Usando SQL e OPLScript na Modelagem de Redes Multicamadas

Leandro A. Rodrigues*, Saulo Bortolon**, Anilton Salles**

*Mestrando em Engenharia Elétrica, **Departamento de Informática

UFES – C.P. 01-9011, CEP: 29060-970, Vitória, ES – Brasil

Resumo: A rede de telecomunicações multicamada é a arquitetura de transporte predominante. Circuitos T1/E1, SDH de baixa e de alta ordem, *wavelength division multiplexing* (WDM), pares e cabos de fibra ótica estão sobrepostos uns sobre os outros. Este artigo propõe um esquema relacional de banco de dados que representa todas as camadas desta rede e proporciona uma maneira fácil de definir e construir modelos de otimização. Dois importantes modelos de otimização são descritos: roteamento de pares de fibra e roteamento de VC em STM. Os modelos de redes de fluxo multiproduto são implementados através OPLScript. O primeiro problema é modelado através da abordagem arco-caminho enquanto que o segundo, através da abordagem nó-arco. Essas formulações são baseadas no esquema de banco de dados proposto.

Palavras chave**: Banco de Dados, Planejamento de Telecomunicações, Otimização.**

I. INTRODUÇÃO

O conceito de camadas tem sido amplamente usado em telecomunicações como uma abordagem de particionamento. Uma arquitetura cliente-servidor deve ser usada para descrever os relacionamentos entre as ruas, galerias de dutos, cabos de fibra ótica, pares de fibra ótica, comprimentos de onda, e assim por diante. Este artigo descreve estes relacionamentos através de um modelo de dados de entidade-relacionamento. São construídos modelos de otimização usando OPLScript e resolvidos alguns problemas de planejamento da rede de transporte metropolitana, tendo os dados provenientes do banco de dados como suporte. OPLScript é uma linguagem de modelagem de problemas de otimização que foi desenvolvida pela ILOG. Todos os modelos construídos foram derivados do problema de projeto de uma rede de fluxo multiproduto, que é um problema comum no planejamento de telecomunicações.

A Figura 1 mostra cabos óticos conectando dispositivos intermediários óticos (DIO). Existem muitas questões a serem respondidas pelos projetistas de rede, por exemplo: **(1)** ***Quantos pares de fibras serão necessários para ligar os centros de fios A e B?*** **(2)** ***Quais cabos de fibra devem ser usados para rotear os pares necessários?*** **(3)** ***Existe a possibilidade de se usar fibras extras?*** **(4)** ***Novos cabos precisam ser lançados?*** **(5)** ***Os pares de fibra do esquema de trabalho&proteção estão roteados em cabos diferentes?***

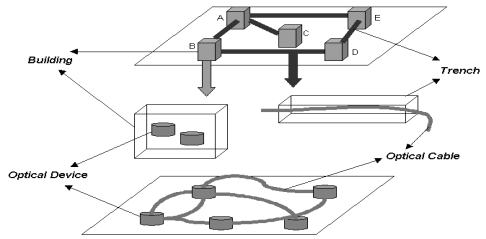

Figura 1 – Uma Rede Multicamada Simples

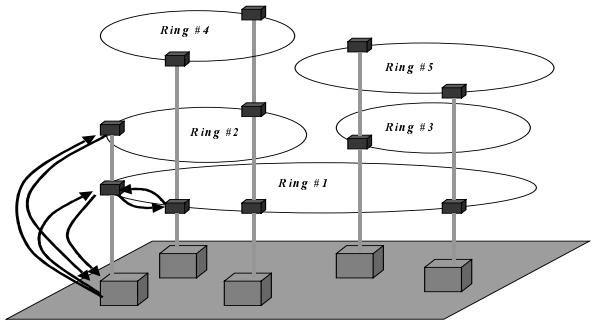

Figura 2 – Uma Proposta de Anéis SDH

A Figura 2 mostra anéis candidatos SDH de alta ordem, propostos como possíveis caminhos para conectar equipamentos *cross-connectors* de baixa ordem (SDxC 4/1). Existe uma matriz de demandas VC-4 entre cada par de SDxC´s que tem que ser roteada através dos anéis. Estes anéis podem ser STM-1, -4, -16 ou até mesmo, STM-64, indicando quantos VC-4 cada um pode transportar. Projetistas têm que escolher quais anéis serão implementados de forma a obter o menor custo e menor tráfego inter-anel [2].

Todas estas questões podem ser respondidas com a ajuda dos modelos de programação linear inteira mista, com centenas ou até milhares de variáveis. Dois desafios principais são colocados para os operadores de telecomunicações: (1) manter atualizados os dados sobre as facilidades instaladas da rede e, (2) possuir tempo suficiente para construir estes grandes modelos de otimização. A fim de ajudar na solução desses desafios, este artigo propõe que toda a informação seja armazenada em um banco de dados existente e que se use uma linguagem de programação de otimização para construir os modelos de projeto de rede.

II. REDE DE TRANSPORTE MULTICAMADA

De um ponto de vista tecnológico, a rede de transporte é construída sobre várias diferentes

camadas e cada uma destas camadas tem seus próprios nós e arcos. Existe um relacionamento do tipo cliente-servidor entre as camadas adjacentes, por exemplo: muitos pares de fibra usam um cabo de fibra, um *cross-connector* WDM está num centro de fios, e assim por diante.

Em redes de grande porte, o processo de digitalização permite o uso destas camadas proporcionando uma economia de escala significativa, embora levando a problemas sérios de *survivability* [3]. Diferentes decisões que devem ser tomadas pelos projetistas são abordadas – é importante observar que decisões sobre uma camada dependem de decisões tomadas em outras camadas; como segue: **(1)** ***Onde podem ser instalados novos nós de uma dada camada e qual será a capacidade deles?*** Quando um projetista se refere a um nó, ele pode estar pensando em um centro de fios (CF); um dispositivo intermediário ótico (DIO); um *cross-connector* WDM (WDMxC); um *cross-connector* SDH (SDxC); um multiplexador *add-drop* (ADM); um multiplexador ótico de linha remoto (OLTM) ou até mesmo, em um *switch* IP. Cada um destes será instalado e dimensionado de acordo com as decisões de projeto; **(2)** ***Onde podem ser instalados novos arcos de uma dada camada e qual será a capacidade deles?*** De acordo com a camada da rede, um arco pode ser uma galeria ou um cabo de fibra ou um par de fibras ou um *link* WDM transportando um sinal STM-n (*Synchronous Transport Module*) ou um *container* VC-4 (SDH *Virtual Container*) ; um VC-12 ou um *link* de 64Kbps entre dois centros de fios. Dados os nós de cada camada da rede, arcos conectando-os devem ser projetados. Cada arco possui um custo associado, assim como, capacidade; **(3)** ***Como devem ser alocadas as rotas de uma camada cliente para uma camada de suporte?*** Esta pergunta pode ser desdobrada em outras, como (3.1) Como pode ser transportado um *link* de 64Kbps entre dois centros de fios, roteando-os através de canais T1/E1? (3.2) Como podem ser roteados canais T1/E1 através de *containers* VC-4? (3.3) Como podem ser roteados os *containers* VC-4 através dos STM-n's?, e assim por diante; até o roteamento de cabos de fibras através das redes de galerias. Em todas estas decisões, o projetista está procurando por uma solução gerenciável, eficiente, resiliente e de baixo custo.

Assim, a rede de transporte possui uma estrutura em camadas com conexões de uma camada, multiplexadas ou roteadas através outras conexões de uma outra camada adjacente. Em outras palavras, a camada de suporte é usada para rotear as demandas de sua camada cliente (e.g., galerias são usadas para rotear um conjunto de cabos de fibras e os cabos são usados para rotear os pares de fibra).

Esta estrutura das multicamadas é mostrada na Figura 3. O lado direito da figura mostra como os arcos de uma camada cliente são roteados através dos arcos de suas camadas de suporte. O lado esquerdo mostra como os nós de *switching* das várias camadas são conectados dentro de um centro de fios (implementação das conexões entre os nós das diferentes camadas). Observe que estes arcos conectam os nós da mesma camada ou de diferentes camadas. No primeiro caso, o par de nós está localizado em centros de fios diferentes; no segundo caso, o par de nós está localizado no mesmo centro de fios.

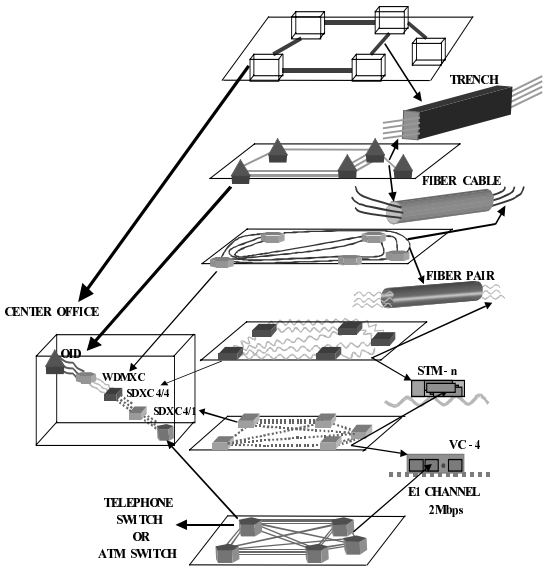

Figura 3 – Rede Multicamada: Camada Cliente é Roteada Através da Camada Suporte

Cada nó e cada arco desta rede multicamada possui custo, capacidade e modularidade associados. A capacidade é usualmente baseada em padrões da indústria: um cabo de fibra ótica pode transportar 8, 16 ou 32 pares de fibras; um dispositivo SDxC pode chavear 64, 256 ou 1024 canais E1. A estrutura de custos admite o tratamendo de linearidade, custos fixos e componentes modulares: custo da instalação de cabos de fibra varia linearmente com seu tamanho; um centro de fios tem um custo fixo de construção; *switches* SDxC podem ter sua capacidade aumentada em módulos.

Por razões de confiabilidade, deve também existir requisitos de roteamento de arcos disjuntos, levando em conta, muitas camadas simultaneamente. Dois pares de fibra configurados no esquema de trabalho&proteção [13] devem ser roteados por diferentes cabos e diferentes galerias. Um anel STM-16 protegido por *line switching* (anel bi-direcional) deve ter seus módulos de segurança STM-16 roteados através de comprimentos de onda que usem fibras, cabos e galerias divergentes.

Todos estes requisitos de rede e todos estes procedimentos de roteamento multicamadas demandam de uma ferramenta computacional de planejamento, provida de interfaces gráficas e modelos de fluxo em redes, ditos, *ready-to-use*. Essas duas facilidades computacionais dependem fortemente de dados de entrada que devem ser fornecidos por uma base de dados. Tal base de dados será descrita na próxima seção.

III. UM ESQUEMA DE BANCO DE DADOS PARA O PLANEJAMENTO DA REDE

Afim de representar todas as características descritas na seção anterior, este artigo propõe o

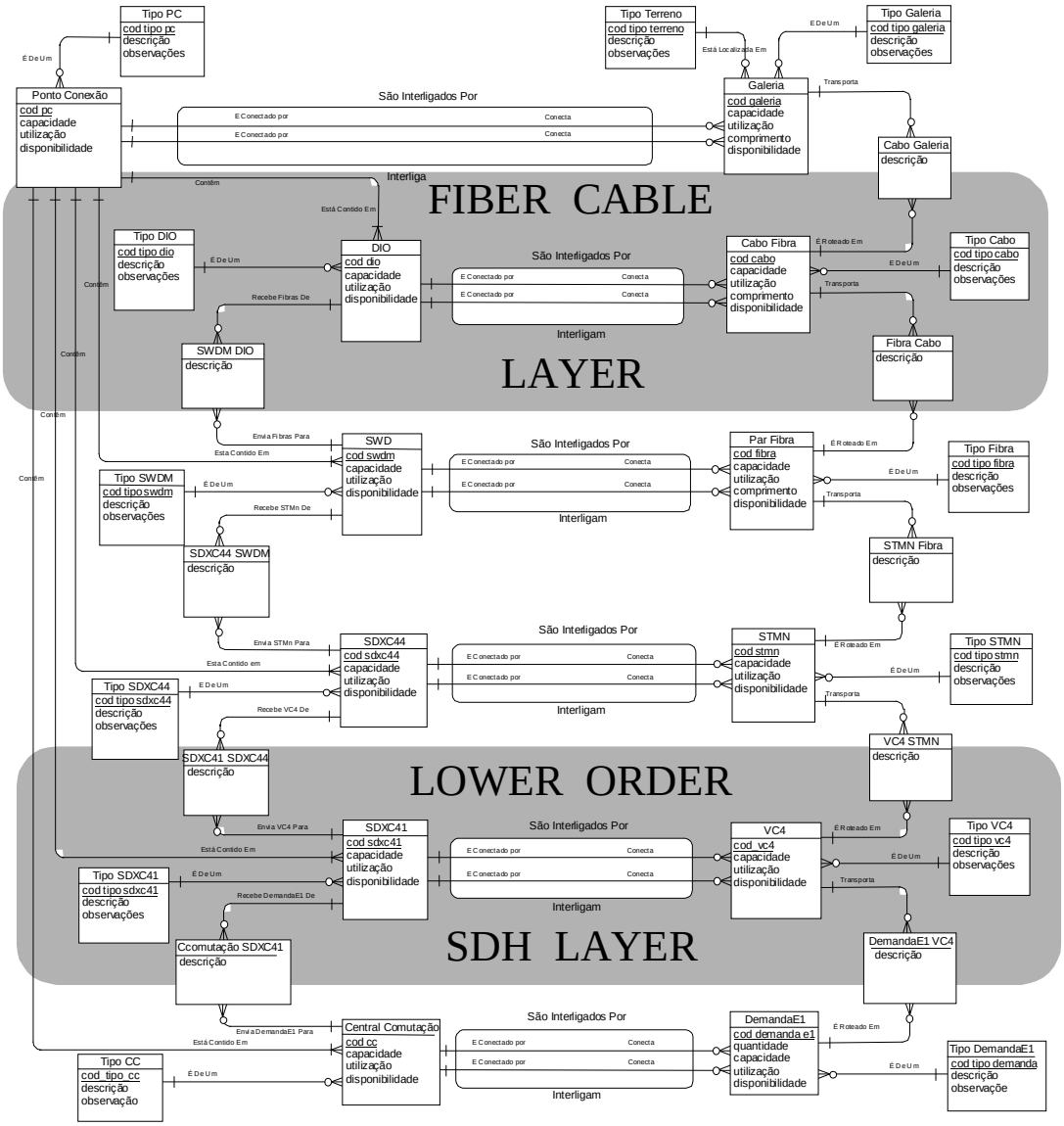

Figura 4 – Esquema de Banco de Dados Multicamada

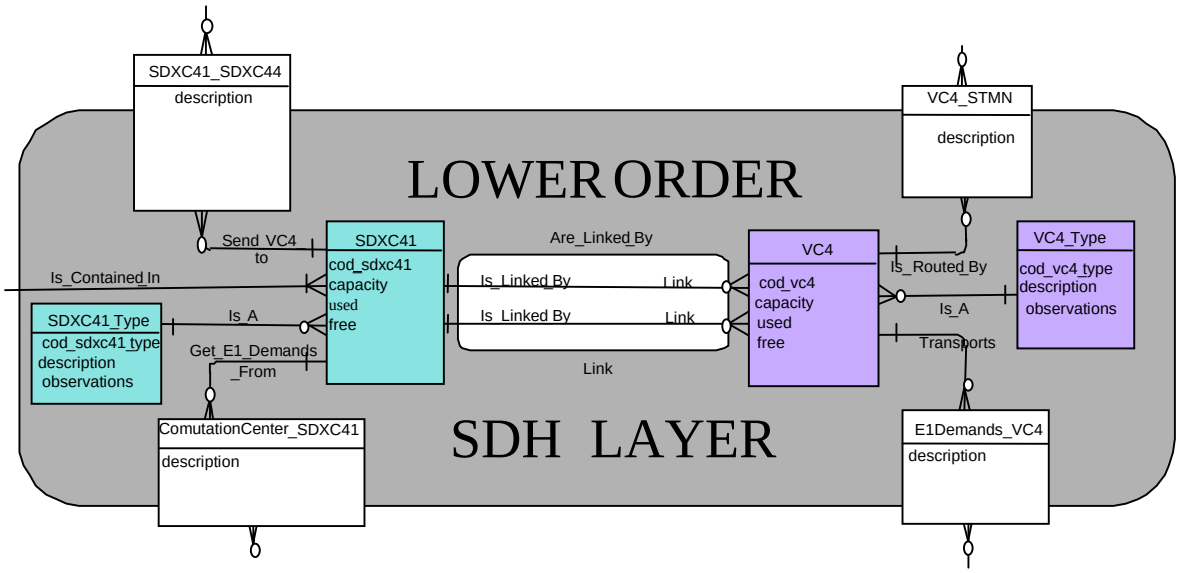

Figura 5 – Esquema da Camada SDH de Baixa Ordem – Extraído da Figura 4

esquema de banco de dados mostrado na Figura 4. Olhando para este esquema de entidade-relacionamento (E-R), alguém pode pensar que cada camada da Figura 3 possui uma "*data layer*" correspondente na Figura 4. Mais, a Figura 5 descreve detalhadamente a camada SDH de baixa ordem que foi *highlighted* na Figura 4 [9].

A Figura 5 descreve que:

- Cada SDxC 4/1 origina um ou mais VC-4 e cada VC-4 conecta dois SDxC 4/1
- Cada VC-4 é roteado por uma seqüência de um ou mais STM-n (seu caminho de mais alta ordem) e cada STM-n roteia até n VC-4 (*payload*)
- Cada VC-4 pode ser enviado de um SDxC 4/1 de origem para SDxC 4/4 de alta ordem
- Cada E1 pode ser enviado de um *switch* de voz para um SDxC 4/1
- Cada E1 é roteado por uma seqüência de um ou mais VC-4 (seu caminho de mais baixa ordem) e cada VC-4 roteia até 63 E1 (*payload*)
- Cada SDxC 4/1 está localizado em um centro de fios

Todas estas relações possuem suas equivalências em cada camada, conforme mostrado na Figura 3. A vantagem de tal esquema é sua capacidade de incorporar novas camadas sempre que necessário. Um projetista de rede pode incluir camadas ATM ou IP no modelo apresentado com poucas mudanças em outras camadas.

IV. MODELOS DE REDE DE FLUXO MULTIPRODUTO

O uso de modelos de otimização em telecomunicações é um prática na indústria para problemas de planejamento de redes desde o início da década de 60 [6]. Em redes multicamadas, todos os nós e arcos possuem custos e capacidades associados. Existem muitos modelos úteis de otimização, como segue:

- Quantos pares de fibras devem ser instalados para conectar centros de fios? Onde devem ser colocados os novos cabos, se preciso for? Como devem ser conectadas as extremidades dos pares de fibra de folga nos DIO's, de modo a criar caminhos óticos de um CF de origem para um CF de destino?
- Quantos VC-4s devem ser instalados para conectar SDxC 4/1? Quais anéis devem ser instalados, se preciso for? Como devem ser roteados os VC-4s através dos anéis de modo a minimizar o custo total e as conexões entre anéis?

Estes e muitos outros modelos de otimização de projeto de redes são baseados nas formulações do problema de fluxo em redes multiproduto [1][7]. Existem duas abordagens possíveis para a modelagem do fluxo multiproduto: arco-caminho e nó-arco. A primeira necessita que um conjunto de caminhos "candidatos" seja descrito antes que se decida como rotear as demandas através dos mesmos. A segunda permite que todos os possíveis caminhos sejam "experimentados", sem se preocupar com qual rota usam. A Figura 6 exemplifica uma rede multiproduto que transporta k = 3 produtos através de n = 6 nós usando m = 10 arcos. Cada arco possui capacidade e custo de instalação, e cada produto possui um volume demandado.

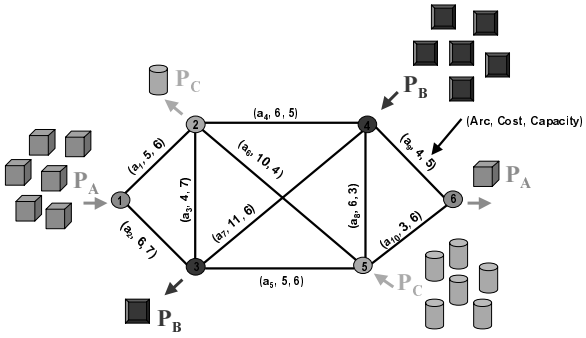

Figura 6 – Exemplo de uma Rede de Fluxo Multiproduto

A formulação abaixo corresponde a um modelo matemático genérico, do tipo arco-caminho, capaz de representar uma rede como a descrita acima e foi extraída de [7]:

$$Minimize \ \sum_j C^j x_j \tag{1}$$

sujeito a:

$$\sum_k \sum_r a_j^{r,k} \cdot f_{r,k} \le x_j b_j \quad \forall j = 1, ..., m \tag{2}$$

$$\sum_r f_{r,k} = d_k \quad \forall k = 1, ..., p \tag{3}$$

$$f_{r,k} \ge 0 \quad \forall r = 1, ..., n_k, k = 1, ..., p \tag{4}$$

$$x_j = 0/1$$

aonde:

$$a_i^{j,k} = \begin{cases} 1, & se\ arco\ i\ pertence\ a\ rota\ R_j^k \\ 0, & caso\_contrário \end{cases}$$

R_j^k rota j, usada somente pelo produto j

p número de produtos a serem roteados

$x_{j,k}$ total de produtos k passando pelo arco j

$C^{j,k}$ custo do fluxo de uma unidade do produto k pelo arco j .

b_i máximo fluxo através do arco i.

d_k valor total do produto k a ser transportado.

Restrições (1) até (3) podem ser entendidas como:

(1) Escolhe os arcos que serão construídos, de modo a minimizar o custo total dos arcos

(2) Cada arco a_i, $i = 1, ..., m$, deve ter seu fluxo igual ou menor que b_i

(3) Cada produto k deve ter todos seus d_k unidades transportadas dos nós de origem aos de destino

Uma formulação nó-arco alternativa é:

$$Minimize \ \sum_j C^j x_j \tag{5}$$

sujeito a:

$$\sum_j a_j^i g_{j,k} = d_{i,k} \quad \forall i = 1, ..., n, \forall k = 1, ..., p \tag{6}$$

$$\sum_k g_{j,k} \le b_j \quad \forall j = 1, ..., m \tag{7}$$

$g_{j,k} \geq 0 \quad \forall j = 1, ..., n, k = 1, ..., p$ (8)

$x_j = 0/1$

aonde:

$$a_i^j = \begin{cases} 1, & se\ arco\ \ j\ iniciar\ \ no\ nó\ i \\ -1, & se\ arco\ \ j\ finalizar\ \ no\ nó\ i \\ 0, & caso\ \_\ contrário \end{cases}$$

p número de produtos a serem roteados

$g_{j,k}$ total do produto k escoando pelo arco j

$C_{j,k}$ custo de transporte de uma unidade do produto k através do arco j

b_i fluxo máximo através do arco i

d_k valor total do produto k a ser transportado

Restrições (1) até (3) podem ser entendidas como:

(1) Escolhe os arcos que serão construídos, de modo a minimizar o custo total dos arcos

(2) Cada arco a_i, $i = 1, ..., m$, deve ter seu fluxo igual ou menor que b_i

(3) Cada produto k deve ter todos seus d_k unidades transportadas dos nós de origem aos de destino

V. ROTEAMENTO DE PARES DE FIBRAS SOBRE CABOS

Este problema pode ser modelado como um problema de fluxo multiproduto arco-caminho, como descrito abaixo:

Rede de Telecomunicações	Modelado como ...
Pares de Fibra	Produtos
Cabos de Fibra	Arcos
Dispositivo Intermediário Ótico	Nós

O programa OPLScript que faz o mapeamento do modelo matemático arco-caminho descrito na seção IV é apresentado na Figura 7. Observe que o script usa conexões com o banco de dados de modo a obter dados da rede real e a atualizar o projeto da rede de acordo com os resultados da otimização.

```
/* Database structures */
struct Node {int+ node;int+ capacity;int+ utilization;}; // OID map
struct Arc {int+ arc;int+ capacity;int+ utilization;int+ length;
            int+ oid1;int+ oid2;float+ cost;}; // Cable map
struct Fiber {int+ fiber;int+ length;int+ oid1;int+ oid2;};
struct Fiber_Cable {int+ fiber;int+ cable;};
/* Artificial structures */
struct Demand {int+ demand;int+ oid1;int+ oid2;int+ volume;};
struct Route {int+ demand;int+ capacity;};
struct Route_Arc {int+ route;int+ arc;int+ demand;};
/* Database connection */
DBconnection bdNet("mssql", "sa//bdNet/TRUTA");
{Node} node from DBread(bdNet, "select cd_oid,capacity_oid,utilization_oid
                                from oid order by cd_oid ");
{Arc} arc from DBread(bdNet, "select cd_cable,capacity_cable,utilization_cable,
                                length_cable,cd_oid_1,cd_oid_2,vl_cost
                              from Cable order by cd_cable ");
DBexecute(bdNet, "create table #demand(demand integer identity,oid1    integer NOT NULL,
                                oid2    integer NOT NULL,volume  integer NOT NULL) ");
DBexecute(bdNet, "insert into #demand (oid1,oid2,volume)
                  select SWDM_DIO1.cd_oid,SWDM_DIO2.cd_oid,count(Fiber.cd_fiber)
                  from Fiber,SWDM_DIO SWDM_DIO1, SWDM_DIO SWDM_DIO2
                  where Fiber.cd_swdm_1 = SWDM_DIO1.cd_swdm and
                        Fiber.cd_swdm_2 = SWDM_DIO2.cd_swdm
                 group by SWDM_DIO1.cd_oid,SWDM_DIO2.cd_oid
                 order by SWDM_DIO1.cd_oid,SWDM_DIO2.cd_oid ");
{Demand} demand from DBread(bdNet, "select * from #demand ");
DBexecute(bdNet, "drop table #demand ");
DBexecute(bdNet, "create table #fiber(fiber integer nodeT NULL,length integer nodeT NULL,
                                oid1 integer nodeT NULL,oid2 integer nodeT NULL) ");
DBexecute(bdNet, "insert into #fiber (fiber,length,oid1,oid2)
                  select Fiber.cd_fiber,Fiber.length_fiber,SWDM_DIO1.cd_oid,SWDM_DIO2.cd_oid
                  from Fiber,SWDM_DIO SWDM_DIO1, SWDM_DIO SWDM_DIO2
                  where Fiber.cd_swdm_1 = SWDM_DIO1.cd_swdm and
                        Fiber.cd_swdm_2 = SWDM_DIO2.cd_swdm order by Fiber.cd_fiber ");
{Fiber} fiber from DBread(bdNet, "select * from #fiber ");
DBexecute(bdNet, "drop table #fiber ");
```

```
{Fiber_Cable} fiber_cable from DBread(bdNet, "select cd_fiber,cd_cable
                                                from Fiber_Cable where st_fiber = 'P' order by cd_cable
");
/* Routes generation */
int+ routes_por_demand = 3; // (direct path + 1 hop + 2 hops) = 3 routes
int+ max_routes = routes_por_demand*card(demand);
int+ arcs_por_demand = 6; // (direct path + 1 hop + 2 hops) = 6 arcs
int+ max_route_arcs = arcs_por_demand*card(demand);
range qtd_routes 1..max_routes;
Route  route[qtd_routes]; // Each demand has 3 possible routes
range     qtd_route_arcs 1..max_route_arcs;
Route_Arc route_arc[qtd_route_arcs]; // All arcs that compose routes
int+ ind_route = 1;
int+ ind_route_arc = 1;
int+ flag = 1;
int+ oid_org = 0;
int+ oid_dst = 0;
initialize {
  forall(ra in qtd_route_arcs) {route_arc[ra].route = 0;route_arc[ra].arc = 0;route_arc[ra].demand = 0;};
  forall(ordered d in demand) {
    forall(i in 1..routes_por_demand) {
      ind_route = i + routes_por_demand*(d.demand - 1);
      route[ind_route].demand    = d.demand;
      route[ind_route].capacity = 1000; // Quem limita é a cap dos arcs
    }; // <demand,capacity>
    forall(i in 1..arcs_por_demand) {
      // Escolhe as routes dos arcs
      if i = 1 \/ i = 2 then ind_route = i + routes_por_demand*(d.demand - 1) endif;
      if i = 4 then ind_route = (i-1) + routes_por_demand*(d.demand - 1) endif;
      // Choosing arcs into routes
      if i = 1 then
        forall(a in arc: ((d.oid1 = a.oid1 & d.oid2 = a.oid2) \/ (d.oid1 = a.oid2 & d.oid2 = a.oid1))){
          route_arc[ind_route_arc].route    = ind_route;
          route_arc[ind_route_arc].arc    = a.arc;
          route_arc[ind_route_arc].demand = d.demand;
          ind_route_arc = ind_route_arc + 1; }
      endif; // direct arc
      flag = 1;
      if i = 2 then
        forall(a1 in arc: (d.oid1 = a1.oid1 \/ d.oid1 = a1.oid2) & (a1.arc <> route_arc[ind_route_arc-1].arc)){
          if d.oid1 = a1.oid1 then oid_org = a1.oid2 else oid_org = a1.oid1 endif;
          forall(a2 in arc: (oid_org = a2.oid1 \/ oid_org = a2.oid2) & (d.oid2  = a2.oid2 \/ d.oid2  = a2.oid1) &
                    (a2.arc <> route_arc[ind_route_arc-1].arc)){
            if flag = 1 then route_arc[ind_route_arc].route = ind_route endif;
            if flag = 1 then route_arc[ind_route_arc].arc = a1.arc endif;
            if flag = 1 then route_arc[ind_route_arc].demand = d.demand endif;
            if flag = 1 then ind_route_arc = ind_route_arc + 1 endif;
            if flag = 1 then route_arc[ind_route_arc].route = ind_route endif;
            if flag = 1 then route_arc[ind_route_arc].arc = a2.arc endif;
            if flag = 1 then route_arc[ind_route_arc].demand = d.demand endif;
            if flag = 1 then ind_route_arc = ind_route_arc + 1 endif;
            if flag = 1 then flag = 0 endif;
          }
        }
      endif; // 1 hop arcs
```

```
flag = 1;
        if i = 4 then
          forall(a1 in arc: (d.oid1 = a1.oid1 \/ d.oid1 = a1.oid2) & (a1.arc <> route_arc[ind_route_arc-3].arc)){
            if d.oid1 = a1.oid1 then oid_org = a1.oid2 else oid_org = a1.oid1 endif;
            forall(a2 in arc: (oid_org = a2.oid1 \/ oid_org = a2.oid2) & (d.oid2  <> a2.oid2 & d.oid2  <>
a2.oid1) &
                          (a2.arc <> route_arc[ind_route_arc-3].arc)){
              if oid_org = a2.oid1 then oid_dst = a2.oid2 else oid_dst = a2.oid1 endif;
              forall(a3 in arc: (oid_dst = a3.oid1 \/ oid_dst = a3.oid2) & (d.oid2  = a3.oid2 \/ d.oid2  = a3.oid1)
&
                            (a3.arc <> route_arc[ind_route_arc-3].arc)){
                if flag = 1 then route_arc[ind_route_arc].route = ind_route endif;
                if flag = 1 then route_arc[ind_route_arc].arc = a1.arc endif;
                if flag = 1 then route_arc[ind_route_arc].demand = d.demand endif;
                if flag = 1 then ind_route_arc = ind_route_arc + 1 endif;
                if flag = 1 then route_arc[ind_route_arc].route = ind_route endif;
                if flag = 1 then route_arc[ind_route_arc].arc = a2.arc endif;
                if flag = 1 then route_arc[ind_route_arc].demand = d.demand endif;
                if flag = 1 then ind_route_arc = ind_route_arc + 1 endif;
                if flag = 1 then route_arc[ind_route_arc].route = ind_route endif;
                if flag = 1 then route_arc[ind_route_arc].arc = a3.arc endif;
                if flag = 1 then route_arc[ind_route_arc].demand = d.demand endif;
                if flag = 1 then ind_route_arc = ind_route_arc + 1 endif;
                if flag = 1 then flag = 0 endif;
              }
            }
          }
        endif; // 2 hops arcs
      }; // <route,arc,demand>

    }; // Create "max_routes" routes and "max_route_arcs" route_arcs
};
/* Functions */
int f_cap_route[i in qtd_routes] = route[i].capacity;
int f_cap_arc[a in arc] = a.capacity-a.utilization;
{int} f_fibers_da_demand[d in demand] = {f.fiber | f in fiber : d.oid1 = f.oid1 & d.oid2 = f.oid2};
/* Decision variables */
var int x[arc] in 0..1,int routeflow[i in qtd_routes] in 0..f_cap_route[i];
/* Linear programming model */
minimize sum(a in arc) a.length * a.cost * x[a]
subject to {
  forall(d in demand)
    sum(i in qtd_routes : d.demand = route[i].demand)
      routeflow[i] = d.volume; // Demand satisfaction
  forall(a in arc)
    sum(i in qtd_route_arcs : a.arc = route_arc[i].arc)
      routeflow[route_arc[i].route] <= f_cap_arc[a] * x[a]; // Arc capacities
};
/* Database update -> Fiber_Cable table */
{Fiber_Cable} proc_routing = {<f.fiber,a.arc> | a in arc & f in fiber & ra in qtd_route_arcs & d in demand :
                routeflow[route_arc[ra].route] > 0 & route_arc[ra].demand = d.demand &
                f.fiber in f_fibers_da_demand[d] & f.fiber < first(f_fibers_da_demand[d]) +
                routeflow[route_arc[ra].route] & route_arc[ra].arc = a.arc & x[a] = 1 };
DBupdate(bdNet, "delete from Fiber_Cable where cd_fiber = ? and cd_cable = ? ")(fiber_cable);
DBupdate(bdNet, "insert into Fiber_Cable(cd_fiber,cd_cable,st_fiber) values (?,?,'P') ")(proc_routing );
```

Figura 7 – Um Exemplo de Arco-Caminho Usando OPLScript

VI. ROTEAMENTO DE VC-4S SOBRE ANÉIS STM-N

Este problema pode ser modelado como um problema de fluxo multiproduto nó-arco, como descrito abaixo:

Rede de Telecomunicações	Modelado como ...
VC-4	Produtos
STM-n	Arcos
SDxC 4/4	Nós

O programa OPLScript que faz o mapeamento do modelo matemático nó-arco descrito na seção IV é apresentado na Figura 8. Observe que o script usa conexões com o banco de dados de modo a obter dados da rede real.

```
/* Database structures */
struct Node {int+ node;int+ capacity;int+ utilization;};// SDxC44 map
struct Arc {int+ arc;int+ capacity;int+ utilization;int+ org;int+ dst;float+ cost;};// STMn map
/* Artificial structures */
struct Demand {int+ demand;int+ org;int+ dst;int+ volume;};
struct Ring {int+ ring;int+ capacity;int+ utilization;};
struct Ring_Arc {int+ ring;int+ arc;};
/* Database connection */
DBconnection bdNet("mssql", "sa//bdNet/TRUTA");
{Node} node from DBread(bdNet, "select cd_sdxc44,capacity_sdxc44,utilization_sdxc44
                                from SDXC44 order by cd_sdxc44 ");
DBexecute(bdNet, "create table #STMN(node_seqn_stmn    integer  identity,
                                     cd_stmn         integer  NOT NULL,
                                     capacity_stmn smallint NOT NULL,
                                     utilization_stmn smallint NOT NULL,
                                    cd_sdxc44_1     integer  NOT NULL,
                                    cd_sdxc44_2     integer  NOT NULL,
                                     vl_cost        float    NULL,
                                    id_status       char(1)  NULL) ");
DBexecute(bdNet, "create table #RING45_STMN(cd_ring45 integer  NOT NULL,
                                            cd_stmn   integer  NOT NULL) ");
DBexecute(bdNet, "execute sp_load_arc ");
{Arc} arc from DBread(bdNet, "select cd_stmn,capacity_stmn,utilization_stmn,
                                cd_sdxc44_1,cd_sdxc44_2,vl_cost
                              from #STMN ");
DBexecute(bdNet, "create table #demand(demand integer  identity,
                                       origin  integer  NOT NULL,
                                       destination integer  NOT NULL,
                                        volume  smallint NOT NULL) ");
DBexecute(bdNet, "insert into #demand (origin,destination,volume)
          select table1.cd_sdxc44,
                 table2.cd_sdxc44,
                 VC4.qt_demand
          from VC4,SDXC41_SDXC44 table1,SDXC41_SDXC44 table2
          where VC4.cd_sdxc41_1 = table1.cd_sdxc41 and
                VC4.cd_sdxc41_2 = table2.cd_sdxc41
          order by VC4.cd_vc4 ");
{Demand} demand from DBread(bdNet, "select * from #demand ");
```

```
{Ring} ring from DBread(bdNet, "select cd_ring45,capacity_ring45,utilization_ring45
                                from RING45 order by cd_ring45 ");
{Ring_Arc} ring_arc from DBread(bdNet, "select cd_ring45,cd_stmn from #RING45_STMN ");
DBexecute(bdNet, "drop table #demand ");
DBexecute(bdNet, "drop table #RING45_STMN ");
DBexecute(bdNet, "drop table #STMN ");
/* Initializations */
int supply[node,demand];
// Indicators to represent whether a node is supply, sink, or intermediate
initialize {
   forall (n in node & d in demand : n.node = d.org) supply[n,d] = -1;
   forall (n in node & d in demand : n.node = d.dst) supply[n,d] =  1;
   forall (n in node & d in demand : n.node <> d.org & n.node <> d.dst) supply[n,d] = 0;};
/* Functions */
int+ capacity[a in arc] = a.capacity-a.utilization;
int+ volume[d in demand] = d.volume;
/* Decision variables */
var float+ traffic[arc,demand],int+ x[arc] in 0..1,int+ y[ring] in 0..1;
/* Linear programming model */
minimize
  sum(a in arc) a.cost * x[a]
subject to {
  forall(n in node & d in demand)
    sum(a in arc : n.node = a.dst) traffic[a,d] -
    sum(a in arc : n.node = a.org) traffic[a,d] = supply[n,d] * volume[d]; // Flow conservation
  forall(a in arc)
    sum(d in demand) traffic[a,d] <= capacity[a] * x[a]; // Arc capacities
  forall(aa in ring_arc & r in ring & a in arc :
         aa.ring = r.ring & aa.arc = a.arc)
    x[a] = y[r]; // Ring assignments
};
display(a in arc, d in demand : traffic[a,d] > 0.1) traffic[a,d]; display x;display y;
```

Figura 8 - Um Exemplo de Nó-Arco Usando OPLScript

VII. CONCLUSÃO

Este artigo apresenta dados e modelos de otimização para o projeto de redes de telecomunicações multicamadas. Baseado em um único esquema de banco de dados e em modelos de fluxo em redes multiproduto foi mostrado como obter boas soluções para os problemas de projeto de rede. Esta abordagem está sendo utilizada como o *kernel* de uma ferramenta de apoio a decisão para planejamento & projeto de redes, atualmente em desenvolvimento.

A principal contribuição da arquitetura baseada em banco de dados e em modelos de otimização é a de fornecer flexibilidade e confiabilidade (por causa do suporte do banco de dados) e desenvolvimento rápido dos modelos de otimização. À medida que a arquitetura proposta for sendo utilizada é possível criar uma biblioteca de modelos conhecidos e usá-la em diferentes problemas de projeto.

O próximo passo é incorporar a este *kernel* da "ferramenta de suporte na decisão de planejamento & projeto de redes", um ambiente gráfico, de maneira a simplificar a análise das soluções por parte dos projetistas de redes.

REFERÊNCIAS

[1] Ahuja R. K., Magnanti T. L., Orlin J. B. "Network Flows Theory - algorithms and applications", Prentice Hall, Englewood Cliffs, New Jersey 1993

[2] Bortolon S. "Planejamento Otimizado de Redes de Transporte na Hierarquia Digital Síncrona". Dissertação de Doutorado, FEE - UNICAMP, Setembro de 1996.

[3] Garcia A. S. "Planejamento do Entroncamento em Redes Telefônicas Urbanas em Processo de Digitalização". Dissertação de Doutorado, FEE - UNICAMP, 1987.

[4] Guerra S. "Um Modelo de Alocação de Canais E1 em Containers SDH". Tese de Mestrado, PPGEE, Depto. de Engenharia Elétrica – UFES, Julho de 1999.

[5] Northern Telecon, "Synchronous Transmission Systems", Nortel (Northern Telecon) Limited, 1995.

[6] Rapp Y. "Planning of Junction Network in a Multi-Exchange Area: General Principles". Ericsson Technics, vol.20, nº 1, 1964, pp. 77-130.

[7] Santos I. P. "A Primal Partitioning Approach to

the Minimum Cost Multicommodity Network Flow Problem", Tese de Mestrado, PPGI, Depto. de Informática – UFES, Agosto de 1999.
[8] Silva R. P. "Uma Metodologia para Alocação de Canais E1 (2Mbps) em Virtual Containers VC-4 em Redes Metropolitanas". Tese de Mestrado, PPGEE, Depto. de Engenharia Elétrica – UFES, Dezembro de 1999.
[9] Trevisol A. G. “Desenvolvimento de uma Base de Dados para Redes Multicamadas” Monografia de Graduação, Colegiado de Ciências da Computação, Depto. de Informática – UFES, Setembro de 1999.
[10] Zanandrea, G.B; Garcia, A.S. “A Heuristic Approach for SDH Transmission Network Evolution”, IV International Conference on Telecommunication Systems, pp. 620-626, Nashville, TN, USA, Março/96
[11] Moura, A. F.; Garcia, A.S. “A Heuristic Procedure for Multilayer Network Planning”, 5th INFORMS TELECOM, Boca Raton, USA, Março/2000
[12] Pereira, J.; Garcia, A.S.; Bortolon, S. “Optimizing the Routing/Allocation of E1 into VC-4 in SDH Netwotks”, First International Workshop on the Design of Reliable Communication Network, Brugge, Belgium, pp., Maio 1998
Wu, Tsong-Ho. Fiber Network Service Survivability, 1992, Artech House

Leandro Altoé Rodrigues é mestrando em Engenharia Elétrica (Automação) pela Universidade Federal do Espírito Santo (UFES). Graduou-se como Engenheiro de Computação na UFES em 1999. Como aluno de graduação, desenvolveu atividades de pesquisa e desenvolvimento nas áreas de análise e modelagem de problemas de otimização em redes de fluxo multiproduto e no desenvolvimento de um software de rede para um ambiente de tempo real.
E-mail: lar@inf.ufes.br

Saulo Bortolon é Engenheiro Civil, graduado pela Universidade Federal do Espírito Santo (UFES) em 1987. É Mestre e Doutor em Engenharia Elétrica pela Universidade Estadual de Campinas (UNICAMP) em 1991 e 1996. Ocupa o cargo de Professor do Departamento de Informática da UFES desde 1989, com interesse nas áreas de aplicações de otimização em telecomunicações, aplicações de sistemas de informação georeferenciadas em telecomunicações e aplicações de informática em saúde.
Email: bortolon@inf.ufes.br

Prof. Dr. Anilton Salles Garcia é Engenheiro Mecânico pela Universidade Federal do Espírito Santo (UFES), graduado em 1976. É Mestre em Otimização e Pesquisa Operacional pela Universidade Estadual de Campinas (UNICAMP) em 1978 e Doutor em Engenharia Elétrica: Automação pela UNICAMP em 1987. Foi pesquisador por 05 (cinco) anos do Projeto UNICAMP/CPqD - Redes Digitais e Engenheiro de Projetos Sênior por 04 (quatro) anos da ELEBRA TELECOM Ltda, onde desenvolveu atividades de pesquisa e desenvolvimento e consultoria técnica especializada para empresas operadoras, nas áreas de Engenharia de Tráfego, Avaliação e Projeto de Sistemas de Comutação e Planejamento de Redes Telefônicas. Foi Consultor do Contrato UNICAMP/TELESP – Metodologia de Planejamento da Transmissão, no desenvolvimento de modelos e ferramentas computacionais para o planejamento do enfeixamento (bundling) em redes SDH. É Professor Adjunto (Licenciado) da UFES e Professor do Mestrado em Telecomunicações do INATEL. As principais áreas de atuação incluem Modelagem, Otimização e Projeto de Redes SDH, Modelagem de Tráfego e Avaliação de Desempenho de Redes, Análise e Projeto de Redes, Redes Multicamadas, Sistemas Distribuídos (CORBA), Redes Inteligentes e Arquiteturas de Gerência de Redes.
E-mail: anilton@inf.ufes.br
Tel: +55 021 27-3352644, Fax: +55 021 27-3352737