dados é definida em função dos estados e dos eventos. Os possíveis estados são:

- *Down:* O enlace não está em serviço.
- *Test:* O enlace de dados está sendo testado. Uma mensagem de teste está sendo periodicamente enviada pelo enlace de dados.
- *PasvTest:* O enlace de dados está sendo verificado pelas mensagens de teste que chegam.
- *Up/Free:* Os enlaces de dados foram testados e estão disponíveis para uso (*in-service*). Eles ainda não se encontram alocados para tráfego de usuários.
- *Up/Alloc:* Os enlaces estão operacionais e alocados para tráfego de usuários.

Os eventos do enlace de dados são gerados pelo processamento de pacotes, pela FSM do canal de controle ou do TE *Link* associado. Todos os possíveis eventos são:

1. *evCCUp:* Evento que indica que o primeiro canal de controle ativo foi estabelecido.
2. *evCCDown*: Evento que indica que a conectividade com o vizinho LMP é perdida, pois o ultimo canal de controle ativo entre os dois vizinhos falha.
3. *evStartTst:* É um evento externo que desencadeia o envio de mensagens de teste pelos enlaces de dados.

4. *evStartPsv:* É um evento externo que desencadeia a ação de escutar as mensagens de teste que vem pelo enlace de dados.
5. *evTestOK:* Evento que indica que a verificação dos enlaces de dados foi feita com sucesso e o enlace pode ser usado como caminho. Pode indicar duas situações:
    a. O processo de verificação de enlace foi feita com sucesso e uma mensagem *TestStatusSuccess* foi recebida pelo canal de controle.
    b. Indica que o enlace está pronto para o estabelecimento de caminho, mas o procedimento de verificação não foi usado. Para a sinalização *in-band* o próprio procedimento para o estabelecimento do canal de controle é suficiente para verificar o enlace.
6. *evTestRcv:* Evento que indica que uma mensagem de teste foi recebida pela porta de dados e uma mensagem *TestStatusSuccess* foi enviada pelo canal de controle.
7. *evTestFail:* Evento que indica que o processo de verificação retornou resultados negativos. Duas possíveis situações ocorrem:
    a. Uma mensagem *TestStatusFailure* foi recebida.
    b. O processo de verificação terminou sem o recebimento de uma mensagem *TestStatusSucces* ou *TestStatusFailure* para o link de dados.
8. *evPsvTestFail:* Evento que indica que a verificação de enlace retornou resultados negativos. Podem acontecer duas situações possíveis:
    a. O tempo de *VerifyDeadInterval* expirou.
    b. O procedimento de verificação foi encerrado e o tempo de *VerifyDeadInterval* ainda não expirou.
9. *evLnkAlloc:* Evento que indica que o enlace de dados foi alocado.
10. *evLnkDealloc:* Evento que indica que o enlace de dados foi desalocado.
11. *evTestRet:* O tempo de retransmissão foi expirado e a mensagem de teste é reenviada.
12. *evSummaryFail:* Evento que indica que a mensagem de *LinkSummary* não foi atingida pela porta de dados.
13. *evLocalizeFail:* Evento que indica que uma falha foi localizada no enlace de dados.

14. *evDCDown:* Evento que indica que o canal de controle não está disponível.

A figura II.1 a seguir mostra o funcionamento da máquina de estados do enlace de dados no modo ativo em forma de um diagrama da transição de estados.

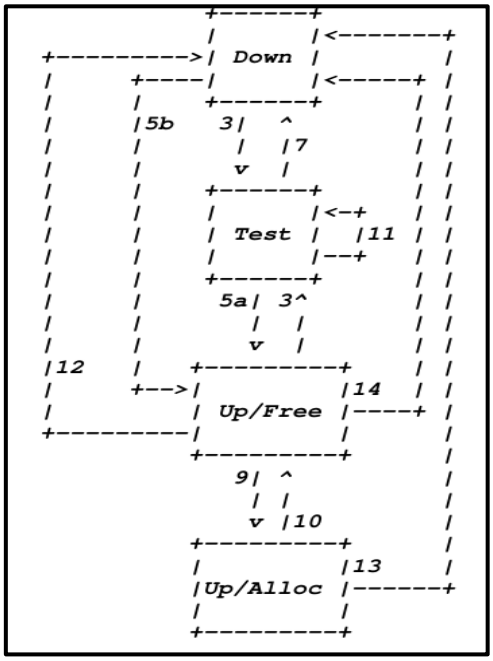

Figura II. 1 - FSM do enlace de dados modo ativo.

A figura II.2 a seguir mostra o funcionamento da máquina de estados do enlace de dados no modo passivo em forma de um diagrama da transição de estados.

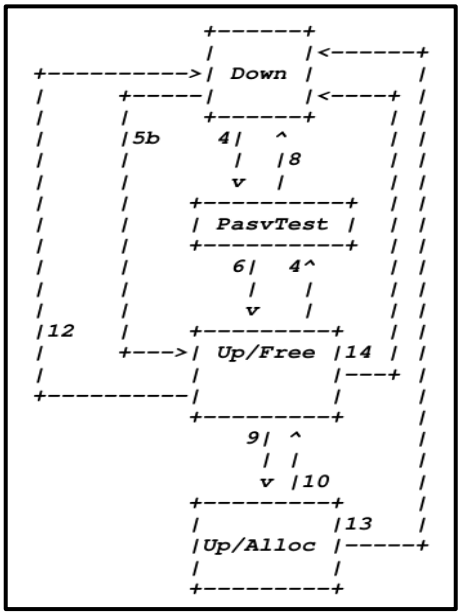

Figura II. 2 – FSM do enlace de dados modo passivo.